Justine CABROL
Parcours : Développeur d'Applications –Java
Projet n°8 : Améliorez votre application avec des systèmes distribués

Tour Guide

Besoins et conception

TOUR GUIDE

BESOINS ET CONCEPTION

1. Présentation du projet

TourGuide est une application permettant aux utilisateurs de voir quelles sont les attractions touristiques à proximité du lieu où ils se trouvent et d'obtenir des réductions sur les séjours d'hôtel ainsi que sur les billets de différents spectacles.
Pour cela TourGuide fait appel au service gpsUtil afin de collecter l'emplacement géographique de l'utilisateur.
Pour que l'application soit efficace, cet emplacement doit être mis à jour régulièrement.
L'application interagit également avec le service RewardsCentral pour obtenir des récompenses pour les attractions via un réseau de fournisseurs.
Elle fait également appel au service TripPricer, lié à différentes agences, qui fournit des offres de séjours personnalisées pour l'utilisateur.

1.1 Objectifs du projet

L'application TourGuide devient très lente lorsque le nombre d'utilisateurs augmente et elle obtient de mauvaises performances.
Le principal objectif est de faire fonctionner l'application avec des performances correctes même lorsque le nombre d'utilisateur connecté simultanément est conséquent (au moins 100 000) et ainsi améliorer de manière significative les temps de réponse, notamment lors de l'appel à gpsUtil et à RewardsCentral.

Quelques bugs de fonctionnement ont été repérés sur l'application, notamment des propositions de séjour ne tenant pas tout à fait compte des préférences de l'utilisateur et une absence de destinations recommandées en fonction de l'emplacement de l'utilisateur.
Le second objectif est donc de proposer des séjours en fonction des préférences de l'utilisateur et de recommander des destinations quelle que soit la localisation de celui-ci.

Certains tests échouent et il faudrait un suivi régulier de ceux-ci, de manière automatisée.
Le dernier objectif est donc de corriger et compléter les tests de l'application et de mettre en place un pipeline d'intégration continue afin de pouvoir assurer un suivi régulier.

1.2 Hors du champ d'application

Analyser les observations des déplacements des utilisateurs pour identifier un schéma logique ou répétitif qui permettra ensuite, si l'analyse est concluante, de proposer de nouvelles suggestions plus ciblées aux utilisateurs.

1.3 Mesures du projet

Les tests permettent de mesurer l'efficacité du projet.

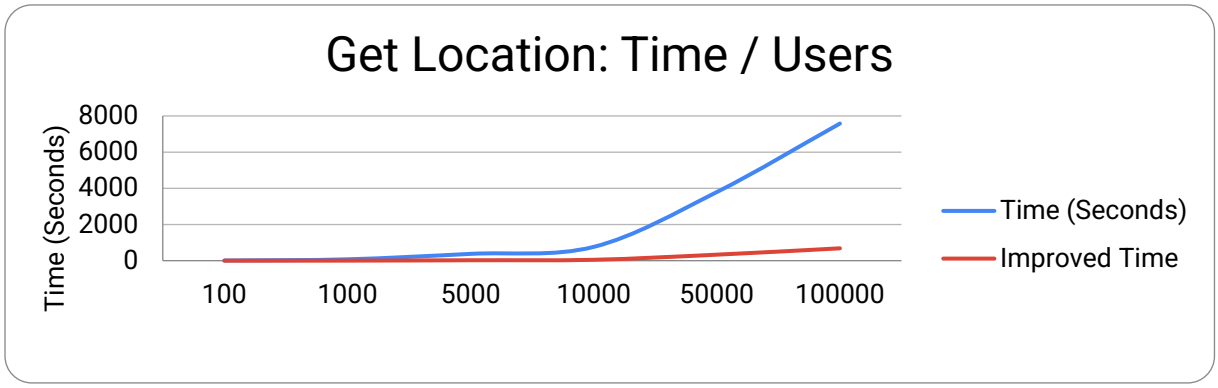
Plusieurs types de tests sont mis en œuvre pour ce projet :

