**Documentation pour le Système de Location de Vélo**

**Titre**: Documentation du Système de Location de Vélo  
**Auteur**: Bachuki Labadze  
**Cours**: Université d’Evry – Licence informatique L3 Modélisation Objet – UML

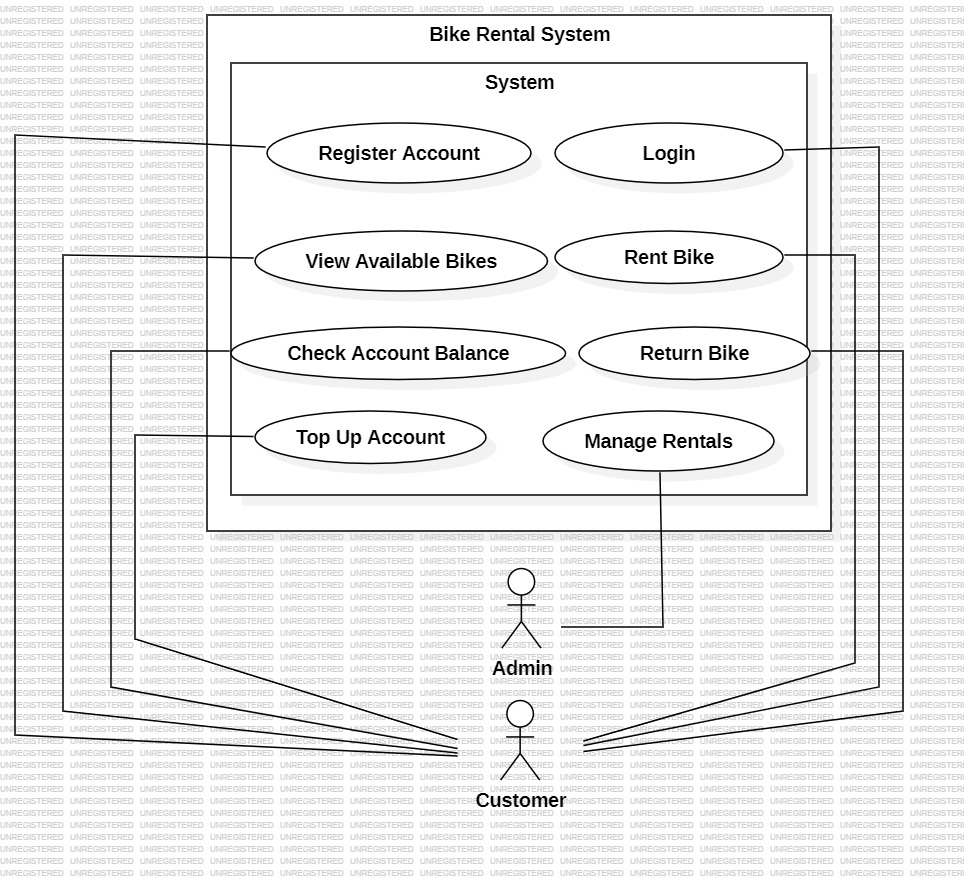
**Table des Matières**

1. Introduction
2. Diagramme de Cas d'Utilisation
3. Descriptions des Cas d'Utilisation
4. Diagramme de Classes
5. Diagrammes de Séquence
6. Conclusion

**1. Introduction**

**Objectif**: Ce document fournit un aperçu du système de location de vélo, détaillant ses cas d'utilisation, sa structure de classes et ses interactions.  
**Portée**: Le système permet aux clients de louer des vélos, de gérer leurs comptes et fournit des fonctions administratives.

**2. Diagramme de Cas d'Utilisation**



**Explication:** Ce diagramme de cas d'utilisation illustre les différentes interactions entre les acteurs (Client et Admin) et les fonctionnalités du système de location de vélo. Il montre comment les clients peuvent créer un compte, louer et retourner un vélo, ainsi que les actions administratives nécessaires pour gérer le système. Chaque cas d'utilisation représente un aspect essentiel du fonctionnement du système, facilitant ainsi la compréhension de ses fonctionnalités et de ses exigences.

**3. Descriptions des Cas d'Utilisation**

**Cas d'Utilisation 1: Créer un Compte**

* **Acteur**: Client
* **Préconditions**: Le client doit avoir des informations valides pour s'inscrire.
* **Postconditions**: Le compte du client est créé.
* **Flux Principal**:
  1. Le client saisit les détails d'inscription.
  2. Le système valide les informations.
  3. Le compte est créé.
* **Flux Alternatifs**:
  1. Si les informations sont invalides, le système invite à corriger.

