**Nom du projet : Followers**

1. **Contexte**
2. **Présentation des acteurs du projet**

Pour ce projet, nous avons comme clients, un groupe de trois personnes (composé de ASSALI Antoine, AZEROT Stéphane et RAMANADAPOULLE Fabien) qui espère beaucoup de la prouesse de ses camarades (MANSEUR Aghiles, SAMIER Christophe et ZIDAT Sara) pour répondre à leurs attentes. Afin que ce projet arrive à ses fins, le professeur M. RAYNAL Benjamin sera là pour encadrer tout le monde (dans la joie, la paix et la bonne humeur 😊).

1. **Présentation de l’objectif du projet**

Tout d’abord, il s’agit du projet annuel de la 1ère année du Mastère en Architecture Logiciels à l’ESGI où les élèves sont encadrés et notés par leur professeur d’algorithme avancée. Chaque année, il y a un projet dans le but de mettre en application les compétences acquises dans les différentes matières, dans un contexte assez similaire à celui trouvé en entreprise. Pour cela, le client devrait donner deux sujets que le prestataire choisirait sa préférence ensuite. Le sujet est de **pourvoir faire une analyse d’un compte sur un réseau social et donner des suggestions à l’utilisateur**. Le public visé, est une personne ayant un smartphone quel que soit sa connexion internet.

Exemple :

On donne un compte en renseignant la plateforme (twitter etc…), cela renvoit les stats globales du compte (nbr de posts/an, nb de posts hebdo, moyenne de j’aime, les comptes/pages likées) puis, après analyse, permet de donner des pages ou personnalités qu’on pourrait aimer.

1. **Besoins et contraintes liés au projet**
2. **Périmètre du projet**

Le thème du projet annuel est de créer 3 composants interagissant ensemble :

* Un **serveur**
* Un **client lourd**
* Une **application mobile**

Le but de ces programmes est de pouvoir effectuer un traitement coûteux en ressource (nécessitant donc un ordinateur), à partir de n’importe où, suivre la progression des traitements, et pouvoir accéder à tout moment aux résultats obtenus.

**Utilisation**

L’utilisateur souhaite effectuer un calcul lourd (par exemple, déterminer les cas de plagiats dans une base de projets). L’utilisation se fait de la façon suivante :

* Il se connecte via l’application mobile à son compte
* Il indique qu’il souhaite effectuer une nouvelle tâche, et fournit les ressources nécessaires à celle-ci (upload de fichiers, liens github/drive/etc…)
* Les informations sont envoyées au serveur
* Au niveau du serveur, la tâche est mise dans la file d’attente de cet utilisateur
* Au niveau du client lourd, on détecte qu’une nouvelle tâche est disponible sur le serveur, et on récupère toutes les ressources
* Le client lourd effectue le traitement, en indiquant régulièrement sa progression au serveur
* Une fois le traitement terminé, le client lourd envoie le résultat au serveur
* Le serveur notifie l’utilisateur que le traitement est terminé, et qu’il peut visualiser le résultat via l’application

**A.1. Les composants**

**A.1.1. Le serveur**

Le serveur permet de faire le lien entre le client lourd et l’application mobile. C’est sur le serveur que l’on gère :

● la gestion des comptes utilisateurs

● la gestion des tâches en attente

● le stockage des résultats des tâches

La connexion des utilisateurs, que ce soit via l’application mobile ou le client lourd, doit être sécurisée par le moyen de votre choix. Le serveur devra impérativement être réalisé en **JEE**.

