Test-Driven Developpment

and

Continuous Integration

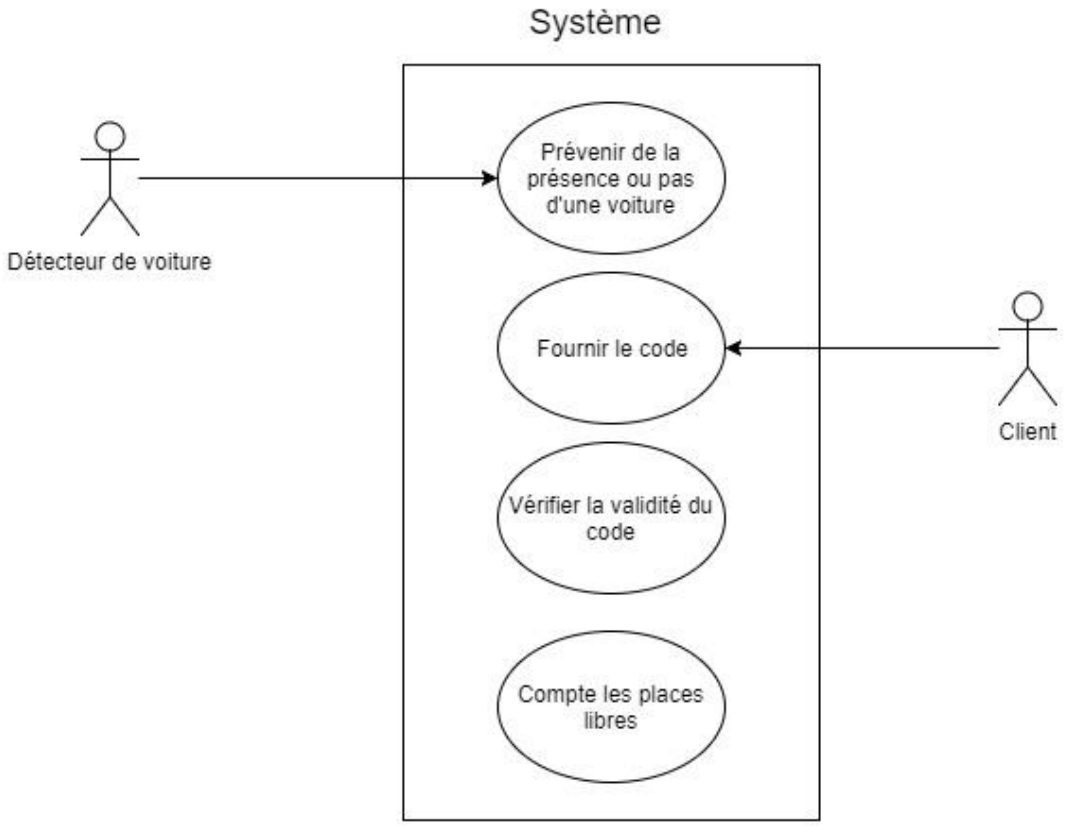
Equipe 7 – Gestion de Parking

Description du projet

Programme permettant de gérer un parking :

- Déterminer le nombre places (Libres, Occupées)
- Indiquer les places libres via une lumière verte au-dessus de celles qui sont libres
- Confirme un code donné par le client qui l'a obtenu dans le magasin

Diagramme de Cas d'utilisation



Cas d'utilisation – Canvas 1

Cas :

Compter nombre de places libres

Objectifs :

Indiquer nombre de places libres

Limitation du cas :

Les informations renvoyées par les capteurs ne sont pas correctes

Hypothèse :

Le système reçoit les informations des capteurs

Flot :

- Lorsqu'une voiture se gare, le système décrémente le nombre de places libres et retire cette place de la liste des places libres

Cas d'utilisation – Canvas 2

Cas :

Vérifier la validité du code

Objectifs :

Eviter les abus de stationnement

Limitation du cas :

Détecteur de code est défectueux ou il y a une erreur dans le code

Hypothèse :

Le code est fourni au système

Flot :

- Le client fait ses achats
- Le client reçoit un ticket avec un code qui est valable 1 heure
- Le code est vérifié par le système
- Le client sort du parking grâce au code

Flot alternatif :

- Le client fait ses achats
- Le client reçoit un ticket avec un code qui est valable 1 heure
- Le code est vérifié par le système
- Le système refuse que le client sorte car le code n'est pas valide

Cas d'utilisation – Canvas 3

Cas :

Client fournit le code

Objectifs :

Analyser le code pour ensuite vérifier sa validité

Limitation du cas :

Le client n'a pas fait d'achat et donc n'a pas de code

Hypothèse :

Le client a fait des achats et a reçu un ticket avec un code unique généré à la caisse

Flot :

- Le client fait ses achats
- Lorsqu'il a fini ses achats, le client reçoit un ticket avec un code qui est valable 1 heure
- Le code est vérifié par le système
- Le client sort du parking grâce au code

Flot alternatif :