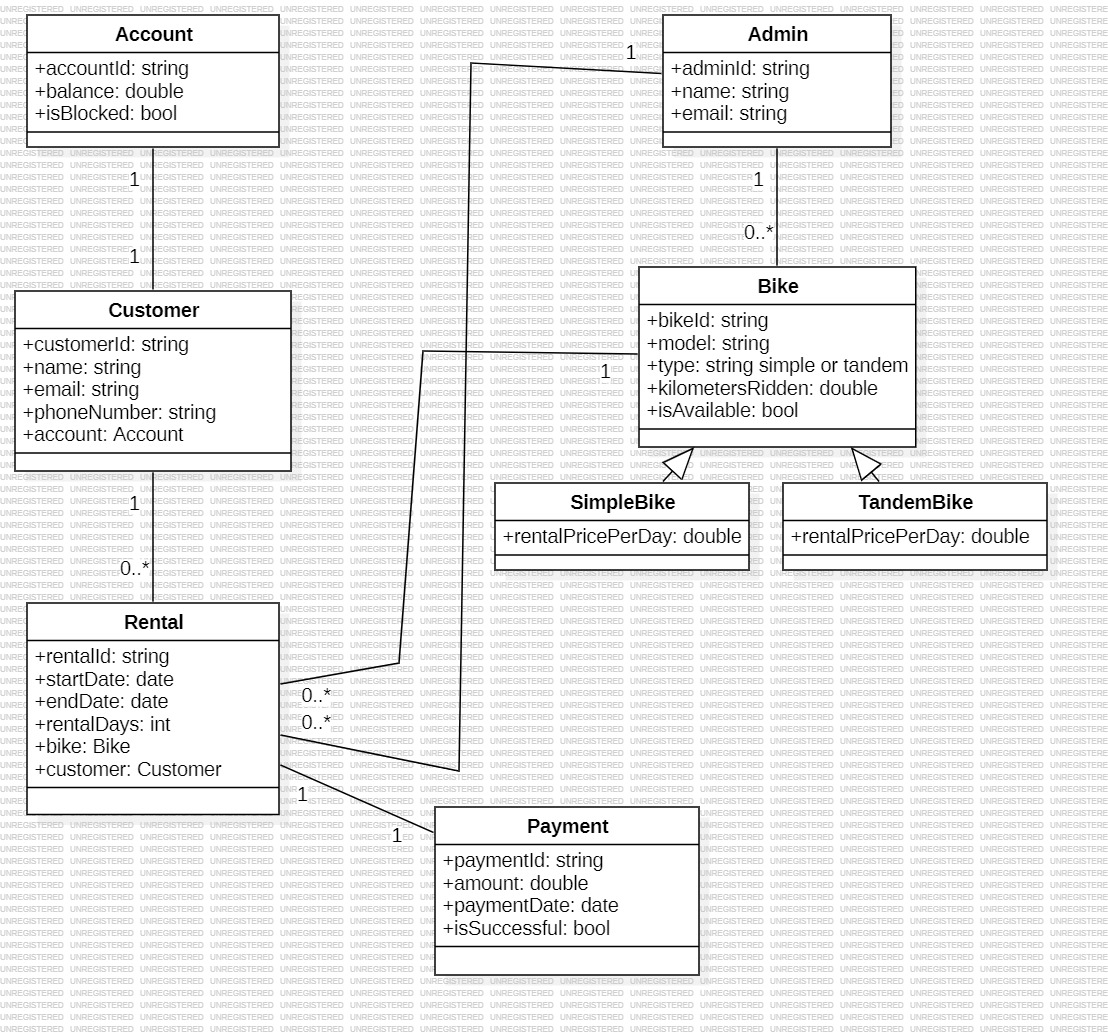
**Cas d'Utilisation 2: Louer un Vélo**

* **Acteur**: Client
* **Préconditions**: Le client doit être connecté et avoir un solde de compte suffisant.
* **Postconditions**: Le vélo est marqué comme loué.
* **Flux Principal**:
  1. Le client sélectionne un vélo.
  2. Le client spécifie la durée de location.
  3. Le système vérifie le solde et débite le compte.
* **Flux Alternatifs**:
  1. Si le solde est insuffisant, inviter le client à créditer son compte.