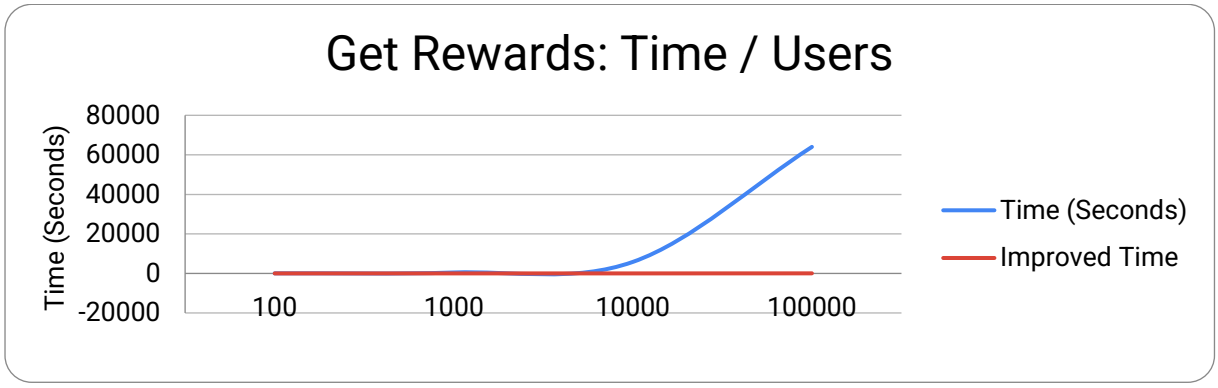
- ***Des tests unitaires*** pour vérifier le bon fonctionnement de chaque méthode, indépendamment des autres, afin de repérer et corriger d'éventuels bugs le plus rapidement possible.
- ***Des tests d'intégration*** pour vérifier la bonne articulation des différents éléments du projet et le bon fonctionnement de l'ensemble.
- ***Des tests de performance*** pour vérifier la rapidité du temps de réponse de l'application et son bon fonctionnement, même en cas de forte affluence.

Les tests sont effectués automatiquement lors de la compilation du projet et lors des commits sur GitLab, grâce à la mise en place d'un pipeline d'intégration continue. Les rapports de tests et de couverture sont générés automatiquement.

Les performances relevées par les tests de performances sont reportées dans un graphique afin d'être rapidement et visuellement observées.

Voici les graphiques actuellement à jour :
(La ligne bleue correspond aux performances relevées avant les modifications apportées à l'application et la ligne rouge correspond aux performances actuelles)





2. Caractéristiques

User story n°1 : Être localisé

Description :

En tant qu' utilisateur de l'application
Je souhaite être localisé
Afin d'obtenir des récompenses et des offres personnalisées.

Critères d'acceptation :

Scénario 1 : Je n'ai jamais encore été localisé

Etant donné que je suis un utilisateur de l'application
Et que je n'ai jamais été localisé

Lorsque la requête « /getLocation » est envoyée

Alors ma position actuelle est calculée par GpsUtil et affichée dans un temps raisonnable (moins de 15 minutes même si le nombre d'utilisateur connecté atteint 100 000 utilisateurs).

Scénario 2 : J'ai déjà été localisé

Etant donné que je suis un utilisateur de l'application
Et que j'ai déjà été localisé

Lorsque la requête « /getLocation » est envoyée

Alors ma dernière position enregistrée est affichée à l'écran.

Scénario 3 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /getLocation » est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu.

User story n°2 : Obtenir les 5 attractions les plus proches

Description :

En tant qu' utilisateur de l'application
Je souhaite obtenir les 5 attractions les plus proches de ma localisation
Afin de choisir une destination.

Critères d'acceptation :

Scénario 1 : Mon nom d'utilisateur est reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur reconnu

Lorsque la requête « /getNearbyAttractions » est envoyée

Alors les 5 attractions les plus proches de ma position sont affichées, quelle que soit la distance qui les sépare de ma position, avec leur nom, leurs coordonnées et le nombre de points qu'elles peuvent me rapporter.

Scénario 2 : Mon nom de l'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /getNearbyAttractions » est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu.

User story n°3 : Consulter les récompenses

Description :

En tant qu' utilisateur de l'application
Je souhaite consulter mes récompenses
Afin de les utiliser.

Critères d'acceptation :

Scénario 1 : J'ai déjà obtenu des récompenses

Etant donné que je suis un utilisateur de l'application
Et que j'ai déjà obtenu des récompenses

Lorsque la requête « /getRewards » est envoyée

Alors mes récompenses sont affichées.

Scénario 2 : Je n'ai pas encore obtenu de récompense

Etant donné que je suis un utilisateur de l'application
Et que je n'ai pas encore obtenu de récompense

Lorsque la requête « /getRewards » est envoyée

Alors une liste vide est affichée.

Scénario 3 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /getRewards » est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu

User story n°4 : Obtenir des récompenses

Description :

En tant qu' utilisateur de l'application
Je souhaite obtenir des récompenses
Afin d'avoir des réductions sur mes séjours.

