



N° Réf :

Centre Universitaire
Abdelhafid Boussouf Mila

Institut des Sciences et de la Technologie

Département de Mathématiques et Informatique

Mémoire de fin d'études pour l'obtention du diplôme de Master en Informatique

Spécialité : Sciences et technologies de l'information et de la communication

Conception et réalisation d'une application mobile sensible au contexte pour un hôtel.

Présenté par : **Benalouache Sassi Mohammed Lazehari**
Zerizer Abdenour

Encadrés par : **Afri Faiza** M.A.A C.U.Abd Elhafid Boussouf.

Président : **Hedjaz Sabrine** M.A.A C.U.Abd Elhafid Boussouf.

Examineur : **Aouag Mouna** M.C.B C.U.Abd Elhafid Boussouf.

Soutenu le ... /... / 2022 devant le jury.

Année Universitaire : 2021/2022

Dédicace

“

*À ma très chère honorable et aimable mère, et À mon très
cher père, Pour les choses qui ne peuvent être exprimées
par des mots ...*

*À mes frères, À mes amis et À tous ceux qui m'ont aidée à
terminer ce travail*

Merci.

”

- **Sassi**

Dédicace

“

À mes très chers parents, Que Dieu le tout puissant, vous comble de santé, de prospérité et vous accorde une longue vie afin que je puisse vous combler à mon tour...

À ma très chère sœur et À mes amis, Qu'Allah, le Très-Haut, fasse que le meilleur reste à venir

Merci.

”

- Abdenour

Remerciements

Tout d'abord, je remercie Allah le tout puissant de m'avoir donné le courage et la patience nécessaires à mener ce travail à son terme.

Je tiens à remercier tout particulièrement mon encadrante **Mme Faiza Afri**, pour l'aide compétente qu'elle m'a apportée, pour sa patience et son encouragement. Son œil critique m'a été très précieux pour structurer le travail et pour améliorer la qualité des différentes sections.

Je désire remercier également **Mme Mounira Zekiouk** pour les renseignements précieux qu'elle m'a fournis ainsi que pour son encouragement.

Que les membres de jury trouvent, ici, l'expression de mes sincères remerciements pour l'honneur qu'ils me font en prenant le temps de lire et d'évaluer ce travail.

Je souhaite aussi remercier tous les professeurs qu'ils m'aident à venir ici depuis le début de mes études.

Pour finir, je souhaite remercier toute personne ayant contribué de près ou de loin à la réalisation de ce travail.

Résumé

Les nouveautés informatiques continuent sans cesse de transformer le paysage technologique dans lequel nous vivons. Elle est désormais omniprésente sous différentes formes, nous accompagne partout. Les objets connectés participent pleinement de cette tendance. Tout ce la a contribué à la naissance d'un nombre énorme d'applications visant à faciliter le quotidien de l'humain tout en l'assistant selon son contexte.

Dans ce travail, nous avons développé un système constitué : d'une application mobile sensible au contexte au sein d'un hôtel. Cette dernière permet d'assister le client durant son séjour au sein de l'établissement (et même à distance) en se basant sur la gestion de son contexte (préférences et l'environnement). Aussi, d'une application qui assure la gestion des tâches de l'hôtel, et qui fournit des entrées pour alimenter l'application mobile.

Pour la simulation, nous avons réalisé une maquette, intégrant un système IoT en utilisant une carte Arduino avec différents capteurs. Ce système se charge de la collecte et l'exploitation des informations captés du contexte.

Nous avons utilisé la plateforme Android Studio pour la réalisation de l'application mobile. Arduino IDE pour la configuration des capteurs, Les autres parties du système sont développées en utilisant différents langages et environnements de développements à l'exemple : java, Firebase, etc.

Mots clés : Contexte, Sensibilité au contexte, IoT, Android, Java, Arduino, Capteur, Apprentissage automatique.

