Fiche PP 6 : Gestion du calcul d'échéances des demandes en PHP

Kern Louis

BTS Services informatiques aux organisations
Session 2020-2021

E6 – Parcours de professionnalisation
Coefficient

Description d'une situation professionnelle

Épreuve ponctuelle ~~Contrôle en cours de formation~~

~~Parcours SISR~~ Parcours SLAM

NOM et prénom du candidat : KERN Louis **N° candidat :** 0319811520

Contexte de la situation professionnelle
Dans le cadre du projet d'amélioration du système d'envoi de demandes au Master Data Manager. Un problème soulevé par cette dernière est que le temps restant pour traiter une demande continue de s'écouler en dehors des heures de travails et pendant le weekend. Une modification du calcul des échéances est envisagée.

Intitulé de la situation professionnelle
Gestion du calcul de l'échéance des demandes en PHP

Période de réalisation : 3ème semestre 2021 Lieu : Phalsbourg

Modalité : ~~Equipe~~ Individuelle

Conditions de réalisation
Alteration du calcul de l'échéance des demandes prenant en compte les weekends.

Productions associées
Le travail sera suivi par la Master data Manager et ses associés pour s'assurer qu'il respecte le cahier des charges, et une fois validé, il sera ensuite mis en production sur les serveurs du réseau FM Logistic.

Compétences mises en œuvre : A.1.1.1 à 3, A.1.2.2 à 5, A.1.3.1, A.1.4.2, A.2.1.2, A.2.2.1, A.2.3.2, A.3.2.2, A.4.1.1, A.4.1.3 à 8, A.4.2.1 et 3, A.5.2.1 et 3 et 4.

Fiche PP 6 : Gestion du calcul d’échéances des demandes en PHP

Kern Louis

Introduction :

Dans le cadre du projet d'amélioration du système d’envoi de demandes au Master Data Manager. Un problème soulevé par cette dernière est que le temps restant pour traiter une demande continue de s’écouler en dehors des heures de travails et pendant le weekend. Il faudrait alors revoir le calcul de l’échéance d’une demande à sa création afin qu’elle prenne en compte, dans un premier temps, les weekends, et, dans un second temps, les heures de travails.

Condition de réalisation :

Environnement de développement
Logiciels : Visual Studio Code, MySQL Workbench

Langages : PHP, HTML, CSS, SQL

Calcul de l’échéance d’une demande :

Lorsque l’utilisateur créé une demande, une échéance lui est automatiquement attribué selon le Service Level Agreement (SLA) établis par le Master Data Manager et son/ses clients. Ses données sont modifiables dans le profil du Master Data Manager si jamais les SLA viendraient à changer :

Dans notre cas de figure actuel, on donne une échéance pour passer les demandes de l’état « nouveau » à l’état « en cours » (de traitement) de 48h, et une fois les demandes en cours de traitements, on donne une échéance de 24h pour pouvoir la traiter (passage de l’état « en cours » à « traitée »).

Fiche PP 6 : Gestion du calcul d'échéances des demandes en PHP

Kern Louis

Ces valeurs sont postées en base de données via un formulaire (qui est présent sur la vue permettant de paramétrer les SLA) et sont stockées dans une table spécifique :

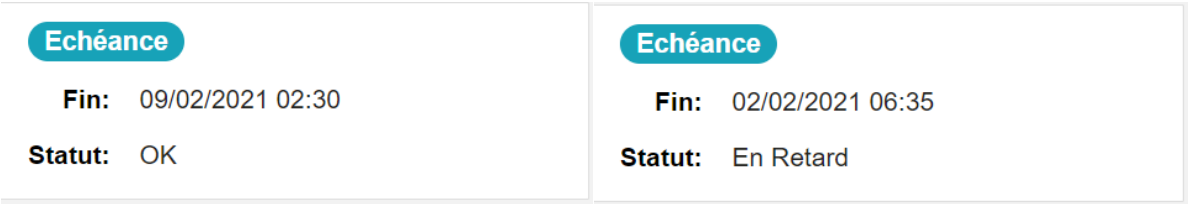
	id	nom	valeur
▶	1	SLA_En_Cours	24
	2	SLA_Nouveau	48
*	NULL	NULL	NULL

Cela va nous permettre de réutiliser ses valeurs pour plus tard.