Critères d'acceptation :

Scénario 1 : Je suis proche d'une nouvelle attraction

Etant donné que je suis un utilisateur de l'application
Et que je suis proche d'une attraction pour laquelle je n'ai pas encore de récompense

Lorsque je suis localisé

Alors de nouvelles récompenses sont calculées et ajoutées à ma liste.

Scénario 2 : Je suis proche d'une attraction pour laquelle j'ai déjà obtenu une récompense

Etant donné que je suis un utilisateur de l'application
Et que je suis proche d'une attraction pour laquelle j'ai déjà obtenu une récompense

Lorsque je suis localisé

Alors il n'y a pas de nouvelle récompense ajoutée.

Scénario 3 : Je ne suis pas proche d'une attraction

Etant donné que je suis un utilisateur de l'application
Et que je ne suis proche d'aucune attraction

Lorsque je suis localisé

Alors il n'y a pas de nouvelle récompense ajoutée.

User story n°5 : Consulter les offres de séjour personnalisées

Description :

En tant qu' utilisateur de l'application
Je souhaite consulter les offres de séjour personnalisées
Afin de choisir un séjour.

Critères d'acceptation :

Scénario 1 : Mes préférences utilisateurs n'ont pas été mises à jour

Etant donné que je suis un utilisateur de l'application
Et que mes préférences utilisateur n'ont pas été mises à jour

Lorsque la requête « /getTripDeals » est envoyée

Alors 5 offres de séjour sont affichées en fonction des préférences par défaut de l'application.

Scénario 2 : Mes préférences utilisateur ont été mises à jour

Etant donné que je suis un utilisateur de l'application
Et que mes préférences utilisateur ont été mises à jour

Lorsque la requête « /getTripDeals » est envoyée

Alors 5 offres de séjour personnalisées sont affichées, en fonction de mes préférences utilisateur.

Scénario 3 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /getTripDeals » est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu

User story n° 6 : Consulter les préférences utilisateur

Description :

En tant qu' utilisateur de l'application
Je souhaite pouvoir consulter mes préférences utilisateur
Afin de les vérifier et éventuellement les mettre à jour.

Critères d'acceptation :

Scénario 1 : Mon nom d'utilisateur est reconnu

Etant donné que je suis connecté à l'application

Lorsque la requête « /getUserPreferences » est envoyée

Alors mes préférences utilisateurs sont affichées à l'écran.

Scénario 2 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /getUserPreferences» est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu

User story n° 7 : Mettre à jour les préférences utilisateur

Description :

En tant qu' utilisateur de l'application
Je souhaite pouvoir mettre à jour mes préférences utilisateur
Afin de recevoir des offres de voyage personnalisées.

Critères d'acceptation :

Scénario 1 : De nouvelles préférences sont saisies

Etant donné que je suis connecté à l'application
Et que j'ai fourni mes nouvelles préférences à la requête

Lorsque la requête « /setUserPreferences » est envoyée

Alors mes préférences utilisateurs sont modifiées et enregistrées
Et elles sont utilisées pour calculer les offres de séjour qui me sont proposées

Scénario 2 : Aucune nouvelle préférence n'est fournie.

Etant donné que je suis connecté à l'application
Et que je n'ai fourni aucune nouvelle préférence

Lorsque la requête « /setUserPreferences » est envoyée

Alors mes préférences utilisateurs ne changent pas.

Scénario 3 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /setUserPreferences» est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu

User story n° 8 : Consulter les données personnelles

Description :

En tant qu' utilisateur de l'application
Je souhaite pouvoir consulter mes informations personnelles
Afin de les vérifier et éventuellement les mettre à jour.

Critères d'acceptation :

Scénario 1 : Mon nom d'utilisateur est reconnu

Etant donné que je suis connecté à l'application

Lorsque la requête « /getUser » est envoyée

Alors mes information personnelles (nom d'utilisateur, email et numéro de téléphone) sont affichées à l'écran.

Scénario 2 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /getUser» est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu

User story n° 9 : Mettre à jour les données personnelles

Description :

En tant qu' utilisateur de l'application
Je souhaite pouvoir mettre à jour mes données personnelles
Afin de les corriger si elles changent.

Critères d'acceptation :

Scénario 1 : De nouvelles données personnelles sont saisies

Etant donné que je suis connecté à l'application
Et que j'ai fourni mes nouvelles données personnelles à la requête

Lorsque la requête « /setUser » est envoyée

Alors mes données personnelles sont modifiées et enregistrées

Scénario 2 : Aucune nouvelle donnée personnelle n'est fournie.

Etant donné que je suis connecté à l'application
Et que je n'ai fourni aucune nouvelle donnée personnelle

Lorsque la requête « /setUser » est envoyée

Alors mes données personnelles ne changent pas.