Abstract

Computer innovations continue to constantly transform the technological landscape in which we live. It is now omnipresent in different forms, accompanies us everywhere. Connected objects fully participate in this trend. All this has contributed to the birth of a huge number of applications aimed at facilitating the daily life of humans while assisting them according to their context.

In this work, we have developed a system consisting of : a context-aware mobile application for a hotel. The application makes it possible to assist the client during his stay (even remotely) based on the management of his context (preferences and environment). Also, an application that manages the hotel's tasks.

For the simulation, we made a model, integrating an IoT system using an Arduino card board with different sensors. It is responsible for collecting and using information captured from the context.

We used the Android Studio platform for the realization of the mobile application. Arduino IDE for configuring sensors. The other parts of the system are developed using different languages and development environments, for example : java, Firebase, etc.

Keywords : Context, context awareness, IoT, Android, java, Ardouino, sensor, machine learning.

Keywords : Context, Context awareness, IoT, Android, Java, Arduino, Sensor, Machine learning.

ملخص

تستمر التطورات في مجال الإعلام الآلي في تغيير المشهد التكنولوجي الذي نعيش فيه باستمرار. الإعلام الآلي موجود الآن في كل مكان بأشكال مختلفة، ويرافقنا في كل مكان. تشارك الأشياء المتصلة بشكل كامل في هذا الاتجاه. كل هذا ساهم في ولادة عدد هائل من التطبيقات التي تهدف إلى تسهيل الحياة اليومية للبشر مع مساعدتهم وفقًا لسياقهم.

في هذا العمل ، طورنا نظامًا يتكون من: تطبيق محمول حساس للسياق داخل فندق . يتيح التطبيق مساعدة العميل أثناء إقامته (حتى عن بُعد) بناءً على إدارة سياقه (التفضيلات والبيئة). أيضًا ، تطبيق آخر يدير مهام الفندق وهو مسؤول عن جمع واستخدام المعلومات التي تم التقاطها من السياق

من أجل المحاكاة ، انجزنا نموذجًا ، يدمج نظام إنترنت الأشياء باستخدام بطاقة Arduino مع أجهزة استشعار مختلفة استخدمنا نظام Android الأساسي لتحقيق تطبيق الهاتف المحمول. IDE Arduino لبرمجة المستشعرات ، تم تطوير الأجزاء الأخرى من النظام باستخدام لغات وبيئات تطوير مختلفة ، على سبيل المثال: java و Firebase وما إلى ذلك .

كلمات مفتاحية :

السياق، الوعي بالسياق، إنترنت الأشياء، جافا، أردوينو، المستشعر، التعلم الآلي.

Table des matières

Dédicace	I
Dédicace	II
Remerciements	III
Résumé	IV
Abstract	V
VI	ملخص
Introduction générale	1
1 La notion de contexte	3
1.1 Introduction	4
1.2 Le contexte	4
1.2.1 Définition	4
1.2.2 Catégories du contexte	5
1.2.3 Caractéristique du contexte	6
1.2.4 Acquisition du contexte	7
1.2.5 Modèles de contexte	7
1.3 La sensibilité au contexte	9
1.3.1 Définition	9
1.3.2 Niveaux de sensibilité au contexte	10
1.3.3 Les systèmes sensibles au contexte	10
1.3.4 Architecture des systèmes sensibles au contexte	11
1.4 Informatique ubiquitaire	13
1.4.1 Définition	13
1.4.2 Environnements ubiquitaires	14
1.4.3 Systèmes ubiquitaires	14
1.4.4 Propriétés des systèmes ubiquitaires	15
1.4.5 Défis de l'informatique ubiquitaire	16
1.5 Conclusion	17
2 Internet des objets	18
2.1 Introduction	19
2.2 Internet des objets	19
2.3 Objets connectés	20

