

# Assistant évènementiel de sorties à Paris : Cahier d'analyse

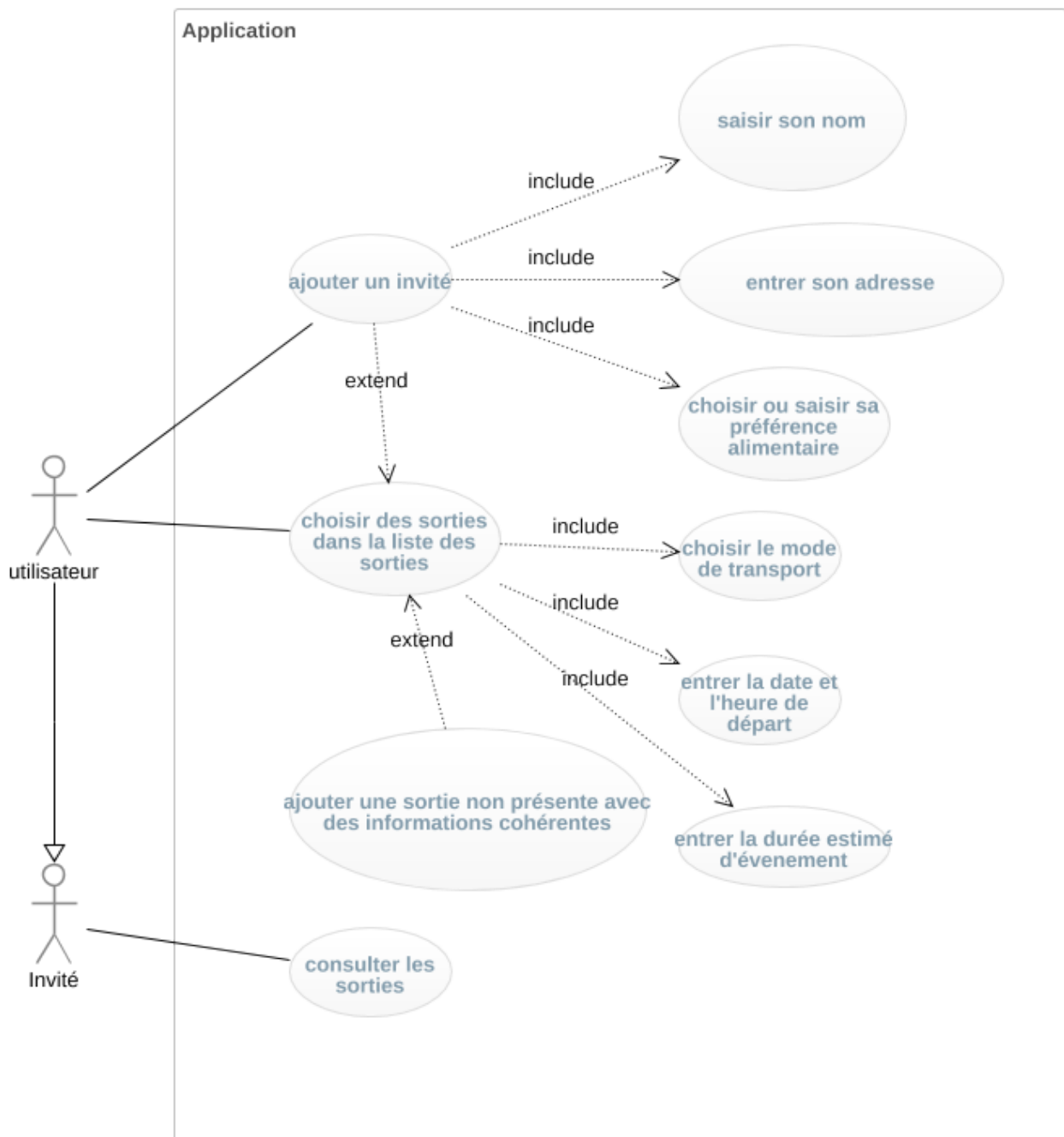
Binello Arthur, Mo Rui, Gan Boyuan

## Table des matières :

<b>1. Diagramme de cas d'utilisation</b>	<b>3</b>
1. Diagramme	3
2. Explications	4
<b>2. Diagrammes de séquence et scénarios</b>	<b>6</b>
1. Ajouter un invité	6
a. Diagramme	6
b. Scénario	7
2. Choisir des sorties dans la liste des sorties	8
a. Diagramme	8
b. Scénario	9
3. Ajouter une sortie non présente avec informations cohérentes	11
a. Diagramme	11
b. Scénario	12
4. Consulter les sorties	14
a. Diagramme	14
b. Scénario	15
<b>3. Diagramme de classe</b>	<b>16</b>
1. Diagramme	16
2. Explications	17

# 1. Diagramme de cas d'utilisation

## 1. Diagramme



## 2. Explications

Dans notre diagramme de cas d'utilisation, on en déduit que notre système sur laquelle les potentiels différents acteurs agissent est l'application en elle-même. Il existe un autre acteur secondaire nommé invité qui peut consulter les résultats des calculs. L'acteur principal hérite des fonctions de l'acteur principal, l'utilisateur participant lui-même à la sortie. Nous avons aussi décidé de n'avoir que ces acteurs là dans le système car il ajoute lui-même les autres participants sur l'application. Un acteur système n'était pas désiré car toute utilisation faite par celui-ci est en réaction par rapport à une utilisation par l'acteur principal.