Scénario 3 : Mon nom d'utilisateur n'est pas reconnu

Etant donné que je suis un utilisateur de l'application
Et que mon nom d'utilisateur n'est pas reconnu

Lorsque la requête « /setUser» est envoyée

Alors un message m'indique que mon nom d'utilisateur n'est pas reconnu

User story n° 10 : Obtenir la localisation de tous les utilisateurs

Description :

En tant que responsable de l'application
Je souhaite obtenir la localisation de tous les utilisateurs
Afin de faire des statistiques pour améliorer l'application.

Critères d'acceptation :

Scénario 1 : Certains utilisateurs n'ont pas encore été localisés

Etant donné que certains utilisateurs n'ont pas encore été localisés

Lorsque la requête « /getAllCurrentLocation » est envoyée

Alors les localisations de tous les utilisateurs s'affichent à l'écran avec leur identifiant
Et la position actuelle des utilisateurs qui n'ont pas encore été localisés est calculée
Et la dernière position enregistrée est affichée pour les autres.

Scénario 2 : Tous les utilisateurs ont déjà été localisés.

Etant donné que tous les utilisateurs ont déjà été localisés

Lorsque la requête « /getAllCurrentLocation » est envoyée

Alors les localisations de tous les utilisateurs s'affichent à l'écran avec leur identifiant
Et la dernière position enregistrée est affichée pour tous les utilisateurs.

User story n° 11 : Ajouter un utilisateur

Description :

En tant que responsable de l'application
Je souhaite ajouter un utilisateur
Afin de lui permettre d'utiliser l'application.

Critères d'acceptation :

Scénario 1 : Le nom d'utilisateur choisi est disponible

Etant donné que le nom d'utilisateur choisi est disponible
Et que j'ai fourni les données personnelles de l'utilisateur à la requête

Lorsque la requête « /createUser » est envoyée

Alors le nouvel utilisateur est créé et enregistré

Scénario 2 : Il n'y a pas de données personnelles.

Etant donné que le nom d'utilisateur choisi est disponible
Et que je n'ai fourni aucune autre donnée personnelle

Lorsque la requête « /createUser » est envoyée

Alors le nouvel utilisateur est créé et enregistré
Et son numéro de téléphone et son e-mail sont mis à « null ».

Scénario 3 : Le nom d'utilisateur choisi est déjà utilisé par un autre utilisateur

Etant donné que le nom d'utilisateur choisi n'est pas disponible

Lorsque la requête « /createUser » est envoyée

Alors un message m'indique que le nom d'utilisateur n'est pas disponible et qu'il faut en choisir un autre.

User story n° 12 : Supprimer un utilisateur

Description :

En tant que responsable de l'application
Je souhaite supprimer un utilisateur
Afin de le retirer de l'application.

Critères d'acceptation :

Scénario 1 : Le nom d'utilisateur est reconnu

Etant donné que le nom d'utilisateur est reconnu

Lorsque la requête « /deleteUser » est envoyée

Alors l'utilisateur correspondant et les informations le concernant sont supprimées de l'application.

Scénario 2 : Le nom d'utilisateur n'est pas reconnu.

Etant donné que le nom d'utilisateur n'est pas reconnu

Lorsque la requête « /deleteUser » est envoyée

Alors un message m'indique que l'utilisateur n'a pas été trouvé et n'a pas pu être supprimé.