2.3.1	Définitions	20
2.3.2	Caractéristiques d'un objet connecté	20
2.3.3	Composants d'un objet connecté	20
2.4	Système IoT	22
2.4.1	Définition	22
2.4.2	Composants d'un système IoT	22
2.4.3	Protocoles de communication	23
2.5	Modèles de communication d'Internet des objets	24
2.5.1	Périphérique à périphérique	24
2.5.2	Périphérique à cloud	24
2.5.3	Périphérique à passerelle	24
2.5.4	Partage des données en amont (Back-End)	24
2.6	Domaines d'application	25
2.6.1	Smart Home	25
2.6.2	Santé	26
2.7	Conclusion	27
3	Applications et Systèmes d'exploitation Mobiles	28
3.1	Introduction	29
3.2	Applications mobiles	29
3.2.1	Définition	29
3.2.2	Types d'applications mobiles	29
3.2.3	Domaines d'application mobile	30
3.2.4	Avantages et inconvénients d'une application mobile	32
3.3	Systèmes d'exploitation mobiles	33
3.3.1	Définition	33
3.3.2	Types des systèmes d'exploitation mobiles	33
3.4	Système d'exploitation Android	36
3.4.1	Histoire du système d'exploitation Android	36
3.4.2	Versions d'Android	36
3.4.3	Architecture Android	37
3.5	Développement d'applications Android	38
3.5.1	Composants Principaux d'Android	39
3.5.2	Outils de développement d'application Android	40
3.6	Conclusion	41
4	Analyse et Conception	42
4.1	Introduction	43
4.2	Processus unifié	43
4.2.1	Inception	43
4.2.2	Elaboration	43
4.2.3	Construction	44
4.2.4	Transition	44
4.3	Langage de modélisation UML	44
4.4	Analyse	44
4.4.1	Identification des acteurs	44

4.4.2	Diagramme de cas d'utilisation	45
4.4.3	Spécification des cas d'utilisation	48
4.5	Conception	77
4.5.1	Diagramme de classes participantes	77
4.5.2	Diagramme de séquence détaillé	90
4.5.3	Diagramme de classes conception	104
4.5.4	Modèle de base de donnée	106
4.6	Conception de la maquette de simulation	107
4.6.1	Définition des besoins	107
4.6.2	Modèle de communication	107
4.6.3	Schéma de création	108
4.6.4	Conclusion	109
5	Implémentation et Réalisation	110
5.1	Introduction	111
5.2	Langages et outils utilisés	111
5.2.1	Langage Java	111
5.2.2	Android Studio	111
5.2.3	Arduino IDE	112
5.2.4	Firebase	113
5.3	Système de capture de contexte	114
5.3.1	Composants matériels utilisés	115
5.3.2	Raccordement des composants du système	120
5.3.3	Bibliothèque Arduino IDE	124
5.3.4	Réalisation du système	125
5.4	Applications Android de l'hôtel	129
5.4.1	Librairies et API utilisés	129
5.4.2	Réalisation de l'application de gestion de l'hôtel	129
5.4.3	Réalisation de l'application mobile	131
5.5	Sensibilité au contexte du système	133
5.5.1	Notification et prévention	133
5.5.2	Classification naïve bayésienne	134
5.6	Conclusion	136
	Conclusion générale	137