L'utilisateur peut réaliser trois fonctions principales : ajouter un invité, choisir des sorties dans la liste des sorties et consulter les sorties. Il y a aussi sept fonctions secondaires : ajouter une sortie non présente avec des informations cohérents, saisir son nom, entrer son adresse, choisir ou saisir sa préférence alimentaire, choisir le mode de transport, entrer la date et l'heure de départ et entrer la durée estimé d'événement.

Les fonctions se décrivent ainsi :

- Ajouter un invité : L'utilisateur peut à tout moment ajouter un nouvel invité à la sortie, mais pour faire cela, il doit utiliser les fonctions secondaires saisir son nom, saisir son adresse, et choisir ou saisir ses préférences alimentaires. Suite à cet ajout, un nouvel acteur invité est créé.
- Choisir des sorties dans la liste des sorties : L'utilisateur peut choisir les sorties qu'il désire effectuer parmi la liste proposée par l'application. Il peut quand il le désire ajouter un nouvel invité. De même, si une sortie ne lui plaît pas, il peut décider d'en créer une personnalisée tant qu'elle est cohérente. Mais pour réaliser cette action, il doit préciser de nouveaux détails à savoir choisir le mode de transport, entrer l'heure et la date de départ et entrer la durée estimé d'événement.
- Consulter les sorties : À tout moment, tous les utilisateurs peuvent consulter l'état de la sortie, que les sorties aient été choisies ou non.
- Ajouter une sortie non présente avec des informations cohérents :
- Saisir son nom : Un invité doit avoir son nom saisi par l'utilisateur pour participer à la sortie.
- Saisir son adresse : Un invité doit avoir son adresse saisi par l'utilisateur pour participer à la sortie. Elle sera choisi en entrant une adresse textuelle ou en la sélectionnant grâce à l'API Google Maps.
- Choisir ou saisir sa préférence alimentaire : Un invité doit avoir ses préférences alimentaires saisis par l'utilisateur pour participer à la sortie. Un liste de toutes

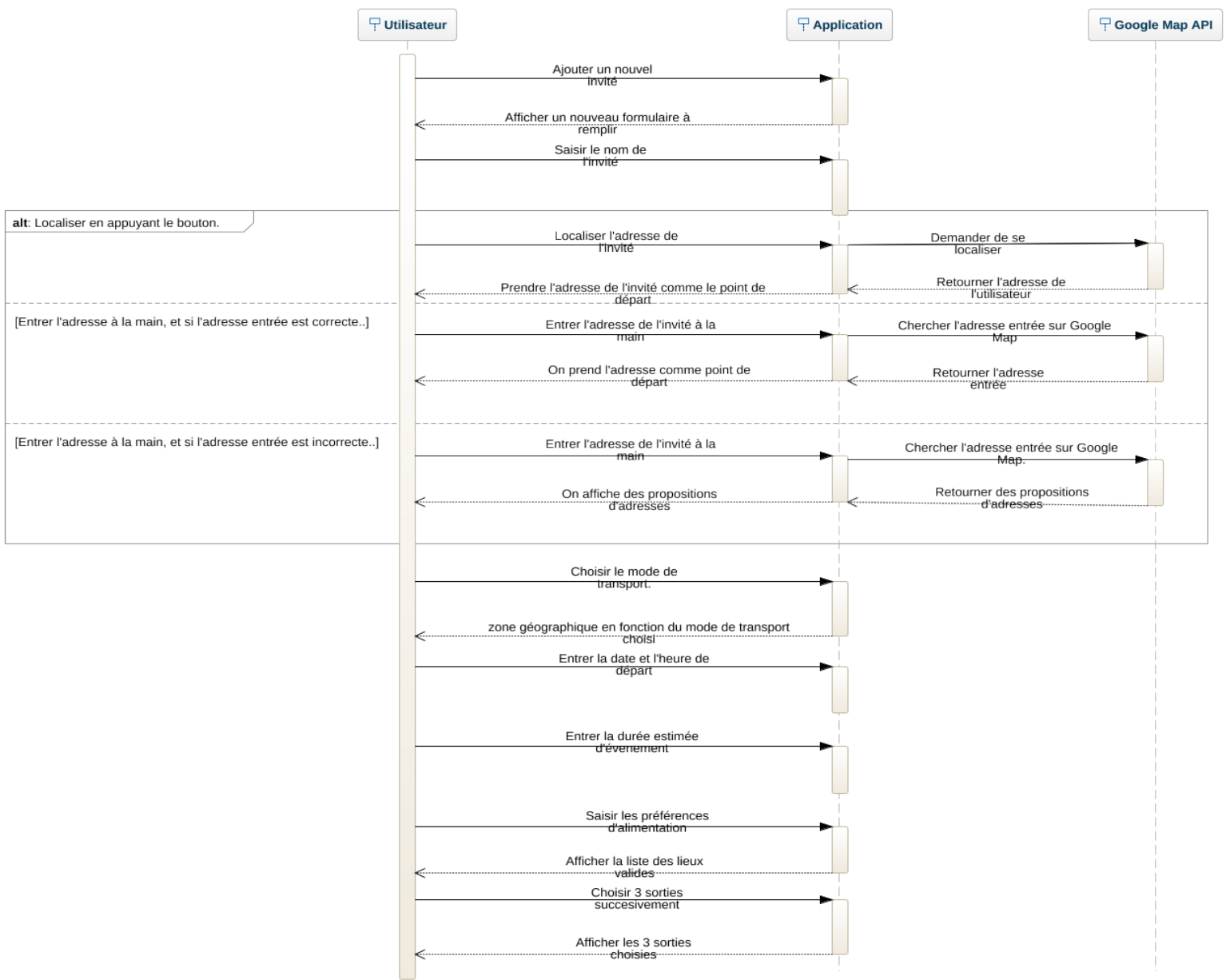
les préférences alimentaires entrées jusqu'à présent est proposée mais l'utilisateur peut en ajouter une nouvelle.

