MANUEL UTILISATEUR

Au démarrage de l'application, il est demandé à l'utilisateur de se connecter pour s'inscrire.

Pour l'instant, nous ne prenons en charge que l'authentification par Google. A l'avenir, nous prendrons en charge davantage de protocoles de connexion.

Une fois connecté, l'utilisateur sera automatiquement redirigé vers l'interface principale de l'application Foody ornée d'une barre de navigation flottante contenant les chemins des trois vues principales soient : la page d'accueil, la page micro et enfin la page des paramètres.

- La page d'accueil permet de mettre en appétit l'utilisateur avec des suggestions aléatoires de recettes qu'il est possible d'explorer d'un simple clic.
- Le cœur de l'application est concentré sur la vue micro. Par un clic, l'utilisateur ouvrira la page de saisie vocale et lors d'un appuie long, il basculera sur la page de saisie vocale. S'il n'est pas déjà sur la vue en question il pourra simultanément lancer l'écoute pour commencer à dire tous les ingrédients qu'il a envie de cuisiner.

À la fin de la saisie vocale, un nouvel appuie sur le micro permettra de pouvoir supprimer des ingrédients partis ceux afficher en cliquant sur la petite croix rouge au-dessus de chacun des ingrédients.

L'utilisateur pourra enfin cliquer sur le bouton « Aller, en cuisine ! » afin de lancer la recherche de recettes contenant le ou les ingrédients de la liste précédemment dictée.

Après une courte page de chargement dont le délai peut varier selon la connexion de chacun, un carrousel horizontal contenant des cartes de recettes sera alors proposé à l'utilisateur. Il peut alors ajouter une recette en favoris sans l'ouvrir.

Une fois la recette sélectionnée, l'utilisateur peut visualiser toutes ses informations (la note reçue par les autres utilisateurs, la durée de préparation et le nombre de portions). Il peut ensuite accéder à une sous-vue scrollable contenant l'intégralité des ingrédients nécessaires à l'élaboration de la recette suivie de la liste de toutes les étapes de préparation.

En dernier lieu, la page des paramètres sert de centre le contrôle de toutes les fonctionnalités annexes comme la liste de favoris, les paramètres de connexion et de notification et contient également les mentions légales (à venir).

LA DOCUMENTATION DE MAINTENANCE ET D'ADMINISTRATION DEVELOPPEUR/ADMINISTRATEUR

La note de cadrage :

Le projet Foody est un projet qui permettra un usage quotidien par les utilisateurs afin de leur permettre de trouver des recettes en fonction des ingrédients qu'ils auront décidés de cuisiner. Il était initialement prévu d'utiliser la reconnaissance visuelle pour reconnaître les ingrédients. Après concertation, il a finalement été décidé d'utiliser la reconnaissance vocale dans un soucis de performance et de facilité d'usage.

Les étapes majeures :

Trouver une API exploitable pour le démarrage répondant aux besoins de Foody et notamment lui permettre de récupérer des recettes lorsque l'on lui soumet une liste d'ingrédients. Il est nécessaire qu'elle soit gratuite ou très abordable pour s'adapter aux fonds limités des initiateurs du projet.

Créer une interface simple et ludique. Le marché des applications liées à l'alimentation est dense mais peu d'entre elles offrent une interface épurée qui permette d'aller à l'essentiel. Il faut se démarquer en proposant quelque chose de différent pour marquer l'esprit du public.

Cette application n'utilisera pas les données des utilisateurs pour gagner de l'argent. Les initiateurs du projet veulent faire preuve d'éthique quant à la manière d'aborder la construction de ce projet.

La projection de chiffres d'affaires se poursuivra à moyen et long terme : nous gardons à l'esprit que c'est avant tout un projet d'étude. Si après une période de beta-test les retours utilisateurs vont en ce sens, nous implémenteront des moyens éthiques de générer des revenus. Plusieurs solutions ont déjà été envisagées :

- Format freemium avec un abonnement permettant de débloquer des options et fonctionnalités.
- Partenariats avec des industriels qui proposeront des recettes sponsorisées valorisant leurs produits.
- Publicités non ciblées sur l'utilisateur mais toujours dans le domaine culinaire.

Spécifications d'exigences fonctionnelles :

- Interface fluide et visuellement agréable
- Saisie d'ingrédients par la voix

- Trouver des recettes qui contiennent ces ingrédients.
- Pouvoir visualiser la recette complète.
- Avoir accès à un espace utilisateur comprenant les réglages de l'application, les paramètres du compte, la liste des favoris et les paramètres de notifications.

Outils de développement de Foody

- L'application Foody en Swift et SwiftUI a été programmée avec l'Environnement de Développement Intégré Xcode préconisé par Apple pour ses plateformes.
- Le scrapper nous servant d'API en Python a été développée en Python avec PyCharm à partir d'un fork d'un projet existant que nous avons adapté à nos besoins.
- Ce scrapper est containérisé avec Docker et déployé sur un serveur NAS de chez QNAP.
- L'identité visuelle de Foody a été élaborée avec Affinity Photo et Affinity Designer.

