	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

Cahier de spécifications fonctionnelles

PROJET: Kanban



	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

Table des matières

1.	Introduction et cadre du projet	3
I.	Résumé du projet	3
II.	Contexte et intervenant	3
III.	Objectifs	3
IV.	Enjeux	3
V.	Livrables	3
VI.	Critères de succès mesurables	3
VII.	Planning	3
VIII.	Risque lié au projet	3
2.	Schéma de base de données	4
3.	Fonctionnalités	4
I.	Étapes et fonctions	4
	Étape 1 : Compléter le schéma	4
	Étape 2 : Création des modèles	4
	Étape 3 : Route et contrôleurs	5
	Étape 4 : Identifier les requêtes	5
	Étape 5 : Ajout de fonctionnalité	5
	Étape 6 : Améliorer l'application (EN OPTION)	6
	Étape 7 : Déploiement	6
4.	Méthodologie	7
5.	Limitations et contraintes	7
I.	Choix technologique	7
II.	Installation et outils	7
6.	Compétences et savoir-faire visé	8

	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

1. Introduction et cadre du projet

I. Résumé du projet

Créer un site standard qui intègre une application pour gérer des tâches par le biais d'un tableau Kanban et les afficher ensuite sur un calendrier.

II. Contexte et intervenant

Un client souhaite mettre en avant la gestion de projet pour tout public. Pour se faire, il souhaite mettre en ligne un petit site vitrine qui intègre également une petite application Kanban.

III. Objectifs

Créer un site dynamique comportant 4 pages:

- Une page d'accueil
- Une page de contact avec un formulaire
- Une page comportant le Kanban
- Une page comportant le calendrier

Créer une API back-end qui servira à gérer les tâches du Kanban et à remonter les informations pour le calendrier.

Consommer cette API pour afficher les informations sur le front et permettre aux utilisateurs de les manipuler.

Travailler sur le front en termes d'UX pour rendre le tout dynamique et agréable à manipuler.


IV. Livrables

- Code PHP, HTML, JS, CSS (Partie Front-end)
- Code PHP et Base de donnée MariaDB (Partie Back-end)

V. Enjeux

Cette application doit pouvoir aider à organiser les tâches pour les utilisateurs de manière AGILE dans un premier temps et dans un deuxième générer un planning des tâches de manière automatique.

Un deuxième enjeu est d'entraîner le travail sur un projet impliquant plusieurs équipes développant sur des parties bien définies. La communication, l'organisation et la transparence seront essentielles. Une utilisation rigoureuse de Git aussi.

	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

En effet, il va y avoir une équipe sur le back pour la création et implémentation de la base de données ainsi que pour le développement du modèle. Elle recueillera les besoins en fonctionnalités des deux équipes fronts.

Une autre équipe back qui s'occupera de la partie controller. Et s'assurera que les données entre front et back transitent correctement, sont au bon format et calculées au mieux. Elle fera la couche API mise à disposition des équipes front.

Une des équipes front va s'occuper du tableau Kanban et de toutes les fonctionnalités dessus. Création de colonne, de card, drag and drop des card d'une colonne à l'autre, etc. Il lui faut aussi réfléchir avec l'équipe back controller au format de données pour consommer l'API.

Une des équipes front va s'occuper du planning. Afficher les tâches dessus selon la date de début et les deadlines et la personne à qui est assigné la tâche. Il lui faut aussi réfléchir avec l'équipe back controller au format de données pour consommer l'API.

VI. Critères de succès mesurables

Les différentes parties du projet sont connectées et communiquent correctement ensemble.

Le tableau Kanban est utilisable et le planning aussi.

Les différentes pages s'affichent correctement.

Si on veut coder un autre front, cela devrait être possible. Y implémenter aussi la gestion des utilisateurs dans le futur.

VII. Planning

Le projet est sur 6,5 jours, à partir du 16 mai c'est à vous de vous organiser entre équipes pour être dans les délais. Les présentations sont fixées au vendredi 27 mai.


VIII. Risque lié au projet

Une importante coordination de s'effectuer entre la partie front-end et la partie back-end pour l'échange de données. L'utilisation de Git de manière rigoureuse, encore une fois, est indispensable.

2. Organisation des équipes et tâches

Le projet sera composé de 5 équipes. Chaque équipe aura un scrum master qui gérera les tâches à l'interne et l'avancement de son équipe.

Le projet global sera géré par un "super scrum master" aura lui une vision d'ensemble du

	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

projet et sera en lien régulier avec les scrum masters de chaque équipe pour s'assurer de la bonne coordination et communication inter-équipe.