- Choisir le mode de transport : L'utilisateur doit choisir le mode de transport parmi la liste proposée par l'application (à pied, voiture, train, etc.). Ce choix correspond à la taille de la zone de sorties proposées en fonction du temps.
- Entrer la date et l'heure de départ : L'utilisateur doit entrer la date et l'heure de départ de la sortie.
- Entrer la durée estimée d'événement : L'utilisateur peut préciser la durée désirée de l'événement et le système calculera l'heure de fin en fonction.

## 2. Diagrammes de séquence et scénarios

### 1. Ajouter un invité

#### a. Diagramme



## b. Scénario

### **Identification :**

**Objectif :** Permet de rajouter un invité


**Acteur concerné :** l'utilisateur

### **Séquencement :**



Le cas d'utilisation se déclenche lorsque l'utilisateur désire ajouter une nouvel invité à la sortie.


**Préconditions :** L'utilisateur n'a pas encore cliqué le bouton ' Soumettre'.

### **Scénario nominal :**

1. L'utilisateur clique le bouton 'PLUS ' à côté de l'ancien formulaire.
2. Le système affiche un nouveau formulaire sur l'interface.
3. L'utilisateur remplit le formulaire.
4. L'utilisateur entre le nom du nouvel invité.
5. L'utilisateur entre l'adresse du nouvel invité.
6. L'utilisateur choisit le mode transport pour le nouvel invité.
7. L'utilisateur entre l'heure de départ du nouvel invité.
8. L'utilisateur entre l'heure sur place (la durée de la sortie).
9. L'utilisateur choisit la préférence du nouvel invité.

### **Scénario alternatif :**

1b, Si l'utilisateur veut ajouter plusieurs nouveaux invités, il doit cliquer le bouton 'PLUS ' et le maintenir enfoncé quelques secondes, puis il affiche une boîte de dialogue   qui demande à l'utilisateur d'entrer le nombre des nouveaux invités.

5b, Si le nouvel utilisateur a la même adresse que l'utilisateur, l'utilisateur peut aussi appuyer le bouton localiser  dans le formulaire pour localiser le nouvel invité, et le système retourne l'adresse de nouvel invité.

9b, S'il n'y a pas les préférences du nouvel invité, l'utilisateur les entre au système à la main.