3. Solution proposée

3.1 Schémas de conception technique

Diagramme des cas d'utilisation

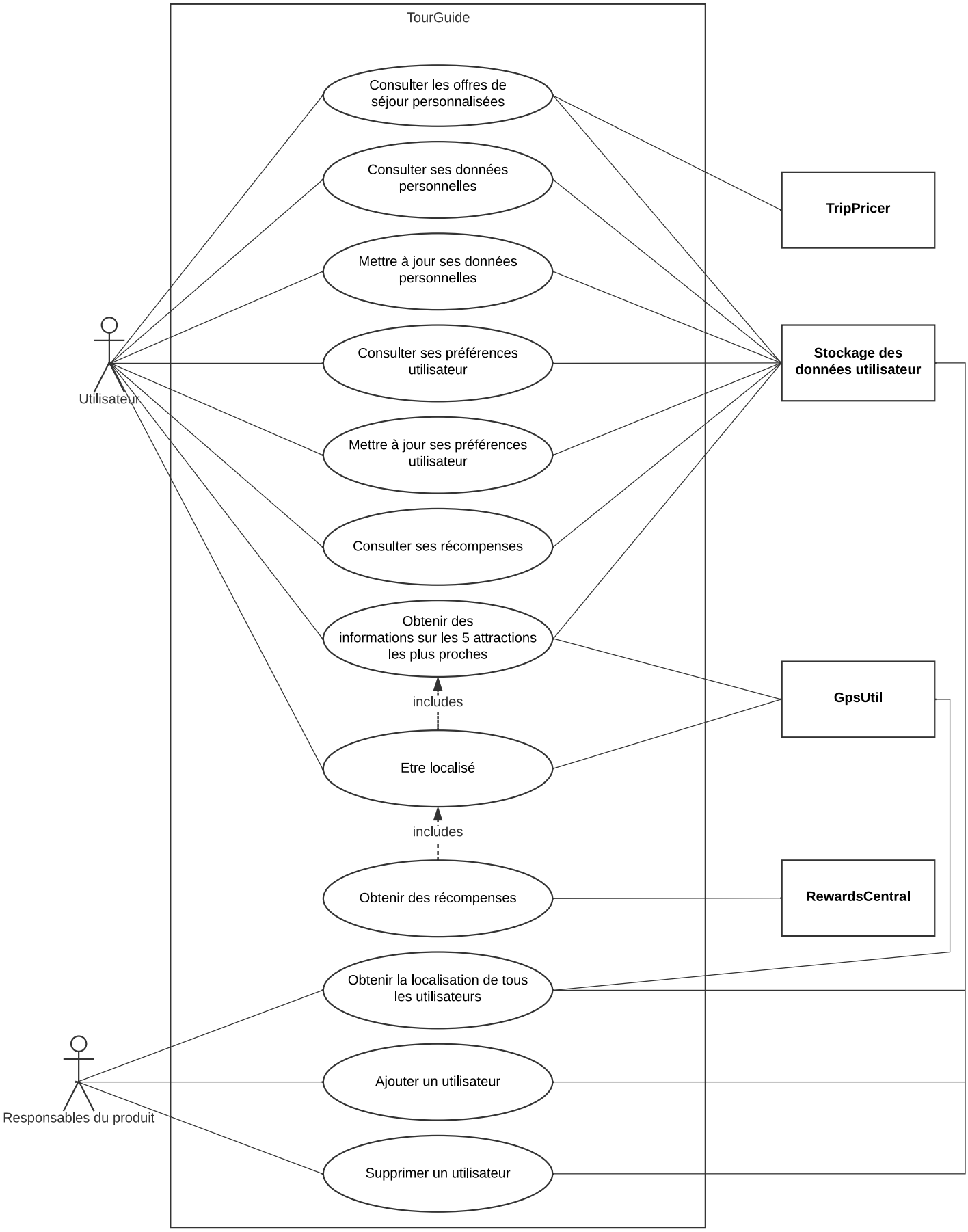
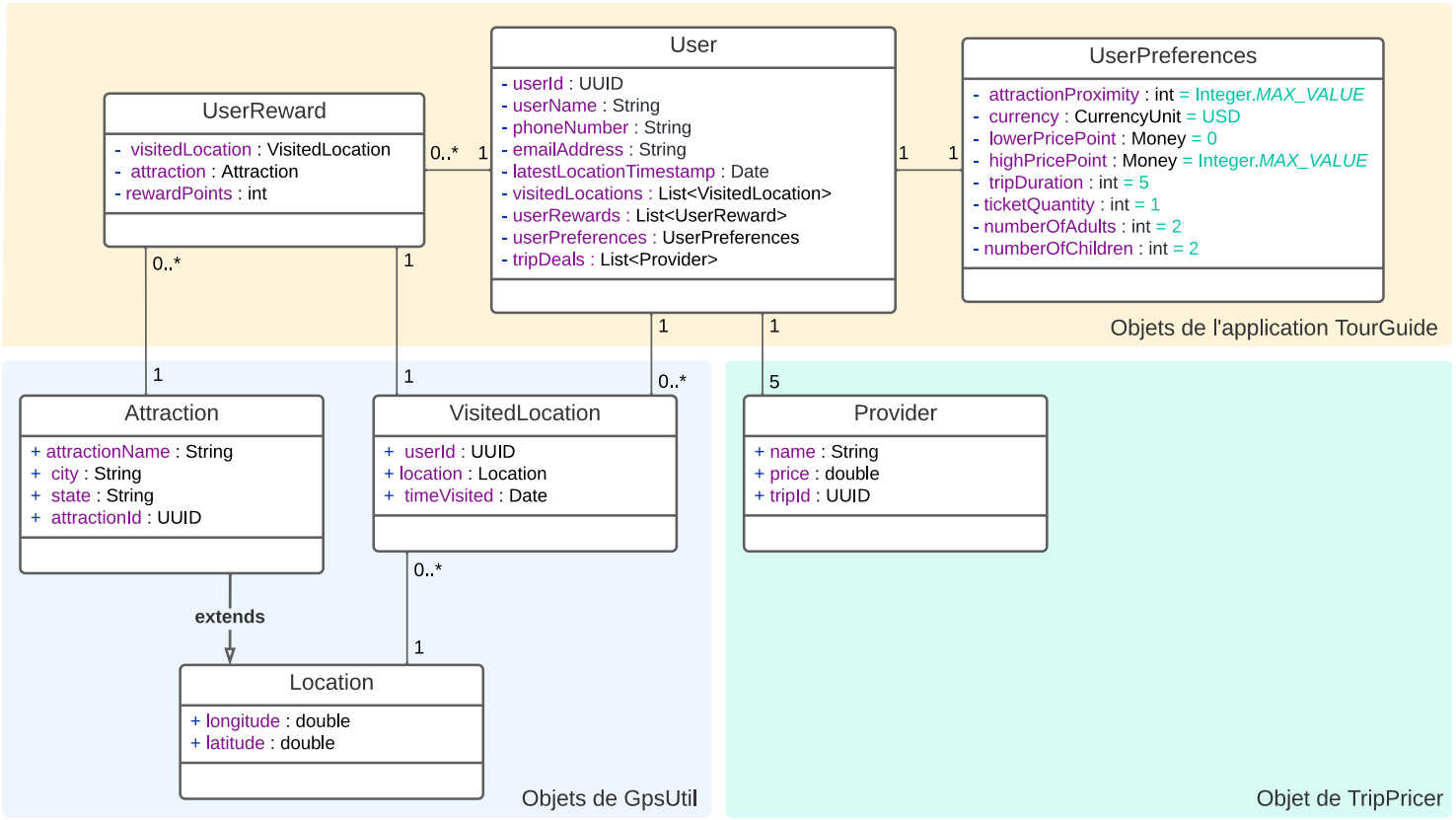
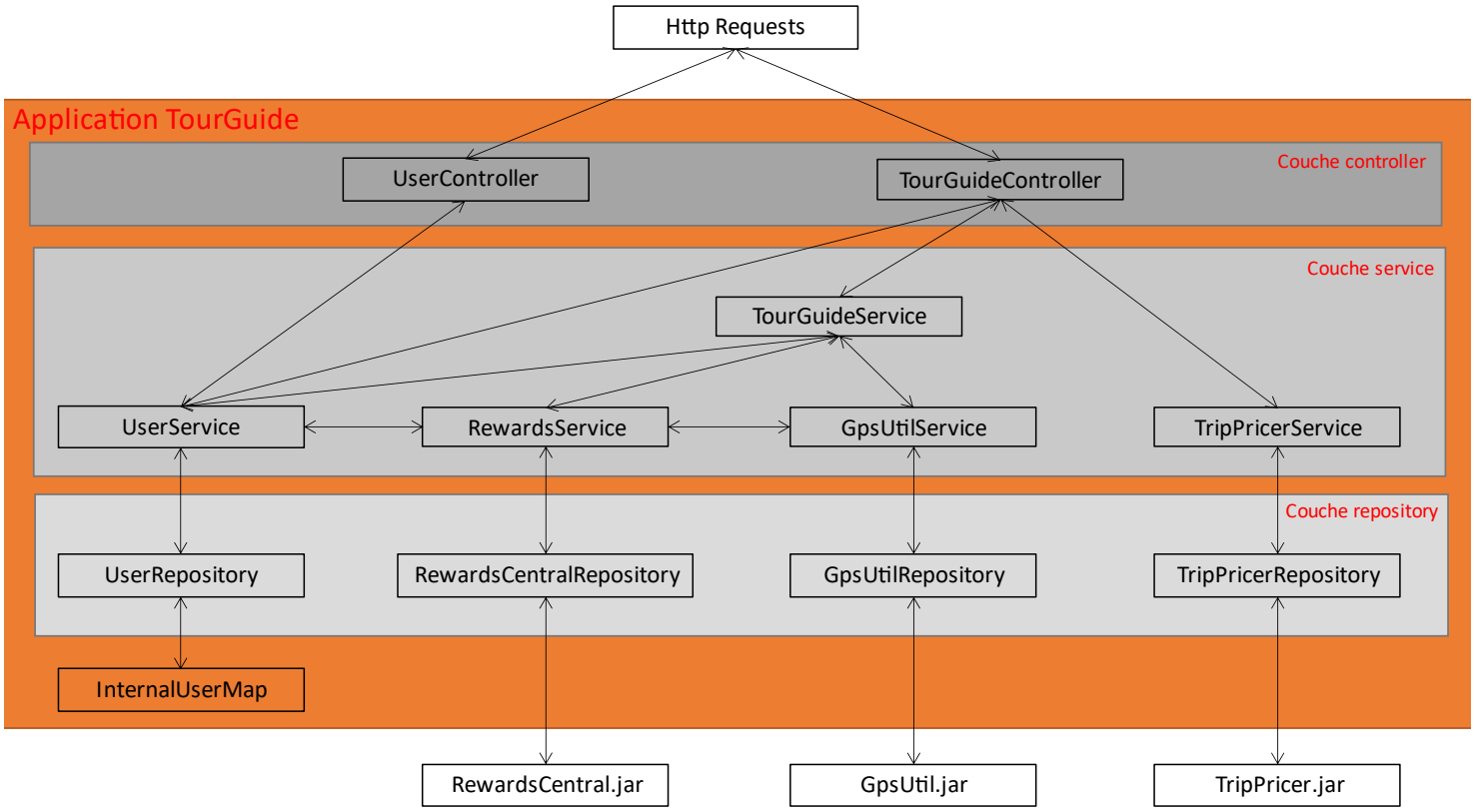