Une fois la demande créé, l'échéance est automatiquement calculée est est affichée comme ceci sur le tableau affichant toutes les demandes :

A gauche, une demande lui restant 24h pour pouvoir la traiter dans les temps, et à droite, une demande, très largement en retard sur son traitement.

Lorsque l'on clique sur une demande afin d'en afficher les détails nous avons aussi une indication sur l'échéance de celle-ci :



Même schéma que l'exemple d'au-dessus, la différence est que cette fois nous avons précisément la date et l'heure d'échéance (ici -> Fin) de la demande en question.

Cette première version du système ne prenait pas encore en compte les horaires de travail du Master Data Manager, plus précisément, les week-ends, en effet, si une demande était formulée par exemple un vendredi, en prenant une SLA de 48h pour la passer à « en cours », elle serait considérée en retard d'ici le lundi. Le but est donc dans notre exemple de rajouter 48h au compteur pour prendre en compte le passage du week-end (nous n'avons pas pris en compte les horaires de travail du Master Data Manager en dehors du vendredi faute de temps).

Fiche PP 6 : Gestion du calcul d'échéances des demandes en PHP

Kern Louis

Pour la partie code, nous avons pratiquement toutes les méthodes qui ne sont pas du routing ou des contrôleurs de base de données en un seul fichier, Echeance.php qui se présente de cette manière :

```php
class Echeance extends Model
{

    protected $table = 'echeances';

    protected $fillable = ['type', 'etat', 'fin', ];

    public function getrequest()
    {
        return $this->belongsTo('App\Request', 'req_id');
    }

    public function getTimeLeft()
    {
        if ($this->type == "En Attente")
        {
            $sla = Parametre::getValeur('SLA_Nouveau');
        }
        if ($this->type == "En Cours")
        {
            $sla = Parametre::getValeur('SLA_En_Cours');
        }
        $datefin = Carbon::parse($this->debut)
            ->addHours($sla);
        $duree = $datefin->diffForHumans();
        return $duree;
    }
```

La méthode getrequest qui permet d'accéder à la requête en question en utilisant son identifiant. getTimeLeft, est une méthode, comme son nom l'indique, qui permet de déterminer le temps restant à une demande en fonction de son état (ce qui va déterminer quelle valeur de SLA choisir pour le calcul), en récupérant l'heure de création de la demande et en y ajoutant la valeur du SLA en question (en convertissant l'heure en valeur entière afin de pouvoir faire le calcul).

Fiche PP 6 : Gestion du calcul d'échéances des demandes en PHP

Kern Louis

Nous avons ensuite la méthode getStatus qui va nous permettre de déterminer l'affichage de l'échéance sur le tableau comme ceci :

Elle est appelée lorsque l'on affiche le tableau afin d'attribuer les bonnes couleurs ainsi que le bon message en fonction de la valeur retournée. Les autres couleurs sont orange pour indiquer une échéance proche et rouge pour indiquer un retard.

```php
if ($this->type == "En Attente")
{
    $sla = Parametre::getValeur('SLA_Nouveau');
}
if ($this->type == "En Cours")
{
    $sla = Parametre::getValeur('SLA_En_Cours');
}
$datefin = Carbon::parse($this->debut)
    ->addHours($sla);
$duree = Carbon::now()->diffInHours($datefin, false);
if ($duree >= 12)
{
    //si il reste + de 12h
    return 1;
}
else if ($duree > 0)
{
    // il reste - de 12h
    return 2;
}
else
{
    // c'est dépassé
    return 3;
}
```

Comme dans getTimeLeft, on identifie quelle SLA utiliser, puis en comparant avec la date actuelle on renvoie ensuite une valeur correspondante a l'un des trois cas de figures.