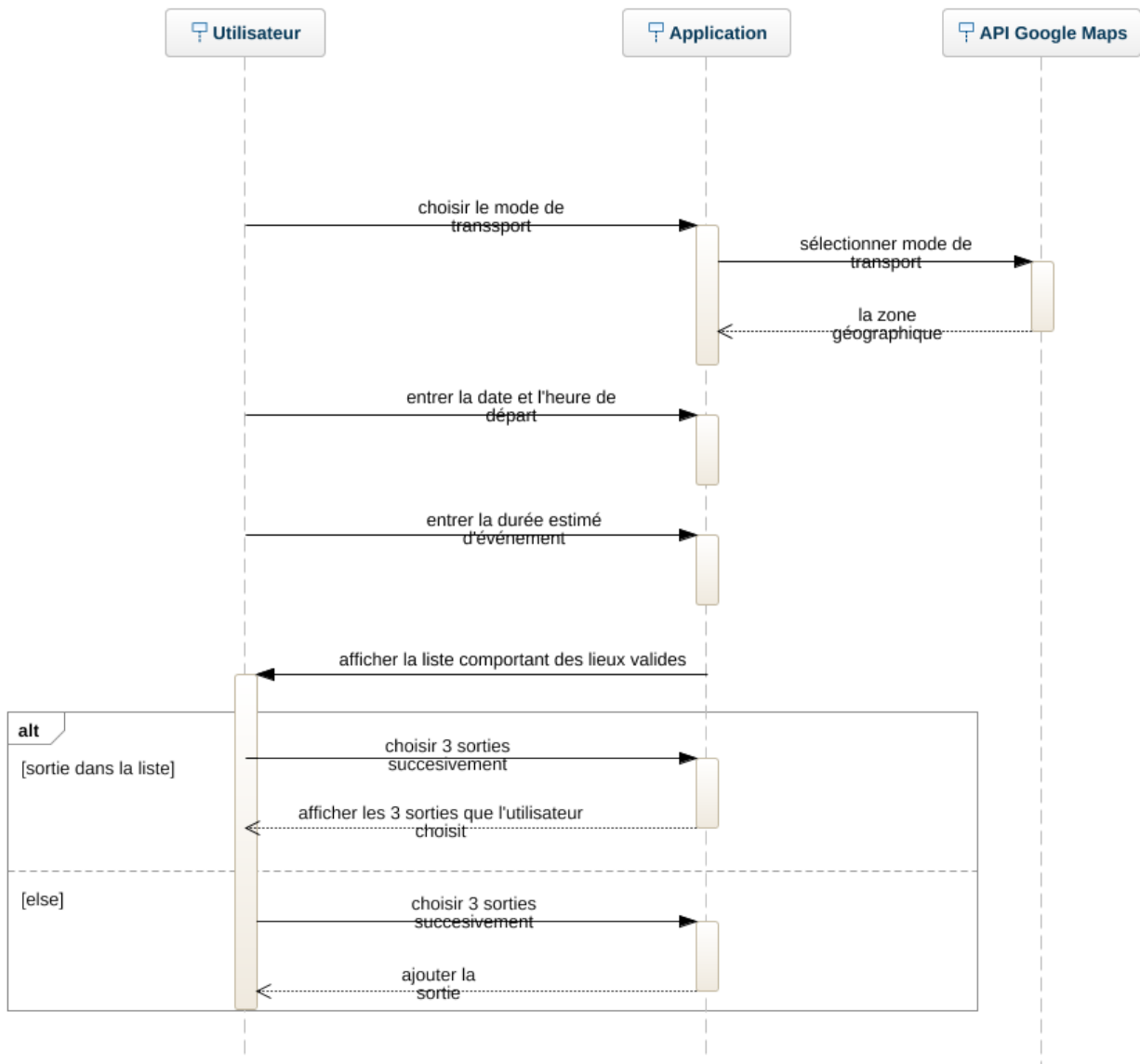
### **Scénario d'exception :**

3b, Si l'utilisateur ajoute une personne qui est déjà présente, le système alerte l'utilisateur et montre l'échec de l'ajout.

5c, Si l'adresse du nouvel invité entrée par l'utilisateur n'est pas correcte, le système proposera des adresses possibles.

## 2. Choisir des sorties dans la liste des sorties

### a. Diagramme





## b.Scénario

### **Identification**

**Objectif:** détaille les étapes permettant à un l'utilisateur de choisir des sorties dans la liste des sorties

**Acteur principal:** l'utilisateur

### **Séquencement**

Le cas d'utilisation commence lorsqu'un utilisateur veut sortir à Paris afin de fêter leurs retrouvailles et ont envie de passer une bonne soirée

**Précondition:** l'utilisateur entre son nom, son adresse et sa préférence alimentaire

### **Scénarios nominal**

1. Lorsque l'utilisateur veut choisir des sorties dans la liste, le système demande l'utilisateur entrer des informations suivantes:

- le mode de transport
- la date et l'heure de départ
- la durée estimé d'événement

2. Le système confirme le mode de transport que l'utilisateur choisit et propose une zone géographique selon les différents mode de transport

- Sortir en voiture : On prend 10 km comme le rayon du cercle.
- Sortir en transport public : On prend 7 km comme le rayon du cercle.
- Sortir en vélo : On prend 5 km comme le rayon du cercle
- Sortir à pied : On prend 2 km comme le rayon du cercle

3. Pour chaque type de lieu, le système a un temps d'activité moyen, si ce temps est supérieur à celui passé sur place, alors ce type de lieu sera éliminé dans le choix des lieux.

4. Pour chaque type de lieu qui a le temps d'activité moyen inférieur au temps passé sur place, le système traverse tous les lieux de ce type dans la zone géographique calculée, et le système calcule le temps total de visite pour les lieux de ces types. Le temps total de visite = l'heure de départ + le temps de trajet estimé (en s'assistant de l'API Google Map) + le temps d'activité moyen. Pour chaque lieu traversé, si le temps total de visite ne dépassera pas l'horaire de fermeture de ce lieu, le système ajoute ce lieu dans la liste à afficher. Après avoir traversé tous les lieux de la zone, le système affiche la liste comportant les lieux valides.

5. L'utilisateur prend trois lieux candidats parmi les lieux valides (de n'importe quel type), l'utilisateur décidera enfin quel sera leur premier lieu à visiter.

6. Pour le deuxième lieu à visiter, le système recalcule l'heure de départ au moment donné. L'heure de départ = l'heure d'arrivée au premier lieu + le temps total de visite du premier lieu. Puis le système recalcule le temps total de visite, ensuite le système répète le processus de filtrage des lieux valides de la même manière.

