**Transcript détaillé pour la présentation orale en français**

Bonjour à tous. Je vais vous présenter mon travail sur le projet MONA réalisé au cours IFT3150 de ce semestre.

Durant ce projet, j'ai rejoint l'équipe de MONA en tant développeur de serveur. Mon mandat ici est divisé principalement en trois phases que je vais vous détailler.

**Phase 1: Installation et Familiarisation avec la structure du serveur**

Au début de mon mandat, j'ai dû installer et configurer localement le serveur de l'application MONA. Cette étape impliquait l'utilisation de technologies comme Laravel, PHP et Docker. J'ai suivi les instructions transmises par Corelie, qui travaillait précédemment sur ce projet.

Durant cette installation, j'ai rencontré des difficultés liées à la version de PHP. Pour résoudre ces problèmes, j'ai collaboré étroitement avec Simon, développeur serveur chez MONA. Grâce à son aide, j'ai pu configurer toutes les dépendances nécessaires.

Une fois l'environnement opérationnel, Simon m'a guidé dans la migration des bases de données vers mon poste local et m'a expliqué le processus pour effectuer des requêtes API. Ce temps d'installation m'a aussi permis de me familiariser avec le code existant et d'approfondir mes connaissances sur Laravel, Vue.js et Docker.

**Phase 2: Génération de la documentation des APIs**

Après avoir installé l'environnement serveur, j'ai commencé à travailler avec les APIs. Mon intérêt principal portait sur les appels API existants pour les ressources comme Artist, Artwork, Heritage et Place.

J'ai créé un pull request intitulée "Generation of APIs documentation and code linting". L'objectif était de générer automatiquement une documentation claire des routes API existantes en utilisant Scramble, un outil qui facilite la création de documentations OpenAPI à partir du code source.

Pour enrichir cette documentation générée, j'ai ajouté des commentaires au format PHPDoc dans le code. Ces commentaires fournissent des informations supplémentaires essentielles à la compréhension des routes et de leurs fonctionnalités. (Ce travail a pris un temps considérable en raison du nombre important de routes à documenter.)

Voici un aperçu de la documentation qui est maintenant disponible via l'interface à l'URL /docs/api [référence à la capture d'écran].

Après avoir créé ma pull request, j’ai travaillé avec Simon pour réaliser un ≪ rebase ≫ de mon pull request. C’était la première fois que je d´découvrais cette notion avancée de gestion de versions avec Git. Concrètement, j’ai appris à regrouper mes multiples commits initiaux en seulement quelques commits principaux. Cette méthode simplifie considérablement l’historique des modifications apportées.

**Phase 3: Ajout de fonctionnalités CRUD aux routes de l'API**

Suite à l'approbation de mon pull request, j'ai poursuivi mon travail sur l'API. J'ai eu de nombreuses discussions avec Simon et Corelie concernant l'ajout de fonctionnalités CRUD supplémentaires aux routes de ressources API.

Avant mon travail, les routes API courants permettait principalement d'afficher la liste des ressources ou une ressource spécifique. Mon objectif était d'ajouter davantage de fonctionnalités, notamment les opérations store, update et destroy. J'ai compilé la liste des fonctionnalités des routes API dans les contrôleurs de ressources pour avoir une vue d'ensemble de ce qui devait être fait.

Voici les fonctions que j'ai implémentées pour les contrôleurs API de ressources [référence à la capture d'écran].

Il est important de noter que les méthodes create et edit dans Laravel retournent uniquement le formulaire HTML pour la requête, ce qui n'est pas nécessaire pour nos API qui renvoient des données JSON.

Après avoir ajouté les fonctionnalités supplémentaires, j'ai enregistré ces dernières dans la liste des routes du serveur.

(Voici un exemple de la nouvelle fonction store que j'ai implémentée pour l'API des œuvres d'art [référence à la capture d'écran].)

J'ai ensuite créé un nouveau pull request intitulée "Add CRUD features for resource APIs" dans le dépôt mona-server.

**Conclusion**

J’ai beaucoup apprécié prendre part à ce projet, dans une équipe très sympathique et dont le but est intéressant. J’ai beaucoup appris grâce à ce projet, et je pense que ce que j’y ai acquis me servira grandement à l’avenir.

Mes plus grandes difficultés viennent toujours de mon manque d’expérience avec les outils utilisés et du fait que je suis arrivée sur un projet déjà commencé. Il est compliqué de comprendre le fonctionnement du serveur sans avoir eu l’occasion de le configurer moi-même. Il faut toujours que je cherche assez longtemps dans la documentation et dans le code pour comprendre le fonctionnement avant de pouvoir y faire des améliorations et des ajouts. Je suis conscient que j’aurai souvent à travailler sur des projets déjà commencés à l’avenir et je ne sous-estimerai plus l’importance de la documentation.

Pour la suite, quelques tâches seraient intéressantes à réaliser:

* Vérifier les nouvelles fonctions des routes APIs
* Continuer à améliorer la documentation des routes API
* Ajouter des pages de vues manquantes pour les routes dans l'interface admin

Je vous remercie de votre attention.