Diagramme UML des classes



Présentation de l'architecture de TourGuide



3.2 Glossaire

Afin d'éviter les confusions et de favoriser une bonne compréhension du domaine par tous les participants au projet, voici quelques définitions du langage omniprésent spécifique à l'application TourGuide.

Location

C'est une localisation précise, définie par ses coordonnées géographique (latitude et longitude).

Attraction

Cet objet hérite de l'objet *Location*. C'est donc une localisation précise, définie par ses coordonnées géographiques.
Cette localisation est spécifique par le fait d'être un point d'intérêt particulier, un lieu à visiter par les utilisateurs. L'objet possède donc trois attributs supplémentaires : le nom du lieu, la ville et l'état dans lequel il se trouve.

VisitedLocation

C'est une localisation précise à laquelle un utilisateur a été localisé. Cet objet est défini par les coordonnées géographiques de la localisation (latitude et longitude) ainsi que l'identifiant de l'utilisateur qui y a été localisé et le moment où il y a été localisé. Chaque *VisitedLocation* appartient donc à un utilisateur et chaque utilisateur possède ainsi en attribut la liste de toutes ses *VisitedLocations*.

Provider

C'est un objet contenant les informations relatives à une offre de séjour personnalisée.
Un objet de type *Provider* est défini par un nom, un prix et un identifiant.
Un *Provider* appartient à un utilisateur et est calculé en fonction de ses *UserPreferences* et de ses *UserRewards*. Chaque utilisateur possède ainsi une liste de 5 *Providers* en attributs.
Les *Providers* sont recalculés à chaque nouvelle consultation.

UserPreferences

C'est un objet qui regroupe l'ensemble des préférences de l'utilisateur, telles que la monnaie utilisée, le prix maximum et le prix minimum, le nombre de jours, le nombre d'adultes et le nombre d'enfants pour un séjour. Cet objet est utilisé par le service TripPricer pour calculer les *Providers* et faire des propositions de séjour personnalisées à l'utilisateur.

UserReward

Il s'agit d'un objet contenant les points attribués à un utilisateurs lorsqu'il se trouve proche d'une attraction, c'est-à-dire lorsqu'il se trouve dans le rayon défini par le *ProximityBuffer*.
Un *UserReward* contient l'*Attraction* à proximité de laquelle l'utilisateur s'est rendu, la *VisitedLocation* où il se trouvait et le nombre de points attribués. Cet objet est utilisé pour calculer les *Providers* proposés à un utilisateur.
Chaque utilisateur possède, en attribut, la liste de ses *UserRewards*.

ProximityBuffer

Il s'agit de la distance maximale entre un utilisateur et une *Attraction* permettant d'obtenir des *UserRewards*.

Tracker

Le Tracker est un objet rattaché à TourGuideService, qui démarre dès que le service est créé. Il détermine la position de chaque utilisateur inscrit sur l'application permettant ainsi de mettre à jour la liste de *VisitedLocations* et les éventuels *UserRewards* de chaque utilisateur. Le Tracker redémarre automatiquement 5 minutes après avoir terminé de repérer tous les utilisateurs.