7. Pour le troisième lieu à visiter, le système répète le même processus que précédemment.

8. Après que l'utilisateur ait choisi toutes ses activités, une liste reprenant les activités choisies sera affichée.

### **Scénario alternatif**

5b. Quand l'utilisateur va choisir des sorties, si la sortie n'est pas dans la liste des sorties, alors l'utilisateur ajoute la sortie et saisit des informations cohérentes.

- Leurs adresses.
- Le lieu précis où ils désirent aller.
- L'activité qu'il désire faire.
- Le mode de transport. (L'utilisateur a 4 choix : en voiture, en transport public, à pied, en vélo.).
- L'heure de départ prévu.
- Le temps d'activité estimé.

Et nous décrivons des spécifications de cette fonctionnalité dans la page 12

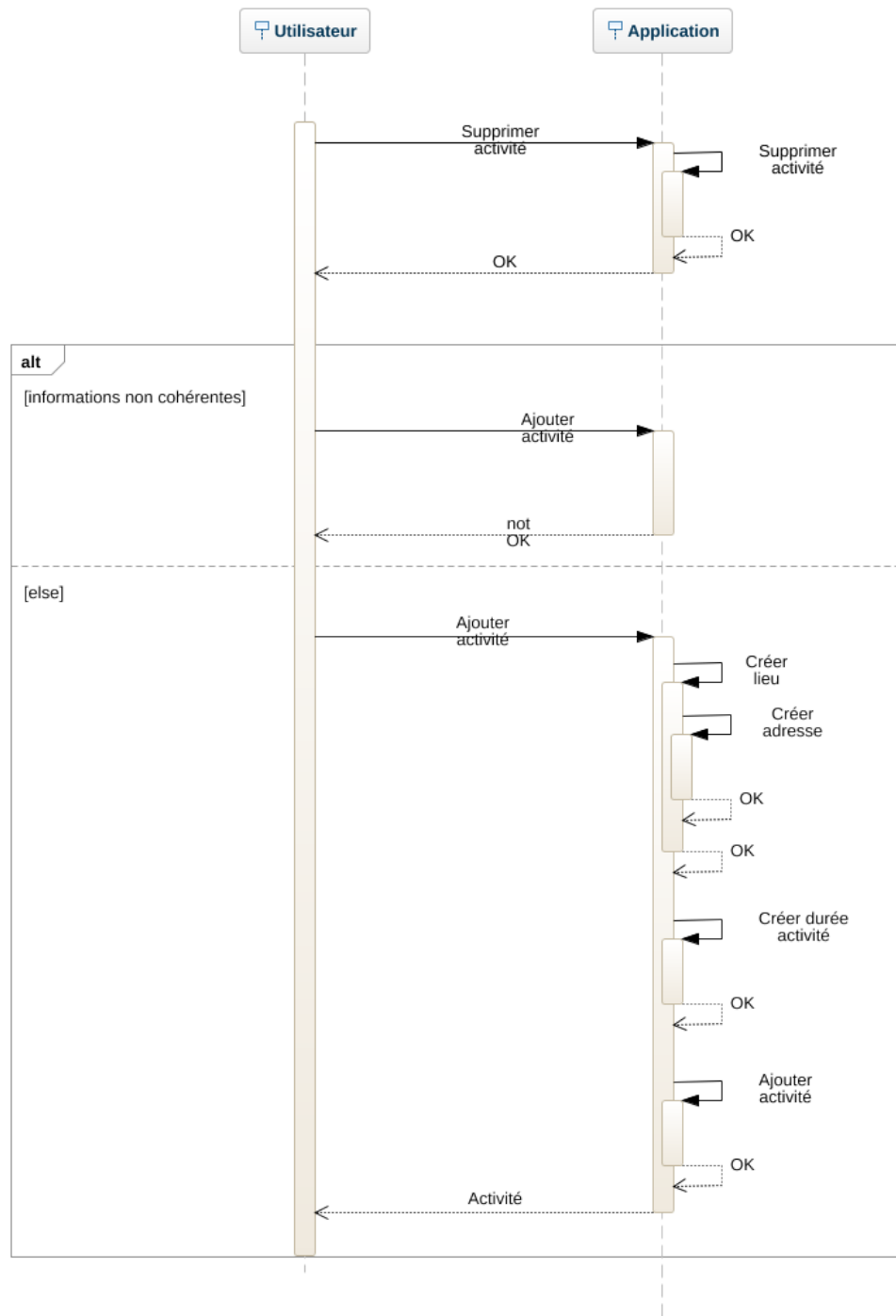
9. L'utilisateur peut ajouter des invités s'il y a plusieurs personnes qui sortent avec l'utilisateur et il faut saisir les informations suivantes:

- Le nom de l'invité
- L'adresse de l'invité
- La préférence alimentaire de l'invité

Pour bien préciser cette fonctionnalité, nous décrivons dans la page 7

### 3. Ajouter une sortie non présente avec informations cohérentes

#### a. Diagramme



## b. Scénario

### **Identification :**

**Objectif :** Permet de rajouter une sortie personnalisée parmi la liste des sorties proposées.

**Acteur concerné :** l'utilisateur

### **Séquencement :**

Le cas d'utilisation se déclenche lorsque l'utilisateur désire ajouter un événement personnalisé à la sortie planifiée pour en remplacer un autre.