**Cas d'Utilisation 3: Retourner un Vélo**

* **Acteur**: Client
* **Préconditions**: Le vélo doit avoir été loué par le client.
* **Postconditions**: Le vélo est marqué comme disponible.
* **Flux Principal**:
  1. Le client retourne le vélo.
  2. Le système met à jour le statut du vélo.
  3. Des frais supplémentaires sont appliqués si nécessaire.
* **Flux Alternatifs**:
  1. Si le vélo est en retard, appliquer des frais de retard.

**4. Diagramme de Classes**



**Classes et Attributs**

1. **Customer (Client)**
   * **Attributs :**
     + **customerId: String — Identifiant unique du client.**
     + **name: String — Nom du client.**
     + **email: String — Adresse e-mail du client.**
     + **account: Account — Référence à l'objet compte du client.**
   * **Relations :**
     + **Association avec Account : Un client a exactement un compte (1 à 1).**
     + **Association avec Rental : Un client peut avoir zéro ou plusieurs locations (1 à 0..\*).**
2. **Account (Compte)**
   * **Attributs :**
     + **accountId: String — Identifiant unique du compte.**
     + **balance: Double — Solde du compte.**
     + **isBlocked: Boolean — Indicateur si le compte est bloqué ou non.**
   * **Relations :**
     + **Association avec Customer : Un compte est associé à un seul client (1 à 1).**
3. **Rental (Location)**
   * **Attributs :**
     + **rentalId: String — Identifiant unique de la location.**
     + **startDate: Date — Date de début de la location.**
     + **endDate: Date — Date de fin de la location.**
     + **bike: Bike — Référence à l'objet vélo loué.**
     + **customer: Customer — Référence au client qui effectue la location.**
   * **Relations :**
     + **Association avec Bike : Une location est associée à un seul vélo (1 à 0..\*).**
     + **Association avec Payment : Une location a un paiement associé (1 à 1).**
     + **Association avec Customer : Une location est effectuée par un client (1 à 1).**
4. **Bike (Vélo)**
   * **Attributs :**
     + **bikeId: String — Identifiant unique du vélo.**
     + **type: String — Type de vélo (simple ou tandem).**
     + **mileage: Double — Kilométrage parcouru par le vélo.**
   * **Relations :**
     + **Association avec Rental : Un vélo peut être loué dans plusieurs locations (0..\* à 1).**
     + **Association avec Maintenance : Un vélo peut avoir zéro ou plusieurs enregistrements de maintenance (1 à 0..\*).**
5. **Maintenance (Entretien)**
   * **Attributs :**
     + **maintenanceId: String — Identifiant unique de l'entretien.**
     + **date: Date — Date de l'entretien.**
     + **description: String — Description des travaux effectués.**
   * **Relations :**
     + **Association avec Bike : Un entretien est associé à un seul vélo (1 à 0..\*).**
6. **Admin (Administrateur)**
   * **Attributs :**
     + **adminId: String — Identifiant unique de l'administrateur.**
     + **name: String — Nom de l'administrateur.**
   * **Relations :**
     + **Association avec Bike : Un administrateur peut gérer zéro ou plusieurs vélos (1 à 0..\*).**
     + **Association avec Rental : Un administrateur peut superviser zéro ou plusieurs locations (1 à 0..\*).**
7. **Payment (Paiement)**
   * **Attributs :**
     + **paymentId: String — Identifiant unique du paiement.**
     + **amount: Double — Montant du paiement.**
     + **paymentDate: Date — Date du paiement.**
   * **Relations :**
     + **Association avec Rental : Un paiement est associé à une seule location (1 à 1).**

**Résumé des Relations**

* **Customer ↔ Account : 1 à 1 (Un client a un compte.)**
* **Customer ↔ Rental : 1 à 0..\* (Un client peut avoir plusieurs locations.)**
* **Rental ↔ Bike : 0..\* à 1 (Une location est associée à un vélo.)**
* **Bike ↔ Maintenance : 1 à 0..\* (Un vélo peut avoir plusieurs entretiens.)**
* **Admin ↔ Bike : 1 à 0..\* (Un administrateur peut gérer plusieurs vélos.)**
* **Admin ↔ Rental : 1 à 0..\* (Un administrateur peut superviser plusieurs locations.)**
* **Rental ↔ Payment : 1 à 1 (Chaque location a un paiement associé.)**

**5. Diagrammes de Séquence**

**Diagramme de Séquence pour "Louer un Vélo"**