Table des figures

1.1	Catégorisation du contexte	6
1.2	Principe des systèmes sensible au contexte	11
1.3	Architecture des systèmes sensible au contexte	13
1.4	Comparaison de différents dispositifs informatiques [10]	16
2.1	Exemples de capteurs	21
2.2	Exemples d'actionneurs	21
2.3	Exemples de Modules de connectivité	22
2.4	Modèles de communication de l'Internet des objets [36]	25
2.5	Smart Home [17]	26
3.1	IHM application mobile The Hilton Honors	31
3.2	IHM application mobile Marriott International	31
3.3	Logo Systèmes d'exploitation IOS	33
3.4	Logo Systèmes d'exploitation Android	34
3.5	Logo Systèmes d'exploitation Windows 10 Mobile	34
3.6	Logo Systèmes d'exploitation Tizen	35
3.7	Logo Systèmes d'exploitation Sailfish OS	35
3.8	Architecture des couches Android	38
4.1	Diagramme de cas d'utilisation de l'application mobile	46
4.2	Diagramme de cas d'utilisation de l'application gestion de l'hôtel	47
4.3	Diagramme du fonctionnement de notre système de capture	48
4.4	Diagramme de séquence S'authentifier	50
4.5	Diagramme de séquence Créer compte	52
4.6	Diagramme de séquence Réservation de chambre	54
4.7	Diagramme de séquence Mettre à jour informations compte	55
4.8	Diagramme de séquence Valider code séjour	57
4.9	Diagramme de séquence Consulter des centres d'intérêt près de l'hôtel	58
4.10	Diagramme de séquence Gérer Smart Room	59
4.11	Diagramme de séquence Demander Room service	61
4.12	Diagramme de séquence Consulter Menu restaurant	62
4.13	Diagramme de séquence Consulter événements	64
4.14	Diagramme de séquence Gestion des réservations	67
4.15	Diagramme de séquence Check-in	69
4.16	Diagramme de séquence Check-out	71
4.17	Diagramme de séquence Gestion du menu restaurant	74
4.18	Diagramme de séquence Gestion des événements	76
4.19	Diagramme de classes participantes S'authentifier	78

4.20	Diagramme de classes participantes Consulter les informations de l'hôtel .	78
4.21	Diagramme de classes participantes Créer compte	79
4.22	Diagramme de classes participantes Réservation de chambre	80
4.23	Diagramme de classes participantes Mettre à jour informations compte . .	80
4.24	Diagramme de classes participantes Gérer préférences	81
4.25	Diagramme de classes participantes Valider code séjour	81
4.26	Diagramme de classes participantes Consulter des centres d'intérêt près de l'hôtel	82
4.27	Diagramme de classes participantes Gérer Smart Room	82
4.28	Diagramme de classes participantes Demander Room service	83
4.29	Diagramme de classes participantes Consulter Menu restaurant	83
4.30	Diagramme de classes participantes Consulter événements	84
4.31	Diagramme de classes participantes Gestion des réservations	85
4.32	Diagramme de classes participantes Check-in	86
4.33	Diagramme de classes participantes Check-out	87
4.34	Diagramme de classes participantes Consulter liste résidents	87
4.35	Diagramme de classes participantes Consulter les informations Capteurs .	88
4.36	Diagramme de classes participantes Gestion du menu restaurant	88
4.37	Diagramme de classes participantes Gestion des événements	89
4.38	Diagramme de classes participantes Gestion du Room Service	89
4.39	Diagramme de séquence détaillé S'authentifier	90
4.40	Diagramme de séquence détaillé Créer compte	91
4.41	Diagramme de séquence détaillé Réservation de chambre	92
4.42	Diagramme de séquence détaillé Mettre à jour informations compte	93
4.43	Diagramme de séquence détaillé Gérer préférences	94
4.44	Diagramme de séquence détaillé Valider code séjour	95
4.45	Diagramme de séquence détaillé Consulter des centres d'intérêt près de l'hôtel	96
4.46	Diagramme de séquence détaillé Gérer Smart Room	97
4.47	Diagramme de séquence détaillé Demander Room service	98
4.48	Diagramme de séquence détaillé Consulter Menu restaurant	99
4.49	Diagramme de séquence détaillé Gestion des réservations	101
4.50	Diagramme de séquence détaillé Check-in	102
4.51	Diagramme de séquence détaillé Check-out	103
4.52	Diagramme de séquence détaillé Gestion du menu restaurant	104
4.53	Diagramme de classes conception	105
4.54	Modèle de base de donnée	106
4.55	Schema de creation du système de capteurs	108
5.1	Logo Langage Java	111
5.2	Logo Android Studio	112
5.3	Logo Arduino IDE	113
5.4	Logo Firebase	114
5.5	Carte Arduino Mega 2560	115
5.6	Lecteur RFID RC522	116
5.7	Capteur de température et d'humidité DHT22	116