1. Equipe PHP-Template

L'équipe aura la responsabilité de créer une site de 4 page, soit:

- Une page d'accueil qui présente plusieurs blocs avec image et texte
- Une page d'enregistrement avec un formulaire
- Une page laissée "blanche" dans l'équipe JS Kanban va insérer sa réalisation
- Une page laissée "blanche" dans l'équipe JS Calendrier va insérer sa réalisation

Voir plus de détail sur la partie "Fonctionnalité"

2. Equipe JS Calendrier

L'équipe JS Calendrier s'occupera de récupérer l'ensemble des tâches depuis le back-end afin de les afficher sur un calendrier issu d'une librairie. Ce calendrier devra être rajouté sur la page "Calendrier" créée par l'équipe PHP-Template.

3. Equipe JS Kanban

L'équipe JS Kanban devra afficher des tâches sur un Kanban issu d'une librairie. Ces tâches pourront être "bougées" (à l'aide d'un drag&drop) d'une colonne à une autre. Il faut également donner la possibilité de créer, éditer ou supprimer une tâche, ainsi que de créer ou supprimer une colonne. L'ensemble de la gestion des tâches et des colonnes sera lié au back-end. Ce calendrier devra être rajouté sur la page "Kanban" créée par l'équipe PHP-Template.


4. Contrôleur PHP

L'équipe contrôleur PHP devra gérer l'ensemble de la logique du CRUD des tâches et des colonnes et utilisateurs et sera également responsable de faire le lien entre les modèles et le schéma JSON requis par le front.

5. Modèle et base de données

L'équipe modèle et base de données devra réaliser le schéma de base de données, créer les différentes requêtes pour la gestion des tâches et des colonnes et créer les modèles dans le back-end.

3. Fonctionnalités

	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

1. Equipe PHP-Template

Page d'accueil

Créer un tableau associatif, contenant un titre, un texte et un lien vers une image. Ce tableau va permettre d'afficher ces informations sous forme de bloc sur la page d'accueil

Page de contact

Récupérer les informations envoyées du formulaire, envoyer les informations au back-end via la fonction **curlPost()** qui vous est fournie et transmettre une notification par e-mail à l'utilisateur qui l'a transmis (il faudra donc à minima récupérer l'e-mail de l'utilisateur). Pour transmettre l'e-mail, vous pouvez utiliser le serveur SMTP d'un des hébergements d'infomaniak, Google ou autre (La configuration SMTP est la même que lorsque vous configurez la boîte e-mail de votre smartphone).

Header et footer

L'entête et pied de page de chacune des 4 pages ne devra pas être figuré sur chaque fichier HTML mais créé qu'une seule fois et affiché grâce à PHP. Le titre de la page devra figurer et changera selon la page dans laquelle on se trouve.

2. Equipe JS Calendrier

A l'aide de la librairie FullCalendar JS, afficher l'ensemble des tâches, ceci grâce aux champs "date de début" et "date de fin" que contiennent chaque tâche. Ces tâches sont récupérées du back-end. Lorsqu'on clique sur la tâche, une modale permet d'afficher le détail de la tâche (se coordonner avec l'équipe JS Kanban pour l'affichage du détail d'une tâche).

3. Equipe JS Kanban


Créer un Kanban en affichant plusieurs colonnes et des tâches dans chaque colonne. A l'aide de la librairie SortableJS, donner la possibilité de faire du drag&drop d'une tâche d'une colonne à une autre. Au moment du drop, un appel au back-end mettra à jour le statut de la tâche.

Il y aura également la possibilité de modifier les informations d'une tâche ou de l'assigner à quelqu'un d'autre.

Enfin, une tâche et une colonne peuvent être créées ou supprimées. Attention, une colonne ne peut être supprimée uniquement si elle ne contient aucune tâche.