3.3 Spécifications techniques

Langages et Frameworks

- *Java 1.8*
- *Gradle 4.8.1*
- *SpringBoot 2.1.6*
- *Java Concurrency Framework :*
    - *ForkJoinPool*
    - *RecursiveTask*
    - *BlockingQueue*

Tests :

- *JUnit 4.1.0*
- *Mockito*
- *JaCoCo*

Pipeline d'intégration continue :

- *GitLab*

3.4 Solutions alternatives

Les services GpsUtil et RewardsCentral pourraient être modifiés afin d'être plus rapides pour améliorer les performances de l'application.
Cependant il s'agit de services externes à l'application. Il est donc plus judicieux de modifier l'application elle-même pour la rendre plus efficace quelle que soit les services auxquels elle fait appel.

3.5 Calendrier prévisionnel et exigences

Exigences au niveau des performances

- **La localisation des utilisateurs doit être plus rapide, 100 000 utilisateurs doivent pouvoir être localisés en moins de 15 minutes**
  La méthode utilisée par le Tracker a été modifiée, «trackAllUsers» prend maintenant en paramètre la liste des utilisateurs, elle utilise une «RecursiveTask» avec un «ForkJoinPool» pour obtenir la localisation de tous les utilisateurs et une «LinkedBlockingQueue» pour mettre ensuite à jour les positions des utilisateurs et calculer les éventuelles récompenses. Ainsi la méthode est plus rapide, 100 000 utilisateurs sont localisés en environ 11 minutes. La position des utilisateurs est mise à jour régulièrement et la dernière position enregistrée peut être utilisée lorsqu'un utilisateur a besoin d'être localisé.

- **Le calcul des récompenses doit être plus rapide, 100 000 utilisateurs doivent pouvoir obtenir des récompenses en moins de 20 minutes**
  La méthode « calculateRewards » a été modifiée : elle utilise une «RecursiveTask» avec un «ForkJoinPool» pour parcourir la liste des attractions et des userRewards de l'utilisateur et renvoyer uniquement les «RewardElements» pour lesquels il faut ajouter une récompense à l'utilisateur. Ces derniers sont ensuite soumis à un «ExecutorService» pour calculer les points de récompense et les ajouter à l'utilisateur.
  De plus, la liste des attractions est mise en attribut de la classe «RewardsService» lors de son instanciation pour un accès plus rapide. Elle peut être mise à jour en cas de modifications avec le setter.
  Les récompenses pour 100 000 utilisateurs sont maintenant calculées en 40 secondes.
  Cette implémentation aura l'avantage de rester rapide même si le nombre d'attractions ou d'userRewards augmente.

Exigences au niveau des fonctionnalités

- **Des attractions doivent toujours être proposées à l'utilisateur, quelle que soit la distance**
  La fonction «getNearbyAttractions» a été modifiée, elle fait appel à la fonction «searchFiveClosestAttractions» qui retourne les 5 attractions les plus proches de l'utilisateur, quelle que soit la distance, avec leur nom et leurs coordonnées géographiques.

- ***Les propositions de séjour doivent toujours tenir compte des préférences de l'utilisateur***
  Un «userController» a été ajouté, il propose des méthodes permettant de visualiser et mettre à jour les données et préférence de l'utilisateur, afin de pouvoir en tenir compte notamment lors du calcul des «TripDeals».

- **Une nouvelle fonctionnalité doit permettre d'obtenir la localisation de tous les utilisateurs connectés**
  Une fonction «getAllCurrentLocations» a été ajoutée, elle permet d'obtenir la position de tous les utilisateurs à partir de la dernière position enregistrée (sauf si l'utilisateur n'a jamais été localisé, dans ce cas la position actuelle est calculée).
  Elle affiche la position de chaque utilisateur avec son identifiant.

Exigences au niveau du code

- **Le code doit respecter les normes de programmation TourGuide, qui reposent sur l'article *The Elements of Modern Java Style* de DZone**
  Le code respecte la convention de nommage CamelCase, les instructions de bloc sont entre accolades, les accolades sont correctement placées, les exceptions sont traitées.

Exigences au niveau de la maintenance

- **Le code doit être maintenable, il doit être réorganisé**
  Le code a été réorganisé pour correspondre aux principes SOLID et au modèle MVC.
- **Les tests doivent être corrigés et complétés**
  L'application est couverte par des tests unitaires et d'intégration, la couverture de code est mesurée par JaCoCo.
- **Il faut assurer un suivi plus régulier pour éviter les problèmes futurs**
  Un pipeline est installé sur GitLab afin d'avoir une intégration continue et d'effectuer les tests avant chaque commit. De cette façon le suivi est régulier et les éventuels dysfonctionnements sont immédiatement repérés.

Calendrier prévisionnel

- **L'application est prête à être mise en production**