**Préconditions :** L'utilisateur doit avoir une liste de sorties sélectionnée et doit sélectionner un événement particulier à remplacer.

### **Scénario nominal :**

1. L'utilisateur choisi parmi la liste des événements disponibles celui qu'il désire remplacer par son nouvel événement personnalisé.
2. Le système supprime l'événement choisi parmi la liste disponible et le fait savoir à l'utilisateur en lui proposant une interface pour ajouter son nouvel événement personnalisé.
3. L'utilisateur envoie les données de l'événement qu'il souhaite ajouter au système via le formulaire proposé. Pour cela, il devra fournir les informations :
  - Le nom de l'activité
  - L'adresse de l'activité
  - Le lieu de l'activité (peut être vide)
  - La durée de l'activité
4. Si les informations sont cohérentes, le système commencer à traiter cette demande en créant un lieu ainsi que son adresse associée.
5. Le système va ensuite créer la durée de l'événement dans la base de donnée pour déterminer si cet événement est possible au niveau du temps.
6. L'activité est ensuite ajoutée à la base de donnée par le système et celle-ci est ajoutée à la liste des événements affichée à l'utilisateur et est donc pris en compte dans la sortie.

### **Scénario alternatif :**

4b. Si les informations ne sont pas cohérents, le système le fait savoir à l'utilisateur et une nouvelle interface pour ajouter de nouvelles données est proposée à l'utilisateur.

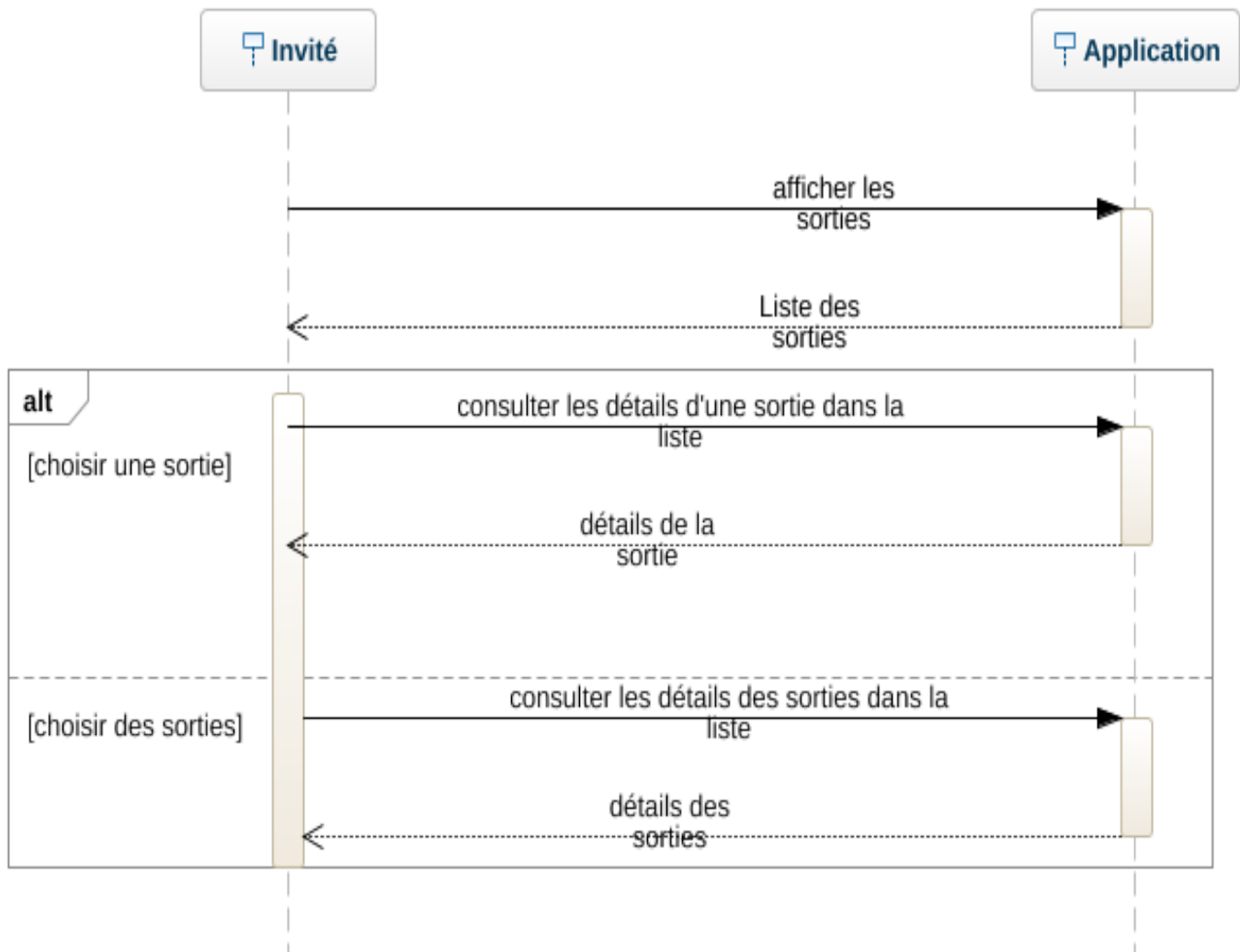
**Scénario d'exception :**

2b. Si la suppression de l'événement de base n'est pas réalisé correctement, le système alerte l'utilisateur de l'erreur et termine le scénario

6b. Si l'ajout de l'activité à la base de données ne se passe pas correctement, le système en alerte l'utilisateur, le scénario se termine et l'activité supprimée et ajoutée à nouveau.