5.8	Capteur de Pluie	117
5.9	Servomoteur	117
5.10	Capteur de luminosité TSL2561	118
5.11	Capteur de Mouvement	118
5.12	Module WiFi ESP8266-01	119
5.13	LCD	119
5.14	Breadboard	120
5.15	Alimentation rattacher à la breadboard MB-102	120
5.16	Raccordement des lecteurs RFID RC522	121
5.17	Raccordement du capteur de température et d'humidité DHT22	121
5.18	Raccordement du capteur de luminosité TSL2561	122
5.19	Raccordement du capteur de Mouvement	122
5.20	Raccordement du Servomoteur	123
5.21	Raccordement du LCD	123
5.22	Raccordement du Module WiFi ESP8266-01	124
5.23	Images 1 de la maquette	126
5.24	Images 2 de la maquette	126
5.25	Images 3 de la maquette	127
5.26	Images 4 de la maquette	127
5.27	Images 5 de la maquette	128
5.28	Données contextuelles dans BDD	128
5.29	Interface application de gestion menu	129
5.30	Interface application de gestion réservation	130
5.31	Interface application de gestion check-in	130
5.32	Interface application de gestion check-out	131
5.33	Interface application de gestion évènements	131
5.34	Interface 1 application mobile	132
5.35	Interface 2 application mobile	132
5.36	Interface 3 application mobile	133
5.37	Exemple de Data set	135

Liste des tableaux

3.2	Versions du Système d'exploitation Android	37
4.2	Description du cas d'utilisation S'authentifier	49
4.4	Description du cas d'utilisation Consulter les informations de l'hôtel	51
4.6	Description du cas d'utilisation Créer compte	51
4.8	Description du cas d'utilisation Réservation de chambre	53
4.10	Description du cas d'utilisation Mettre à jour informations compte	55
4.12	Description du cas d'utilisation Gérer préférences	56
4.14	Description du cas d'utilisation Valider code séjour	56
4.16	Description du cas d'utilisation Consulter des centres d'intérêt près de l'hôtel	58
4.18	Description du cas d'utilisation Gérer smart Room	59
4.20	Description du cas d'utilisation Demander Room service	60
4.22	Description du cas d'utilisation Consulter Menu restaurant	62
4.24	Description du cas d'utilisation Consulter événements	63
4.26	Description du cas d'utilisation Gestion des réservations	65
4.28	Description du cas d'utilisation Check de disponibilité	68
4.30	Description du cas d'utilisation Check-in	68
4.32	Description du cas d'utilisation Check-out	70
4.34	Description du cas d'utilisation Consulter liste résidents	72
4.36	Description du cas d'utilisation Consulter les informations Capteurs	72
4.38	Description du cas d'utilisation Gestion du menu restaurant	73
4.40	Description du cas d'utilisation Gestion des événements	75
4.42	Description du cas d'utilisation Gestion du Room Service	77

Liste des sigles et acronymes

GPS	<i>Global Positioning System</i>
API	<i>Application Programming Interface</i>
HTTP	<i>Hypertext Transfer Protocol</i>
JSON	<i>JavaScript Object Notation</i>
Web	<i>World Wide Web</i>
RFID	<i>Radio Fréquence IDentification</i>
XML	<i>eXtensible Markup Language</i>
URL	<i>Uniform Resource Locator</i>
UML	<i>Unified Modeling Language</i>
PDA	<i>JPersonal Digital Assistant</i>
PC	<i>Personal Computer</i>
IoT	<i>Internet of Things</i>
LED	<i>Light-Emitting Diode</i>
LCD	<i>Liquid Crystal Display</i>
GSM	<i>Global System for Mobile</i>
NFC	<i>Near Field Communication</i>
UDP	<i>User Datagram Protocol</i>
TCP	<i>Transmission Control Protocol</i>