Fiche PP 6 : Gestion du calcul d'échéances des demandes en PHP

Kern Louis

Nous avons ensuite la méthode getLimit, qui va nous permettre de traiter le cas où la demande est formulée en fin de semaine et où l'échéance serait due pendant le week-end :

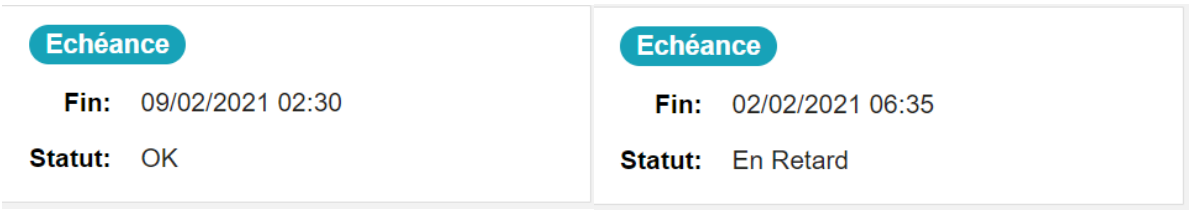
```php
public function getLimit()
{
    //fonction qui calcule les dates de fin
    if ($this->type == "En Attente")
    {
        $sla = Parametre::getValeur('SLA_Nouveau');
    }
    if ($this->type == "En Cours")
    {
        $sla = Parametre::getValeur('SLA_En_Cours');
    }

    $datedebut = Carbon::parse($this->debut);

    $datefin = $datedebut;

    $i = 0;
    while ($i < $sla)
    {
        $datefin = $datefin->addHour();

        $isWE = $datefin->isWeekend();
        $isFri = $datefin->dayOfWeek == 5 && $datefin->hour >= 17;
        $isMon = $datefin->dayOfWeek == 1 && $datefin->hour < 8;

        if (!$isWE && !$isFri && !$isMon)
        {
            $i = $i + 1;
        }
    }
```

Toujours, nous identifions quelle SLA est appliquée sur la demande et avec l'aide des méthodes déjà présente dans PHP, nous d'abords définir les variables isWe (qui vérifie si on est pendant le week-end) et isFri et isMon (qui détermine si nous sommes entre vendredi à partir de 17h jusqu'au lundi suivant avant 8h). A partir de ça, tant que le compteur i est inférieur au SLA indiqué et qu'on ne soit pas dans un cas où nous sommes en pleins week-end entre vendredi 17h et lundi 8h, on ajoute une heure à l'échéance.
On retourne ensuite la valeur finale.

Fiche PP 6 : Gestion du calcul d'échéances des demandes en PHP

Kern Louis

Enfin nous avons la méthode getEtat, qui nous permet simplement de déterminer si la demande a été traitée dans les temps ou non comme ceci :




```php
public function getEtat()
{
    $datefin = $this->getLimit();
    $duree = Carbon::now()->diffInHours($datefin, false);

    if ($duree > 0)
    {
        return "OK";
    }
    else
    {
        return "En Retard";
    }
}
```

On récupère la valeur de getLimit, et avec la valeur retournée, si elle est supérieure à 0, la demande a été traitée dans les temps, dans le cas contraire, elle est en retard.

Conclusion :

Ce stage a été très enrichissant de pars l'utilisation d'éléments nouveau pour moi, dont le traitement des dates en PHP et l'utilisation du Framework Laravel. Aussi, le fait de travailler sur un outil répondant aux besoins d'un « client », qui au fur et à mesure de l'avancement du projet, affine ses besoins, est très intéressant d'un point de vue relationnel et de conduite de projet.