Stockage de données :

Les données sont stockées dans une base de données Firebase. Nous stockons les informations des utilisateurs nous permettant ainsi de consigner leurs recettes favorites. La partie connexion est entièrement déléguée à Google. Nous n'accédons donc à aucunes données sensibles tels que les mots de passe. Nous implémenterons bientôt d'autres protocoles de connexion.

L'application est compatible avec tous les iPhones sous iOS 13 minimum. Nous envisageons d'adapter l'application prochainement pour iPadOS.

Les rapports d'incidents :

Initialement, nous utilisions Trello pour s'inspirer de l'usage que l'on peut avoir d'un Jira. Mais lorsque l'équipe de développement s'est réduite pour diverses raisons, nous nous sommes contentés de favoriser la communication lors de nos réunions. Et c'est ainsi que nous nous attribuons les tâches. Lorsque Nora a rejoint notre équipe, le ScrumMaster a utilisé Notion (un outil inspiré de l'Agile et qui permet de gérer des petit projets) pour suivre l'évolution du projet.

Lors de la mise en bêta-test via la plateforme Test Flight mise à disposition par Apple, la remontée des bugs sera gérée par celle-ci et un formulaire déjà éprouvé pour cette tâche sera mis à disposition.

EVALUATION ET PERFORMANCE

Nous avons décidé que l'audit se concentrera sur les requêtes réseau du scraper en python ainsi que sur le déploiement sur le serveur. D'une part, parce que la partie connexion est gérée par Firebase et donc bénéficie des protocoles de sécurité éprouvés mis en place par Google et d'autre part, car ce service est le cœur du produit Foody.

Nous avons tout d'abord testé la qualité de code et les potentielles failles de sécurité contenues dans le code et dans la façon dont a été programmé le scraper en python servant d'API qui récupère les données du site Marmiton. Nous avons pour cela utilisé un outil nommé DeepSource qui analyse le code d'un repo GIT et relève les failles de sécurité, la propreté, la qualité et la performance du code. Cet outil n'a décelé aucunes failles en termes de sécurité ou de performance. Il a seulement relevé une légère anomalie sur le non-respect des normes PEP8 s'appliquant au programme en python qui a pu être corrigé immédiatement via une réparation automatique proposée par l'outil.

Ensuite, nous avons pu entamer les tests de performance et de fiabilité s'appliquant aux requêtes http qui transitent vers le service et qui à son tour, envoie des requêtes vers Marmiton afin de récupérer les recettes. Sur un test de cinquante requêtes d'affilées, le temps moyen entre la requête et la réponse varie considérablement, soit entre 100 et 25 000 ms. Lors du début du test, les premières requêtes mettent entre 100 et 200 ms puis progressivement, le délai de traitement et de réponse augmente. Le serveur physique qui héberge ce service bénéficie d'une connexion avec un débit variant de un à deux giga-octets par seconde. Nous avons conclu avec certitude que les défaillances ne viennent pas du débit mais de la puissance de calcul limitée du serveur. Effectivement, le serveur NAS en question possède un processeur Intel Celeron N3150 Quad-Cœur cadencé à 1,6 GHz. En considérant le fait que l'appareil n'est pas un serveur dédié aux services que nous analysons, il ne nous est pas possible de garantir une fiabilité et une stabilité requise pour l'usage du service destiné à Foody tant les résultats varient d'un essai à l'autre.

Pour résoudre cette faille de stabilité et de performance, il faudrait :

- Un serveur plus puissant qui serait dédié à ce service. Cela permettrait d'obtenir des résultats stables et d'adapter le choix matériel à l'usage prévu. C'est une solution onéreuse et non évolutive qui par conséquent ne serait pas scalable si la fréquentation d'utilisation du service augmente de façon considérable. Sachant que le budget moyen d'un serveur à usage professionnel varie plusieurs milliers à plusieurs dizaines de milliers d'euros.
- Ou encore, et c'est certainement la solution la plus viable à court terme, un hébergement du service via un prestataire. Cela offrirait la possibilité d'adapter

les capacités du serveur loué en fonction des besoins évolutif du projet afin d'avoir et proposer un service fiable, stable et sans parasites tiers.

Faille de sécurité

Nous avons ensuite analysé la sécurité des requêtes réseau. Le serveur qui héberge actuellement le service bénéficie d'une protection par certificat SSL non certifié. Cela protège principalement le serveur mais pas les communications qui transitent vers celui-ci. Les requêtes sont donc nues et sans cryptage. Il a été établi que les requêtes qui ne sont pas protégées ne contiennent pas de données utilisateurs sensibles. Le risque se situe donc ailleurs. L'adresse du service étant exposé, il pourrait alors devenir la cible d'une attaque par surcharge, ce qui provoquerait le crash du service ou l'arrêt du serveur hébergeant ce dît service.

Pour corriger cette vulnérabilité, il est donc recommandé à Foody d'investir dans un certificat SSL certifié permettant de protéger et de crypter les requêtes passées vers le service afin de protéger les communication, de prévenir une attaque par surcharge du service et de préserver la machine.

Comme le service est containérisé dans un container Docker, il est possible de brider la puissance de calcul dédiée au container du service en question afin d'éviter la surcharge du serveur mais cela ne réglerait en aucune façon la faille de sécurité des communications non protégées.