REST	<i>Representational State Transfer</i>
IP	<i>Internet Protocol</i>
IoHT	<i>Internet of Healthcare Things</i>
SMS	<i>Short Message System</i>
HTML	<i>HyperText Markup Language</i>
SDK	<i>Software Development Kit</i>
DVM	<i>Dalvik Virtual Machine</i>

Introduction générale

Contexte

La technologie mobile, alimentée par l'internet et la présence d'applications, est une évolution qui change la donne pour divers secteurs en ce qui concerne l'amélioration du service et de l'engagement des consommateurs.

La technologie mobile dans l'industrie hôtelière qui est censée être une industrie de services a créé de nouveaux paramètres pour ces dernières en créant des expériences personnalisées et transparentes pour les clients.

À l'ère de la technologie actuelle, les entreprises du secteur de l'hôtellerie adoptent une approche différente pour stimuler la croissance. Comme l'expérience du client doit être de premier ordre, les hôtels s'appuient sur **les applications mobiles** et des systèmes intelligents tel que **les chambres intelligentes** pour une expérience unique. Par conséquent, disposer d'une application solide dans ce secteur est une nécessité pour les hôtels et les marques d'hôtellerie en général.

Des fonctionnalités solides au sein de l'application peuvent offrir une expérience client complète. Elles créent ainsi un pont entre l'utilisateur et l'hôtel. Les applications dotées de telles fonctionnalités favorisent la fidélisation des clients. La fidélisation des clients conduit à la recommandation, ce qui conduit à la croissance.

Problématique

L'hôtellerie est un des services les plus dépendants de la satisfaction du client, Afin de rester compétitifs, les hôteliers sont obligés d'être à l'écoute à tout instant et exhausser les désirs des résidents tout en restant proche de lui, même s'il est en dehors de l'établissement ce qui s'avère très difficile à accomplir vu l'effectif du personnel et les moyens déployés.

La problématique peut alors être présentée ainsi : comment garantir une prestation de qualité et satisfaire le client. Comment permettre aux hôteliers d'économiser les coûts, augmenter leurs chiffres d'affaires et offrir une meilleure expérience au client. Comment assister le client minute par minute durant son séjour et parvenir à satisfaire ses désirs sans même qu'il en demande, même s'il est loin de l'établissement ?

Objectifs

L'objectif de notre travail est de développer une **application mobile sensible au contexte** pour un hôtel. Cette dernière sera en mesure de découvrir des services et les adapter selon le contexte des utilisateurs afin de parvenir à satisfaire leurs besoins avec un minimum d'action de leur part.

Cela implique aussi la conception d'un système intelligent pour la capture des informations contextuelles, une **chambre intelligente** (Smart room) à l'aide de l'**Internet des objets** ou IoT.

Organisation du mémoire

Ce mémoire est organisé en cinq chapitres :

Le premier chapitre "**La notion de contexte**" a pour objectif de présenter les trois notions : le contexte, la sensibilité au contexte, et l'informatique ubiquitaire. La première définit toutes les notions relatives au contexte, la deuxième qui est dédiée à l'étude de la sensibilité des applications au changement du contexte, et enfin l'informatique ubiquitaire et sa relation avec le contexte.

Le deuxième chapitre "**Internet des objets**" a pour objectif de présenter un aperçu général sur l'IoT. Nous allons aborder le concept d'Internet des objets, et toutes les notions relatives aux objets connectés et au Système IoT.

Le troisième chapitre "**Applications et Systèmes d'exploitation Mobiles**" expose ce qu'est l'informatique mobile, quelques concepts de base des applications mobiles, nous évoquerons le système Android, son architecture et le développement d'applications Android.

Le quatrième chapitre "**Analyse et Conception**" nous essaierons d'apporter des éléments de réponse à notre thématique, en expliquant l'analyse et la conception du système crée : cette partie contiendra les différents diagrammes ainsi que les spécifications des besoins systèmes, ainsi que l'architecture globale du système.