## 4. Consulter les sorties

### a. Diagramme



## b. Scénario

### **Identification**

**Objectif:** détaille les étapes permettant à un l'utilisateur consulter une sortie ou plusieurs sorties dans la liste

**Acteur principal:** l'invité ou l'utilisateur (qui est aussi un invité)

### **Séquencement**

Le cas d'utilisation commence lorsqu'un invité veut consulter un événement dans la liste

**Précondition:** l'invité saisit toutes les informations cohérentes afin de demander le système lister des sorties après.

#### **Scénarios nominal:**

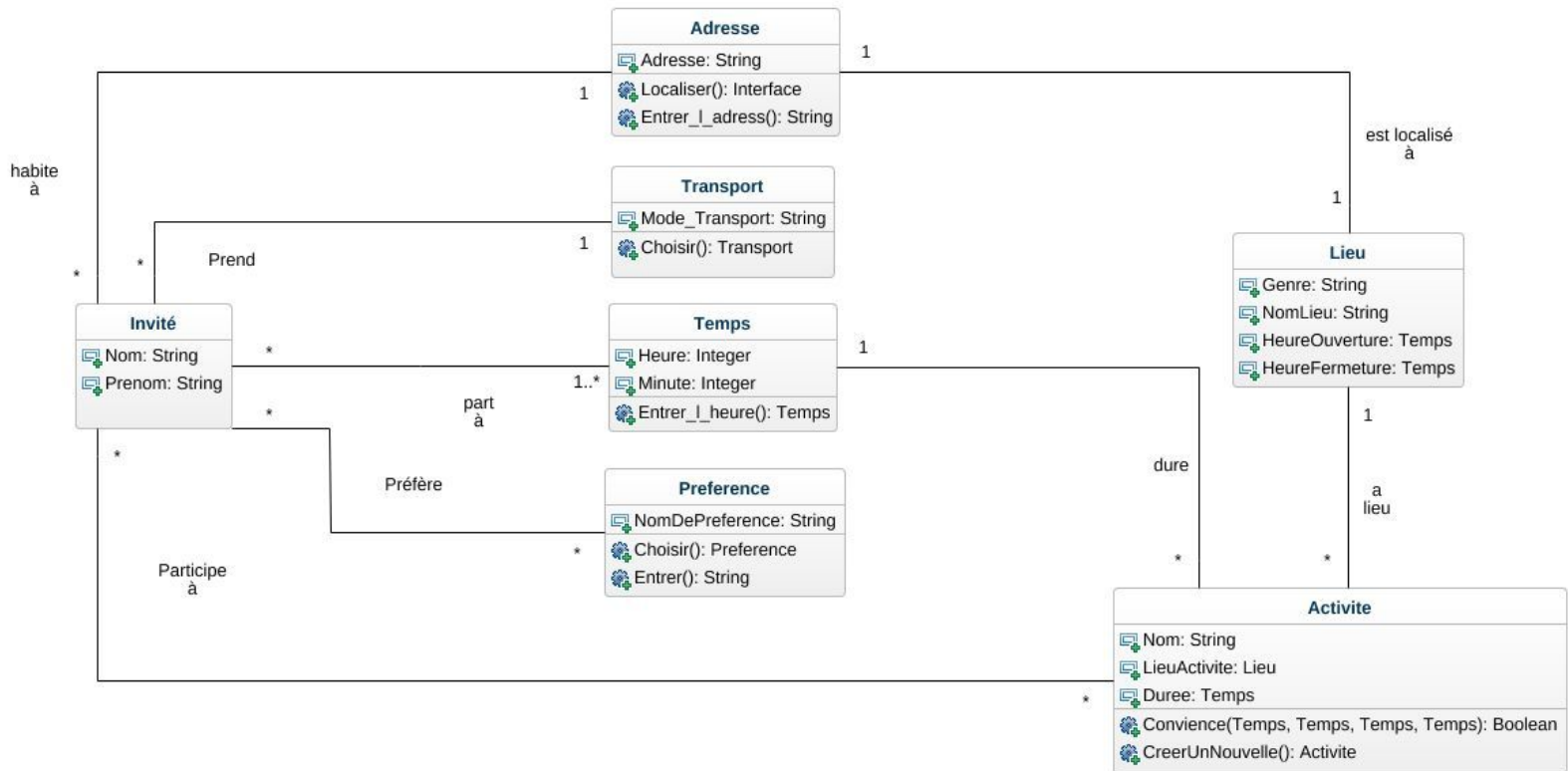
- 1 L'utilisateur demande le système lister toutes les sorties
- 2 Le système confirme la demande et affiche la liste des sorties
3. L'utilisateur choisit une sortie et demande le système afficher les détails de cette sortie
4. Le système confirme la demande et affiche les informations cohérentes de cette sortie

#### **Scénario alternatif:**

3b. Si l'utilisateur veut consulter plusieurs sorties, le système répète le même processus que précédemment.