- Le client fait ses achats
- Lorsqu'il a fini ses achats, le client reçoit un ticket avec un code qui est valable 1 heure
- Le code est vérifié par le système
- Le système refuse que le client sorte car le code n'est pas valide

Cas d'utilisation – Canvas 4

Cas :

Détecteur de voiture prévient de la présence ou pas d'une voiture

Objectifs :

Permettre au système de connaitre les places libres

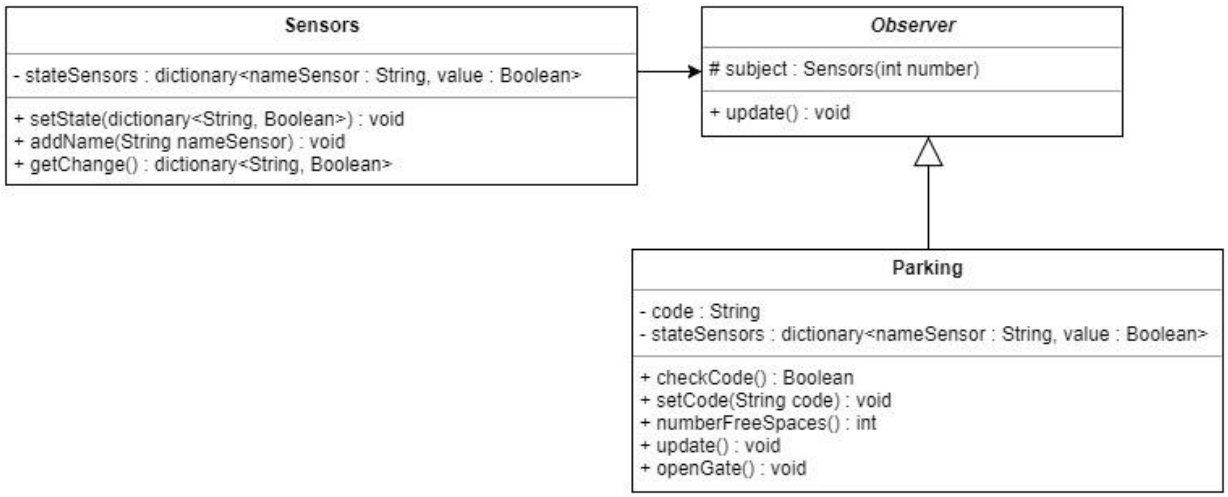
Hypothèse :

Les détecteurs de voiture fournissent des informations correctes au système

Flot :

- Le client retire sa voiture de la place de parking
- Le capteur détecte qu'il n'y a plus de voiture
- Le capteur renvoie l'information au système

Diagramme de Classes



Convention de codage/nommage

Nom Entité

Nom général pour les différents identifiants

- On ne nomme pas une entité comme suit : name_, mName, s_name and kName.

Nom classe

- UpperCamelcase.

  Exemple : XmlHttpRequest.

Nom méthode

- lowerCamelcase.

  Exemple : supportsIpv6OnIos.

Constante

- CONSTANT_CASE = ..., tout est écrit en majuscule.

Variable

- lowerCamelcase.

  Exemple : supportsIpv6OnIos.

Variable/objet nouvelle classe

- A name in the form used for classes followed by the capital letter.

Format Ligne

Indentation : 4 espaces.

Nombre de caractère par ligne maximum : 100.

Métrique

Ligne de Code SLOC

- Ligne physiques : Présentes dans le fichier
- Lignes logiques : Effectivement exécutées

Densité des commentaires par rapport aux lignes de code DC

- DC = CLOC/SLOC

  CLOC: Comment Line Of Code – Nombre de lignes de commentaires

  SLOC: Source Line Of Code – Nombre de lignes de code

Couverture de code

- Proportion de code couverte par des tests

Duplication de code

Couplage

- Couplage efferent Ce : Nombre références vers classe mesurée
- Couplage afferent Ca : Nombre types que la classe connait

Instabilité

- Résistance d'un module au changement
  Ce/Ce+Ca

Lack of Cohesion of Methods

- Manque de cohésion des méthodes

  LCOM = 1 - $\frac{\sum_F MF}{MxF}$ M : Nombre méthodes F : Nombre champs d'instance MF : Nombre de méthodes appelant un champ donné

Nombre d'éléments

- Nombre d'éléments dans une classe
  Paramètres <=5 Variables <=8 Surcharges <=6

Paradigme

Manière de représenter le monde, de voir les choses.

Programmation Orientée Objet

Ensemble de classes et d'objets représentant des objets réels ou des concepts et interagissant ensemble. Chaque objet possède des comportements.

Design Pattern : Observer

Cas d'utilisation

Un détecteur de voiture est placé au-dessus de chaque place de parking, celle-ci est verte lorsque la place est vacante.

Explication du choix

Le détecteur est l'observateur car il change d'état en fonction de l'occupation de la place de parking.

Le pattern Observer est donc un choix tout à fait justifié.

Diagramme de classe du pattern