4. Contrôleur PHP

L'équipe contrôleur PHP définit les routes nécessaires à l'application ainsi que les différentes règles de gestion avant de faire appel aux modèles. Il devra également créer les

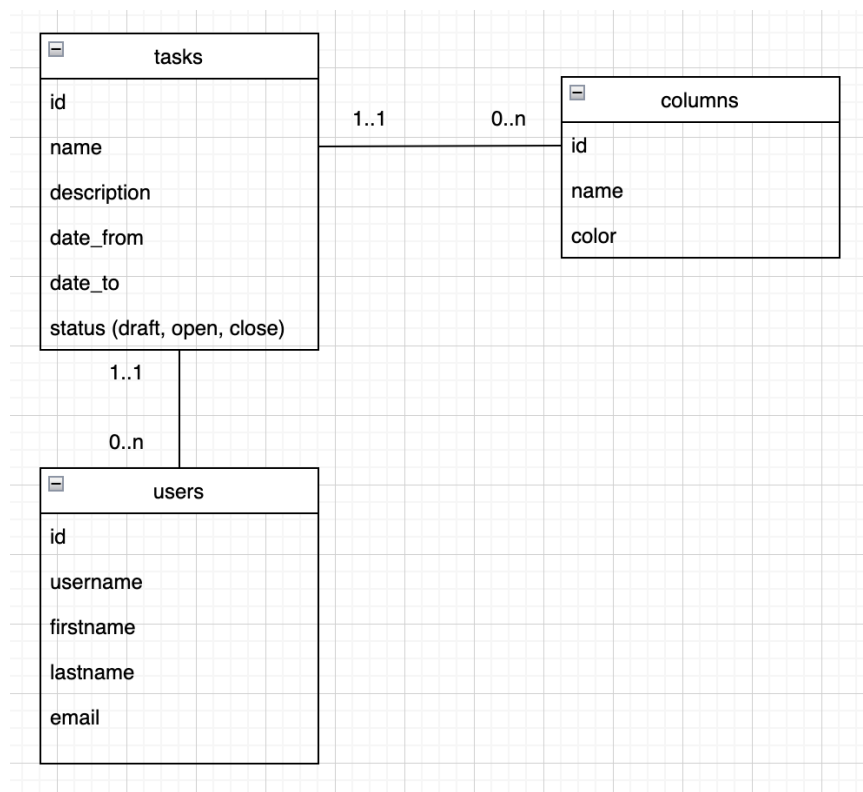
	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF


différentes règles de validation avant l'enregistrement des données et mettre en place un système efficace de gestion d'erreur.

5. Modèle et base de données

Sur la base du schéma suivant, créer la base de données à l'aide de l'UI de PHPMYAdmin. Une fois créé, consulter l'équipe Contrôleur PHP et Equipe JS Kanban pour analyser les différentes requêtes qui leur seront nécessaires. A minima, on aura les requêtes CRUD pour le modèle "tâche" et "colonnes".

4. Schéma de base de données



	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

5. Méthodologie

L'ensemble du projet doit être développé selon la méthode Agile + SCRUM.

La semaine sera organisé avec chaque début de matinée un stand-up :

- Prise de température du groupe
- Planification de la journée
- Partage des tâches
- Identification d'aide et soutien pour les tâches

Puis, à chaque fin de journée, une rétrospection :

- Quels ont été les obstacles (collaboratif ou technique) ?
- Y-a-t-il du retard ?
- Quels sont les succès ?

Chaque développeur·euse aura sa branche. On doit pouvoir remonter celles-ci pour repartir d'une étape en particulier.


6. Limitations et contraintes

I. Choix technologique

- PHP
- MySQL


II. Installation et outils

- Repository GitHub
- Docker avec Apache ou Nginx + PHP-FPM et un hébergement externe

	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

7. Compétences et savoir-faire visé

1. Lire des cahiers des charges/spécifications
2. Appréhender le cycle de vie d'un projet
3. Identifier les tâches dans des cahiers des charges et spécifications et les reporter dans les outils AGILE
4. Utiliser les outils de versioning et de collaboration

	Cahier des spécifications fonctionnelles 10.02.22 <i>- Projet Kanban API -</i>	Rédaction: FL - 10.05.2022
		Validation: EF

8. Fonction utile

```
function curlPost($url, $data = NULL) {

    $ch = curl_init($url);

    curl_setopt($ch, CURLOPT_USERAGENT, 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Safari/537.36');

    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);

    curl_setopt($ch, CURLOPT_TIMEOUT, 5); //timeout in seconds

    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);

    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);

    curl_setopt($ch, CURLOPT_ENCODING, 'identity');

    if (!empty($data)) {

        curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($data));

    }

    curl_setopt($ch, CURLOPT_HTTPHEADER, ["Content-Type"=>"application/json"]);

    $response = curl_exec($ch);

    if (curl_error($ch)) {

        trigger_error('Curl Error:' . curl_error($ch));

    }

    curl_close($ch);

    return $response;

}
```

Exemple d'utilisation:

```
$response = curlPost("http://www.test.ch", ["myField1"=>"myValue1"]);
```