### 3. Diagramme de classe

#### 1. Diagramme





## 2. Explications

Nous avons 7 classes en total.

### Descriptions des classes:

#### (1). Invite

Cette classe est pour créer l'objet de la personne qui veut sortir.

On a deux attributs dans cette classe:

(i), **Nom**: du type String.

(ii), **Prenom**: du type String.

#### (2). Adresse

Cette classe est pour créer les adresses à l'aide de l'interface Google map.

On a un attribut dans cette classe:

**Adresse**: du type String, cet attribut est pour stocker l'adresse des personnes ou des lieux.

On a deux méthodes dans cette classe:

(i), **Localiser()**: cette méthode réalise l'interface de Google Map, l'utilisateur peut se localiser par cette méthode.

(ii), **Entrer\_l\_adresse()**: du type String. L'utilisateur peut entrer l'adresse à la main.

#### (3). Transport

Cette classe est pour créer l'objet de transport au choix pour sortir.

On a un attribut dans cette classe:

**Mode\_Transport**: du type String, cet attribut est pour stocker le mode de transport.

On a une méthode dans cette classe:

**Choisi()**: du type Transport. Cette méthode permet l'utilisateur choisir le mode de transport pour sortir.

#### (4). Temps

Cette classe est pour créer l'objet heure.

On a deux attributs dans cette classe:

(i), **Heure**: du type Integer, cet attribut est pour stocker l'heure du temps.

(ii), **Minute**: du type Integer, cet attribut est pour stocker la minute du temps.

On a une méthode dans cette classe:

**Entrer\_l\_heure()**: du type Temps, cette méthode permet l'utilisateur d'entrer l'heure.

#### (5). Preference

Cette classe est pour créer l'objet Préférence d'alimentation,

On a un attribut dans cette classe:

**NomDePreference**: du type String, est pour stocker le nom de la préférence d'alimentation.

On a deux méthodes dans cette classe:

(i), **Choisir()**: du type Preference. Cette méthode permet l'utilisateur de choisir les préférences d'alimentation.

(ii), **Entrer()**: du type String. S'il n'y a pas la préférence que l'utilisateur veut dans la liste, cette méthode permet l'utilisateur entrer la préférence à la main.

#### (6). Lieu

Cette classe est pour créer l'objet Lieu.

On a cinq attributs dans cette classe:

(1), **Genre**: du type String. Cet attribut représente le genre du lieu, par exemple: Club, Resto, Pub, Piscine etc.

(2), **NonLieu**: du type String. Cet attribut stocke le nom du lieu. Par exemple: La tour d'argent, Guy Savoy, etc.

(3), **HeureOuverture**: du type Temps. Cet attribut stocke l'heure d'ouverture.

(4), **HeureFermeture**: du type Temps. Cet attribut stocke l'heure de fermeture.

(5), **PlatPrincipal**: du type Preference. Cet attribut indique le plat principal servi du lieu.

#### (7). Activite

Cette classe est pour créer l'objet activité.

On a trois attributs de cette classe:

**Nom**: du type String, Cet attribut stocke le nom de l'activité.

**LieuActivite**: du type Lieu. Cet attribut stocke le lieu où l'activité a lieu.

**Duree**: du type Temps. Cet attribut stocke la durée estimée.

On a deux méthodes de cette classe:

(i), **Convience(Temps, Temps, Temps, Temps)**: du type Boolean. Cette méthode prend 4 arguments, ces 4 arguments sont tout du type Temps. Le premier argument représente l'heure que l'invité compte de sortir. Le deuxième argument représente l'heure que l'invité prendra pour venir au lieu de l'activité. Le troisième

argument représente l'heure que l'heure d'ouverture du lieu. Le quatrième argument représente l'heure que l'heure de fermeture du lieu.

(ii), **CreerUneNouvelle()**: du type *Activite*. Cette méthode est pour créer une nouvelle activité s'il n'y a pas l'activité que l'utilisateur veut.

### Relations entre les objets engendrés par les classes:

On pose des objets engendrés par les classes:

Classe	Objet engendré
Invite	invité
Adresse	adresse
Transport	transport
Temps	L'heure de départ, L'heure sur place, L'heure de transport, L'heure d'ouverture, L'heure de fermeture.
Preference	préférence
Lieu	lieu
Activite	activité

- 1, L'invité habite à une adresse.
- 2, L'invité choisit l'heure de départ.
- 3, L'invité choisit de prendre un genre de mode transport.
- 4, Différents des modes de transport choisis ont des différentes heures de transport.
- 5, L'invité a des préférences d'alimentation.
- 6, L'invité participera à au moins une activité pendant une sortie.
- 7, Chaque lieu se localise à une adresse.
- 8, Chaque activité a lieu à un lieu.
- 9, Chaque activité a lieu avec une durée.