**A.1.2. Le client lourd**

Le client lourd permet d'effectuer le calcul coûteux en ressources. Il doit pouvoir :

● se connecter au serveur au démarrage (via un fichier de configuration et/ou des paramètres sur la ligne de commande)

● vérifier à intervalle régulier la présence de nouvelles tâches

● récupérer les données nécessaires au traitement

● effectuer le traitement demandé

● envoyer le résultat du traitement au serveur

Le client peut être réalisé dans le langage de votre choix parmi les suivants : **C, C++, Java, C# ou Python**

**A.1.3. L’application mobile**

L’application mobile sert ici d’interface utilisateur. C’est grâce à elle que l’on peut :

● indiquer de nouvelles tâches à effectuer

● suivre la progression des tâches

● visualiser les résultats

L’application sera réalisée dans un langage de votre choix, parmi les suivants, après concertation entre client et prestataire : **Java, Kotlin, Swift, Flutter.**

**A.1.4. Le traitement lourd**

Le traitement à effectuer devra avoir une partie algorithmique conséquente (il ne faut pas juste faire un milliard de calculs simples dans une boucle). Il pourra s’agir par exemple :

● de trouver les meilleurs couplages (voir Algorithme Hongrois)

● effectuer de la détection de plagiat au sein d’un corpus de textes/codes

● trouver les parties en commun dans un ensemble de textes/musiques/vidéos

● extraire de mesures d’un texte : calcul de redondance, extraction des mots les plus fréquents, et de la, pourquoi pas détecter la thématique

● effectuer des traitements en ligne (recherche de tweets, récupération de ressources, etc…).

**A.1.5. Autres informations**

L’application mobile doit être capable d’avoir l’utilisation de la **géolocalisation, une création de compte ou associer son compte** (Facebook, Google, etc…), et peut-être (dans le meilleur des cas) la notification push et le multi langage.

1. **Analyses fonctionnelles**

Il doit respecter différents scénarii tels que :

**Coté application mobile**

* Un utilisateur télécharge l’application et l’ouvre, il est capable de créer son compte grâce à un pseudo (username), son adresse mail et son mot de passe (password).
* Un utilisateur doit saisir le compte (ou le lien du compte) qui souhaite connaître et choisir la plateforme pour effectuer le traitement.
* Si un compte n’existe pas, un message d’erreur est capable d’indiquer que ce compte n’existe pas.
* Affichage d’un loader montrant la progression du traitement (avec le pourcentage).
* L’utilisateur peut se connecter soit avec le compte qu’il a créé, soit avec un compte (google, facebook, instagram etc…)
* L’utilisateur n’est pas obligé de s’authentifier à chaque fois, il a un délai de 4h, s’il ne s’est pas déconnecté lui-même, à partir de là un pop-up doit l’indiquer qu’il a été déconnecté et qu’il se reconnecte.
* Grâce à un bouton, l’utilisateur est capable d’interrompre le traitement s’il le trouve.
* L’utilisateur peut voir ses informations et le modifier si c’est un compte créé.

**Client lourd**

* Il peut récupérer les demandes de l’utilisateur grâce au serveur.
* Après la demande de l’utilisateur, le client lourd sera capable de chercher des informations du compte sur internet comme le nombre de posts/an, nombre de posts hebdo, moyenne de j’aime, les comptes/pages likées.
* Il peut indiquer l’évolution de l’analyse au serveur.
* Il peut être capable d’analyser les données et donner des recommandations, même selon l’historique des requêtes.

1. **Résultats attendus**
2. **Le contenu**

Le projet doit être opérationnel, l’application est mise en ligne sur Play Store. Au niveau du code, il doit respecter les cours faits cette année comme l’intégration continue, clean code.

Au niveau du graphisme de l’application mobile, il se doit d’être **moderne**, éviter trop de couleurs pour qu’il ne devient pas une application arc-en-ciel et non plus à Desigual (oups désolé les mots sont sortis de ma bouche emoji *visage\_main\_sur\_la\_bouche*). L’application doit avoir un **menu, un logo** et que l’on peut, si le smartphone a la fonctionnalité, de se connecter par **empreinte digitale** pour la sécurité (facultatif). Pour la police, il se doit être **lisible**, utiliser des typographies **Sans Serif.** Une maquette devra être faite pour la validation du client.

1. **Les étapes du projet**

|  |  |
| --- | --- |
| **Date souhaitée** | **Tâches** |
| mi-avril 2020 | Création des maquettes mobiles |
| 20/04/2020 | Proposition d’un logo, si possible, faire trois types de logo |
|  |  |
|  |  |
|  |  |
|  |  |