Le cinquième chapitre "**Implémentation et Réalisation**" où nous exposerons le matériel utilisé, les techniques d'implémentation que nous avons mises en œuvre et nous essayons de donner un aperçu final à travers quelques interfaces des applications réalisées et de la maquette construit pour la simulation.

Le document se termine par une conclusion générale qui présente une synthèse du travail ainsi que les perspectives que nous avons tracées pour l'amélioration.

Chapitre 1

La notion de contexte

1.1 Introduction

Les humains réussissent assez bien à se transmettre des idées et à réagir de manière appropriée. Cela est dû à de nombreux facteurs : la richesse du langage qu'ils partagent, la compréhension commune du fonctionnement du monde et une compréhension implicite des situations quotidiennes. Lorsque des humains communiquent entre eux, ils peuvent utiliser des informations situationnelles implicites, ou le contexte. Malheureusement, cette capacité à transmettre des idées ne se transfère pas bien aux humains qui interagissent avec des ordinateurs. Dans l'informatique interactive traditionnelle, les utilisateurs disposent d'un mécanisme pauvre pour fournir des données aux ordinateurs. Par conséquent, les ordinateurs ne sont pas actuellement en mesure de tirer pleinement parti du contexte.

En améliorant l'accès de l'ordinateur au contexte, nous augmentons la richesse de la communication dans l'interaction homme-ordinateur et rendons possible la production de services informatiques plus utiles.

Pour utiliser le contexte efficacement, nous devons comprendre à la fois ce qu'est le contexte et comment il peut être utilisé. La compréhension du contexte permettra aux concepteurs d'applications de choisir le contexte à utiliser dans leurs applications. Une compréhension de la façon dont le contexte peut être utilisé aidera les concepteurs d'applications à déterminer les comportements contextuels à prendre en charge dans leurs applications.

Dans ce chapitre nous passerons en revues trois parties essentielles : le contexte, la sensibilité au contexte et l'informatique ubiquitaire. La première définit toutes les notions relatives au contexte, la seconde est dédiée à l'étude de la sensibilité des applications au changement du contexte et la troisième évoque l'informatique et les systèmes ubiquitaires et leur relation avec le contexte.

1.2 Le contexte

1.2.1 Définition

Ce terme a été abordé pour la première fois par Schilit et Theimer en 1994 [65], ils font référence au contexte comme étant l'emplacement, l'identité des personnes et des objets à proximité, et les changements apportés à ces objets.

Dans une définition similaire, Brown [13] définit le contexte comme étant l'emplacement, l'identité des personnes autour de l'utilisateur, l'heure et la date du jour, la saison, la température, etc.

Une Définition linguistique par l'encyclopédie Larousse [26] définit le mot contexte comme étant "l'ensemble des conditions naturelles, sociales, culturelles dans lesquelles se situe un énoncé ou un discours. Il est aussi l'ensemble des circonstances dans lesquelles se produit un événement où se situe une action".

Différents chercheurs se sont intéressés à définir et clarifier la notion de contexte. En effet, il est difficile de trouver d'une manière claire et unique, une définition valable dans

tous les travaux impliquant cette notion. La plupart des chercheurs ont une idée générale de ce qu'est le contexte et utilisent cette idée pour guider leur utilisation. Cependant, une vague notion du contexte n'est pas suffisante afin d'utiliser le contexte de manière efficace, nous devons parvenir à une meilleure compréhension de ce qu'est le contexte, Cette dernière permettra aux concepteurs d'applications de choisir le contexte à utiliser dans leurs applications et de fournir des informations sur les types de données qui doivent être pris en charge et sur les abstractions et les mécanismes nécessaires à la prise en charge du contexte.

Une définition plus globale que le contexte est toute information qui peut être utilisée pour caractériser la situation d'une entité. Une entité est une personne, un lieu ou un objet qui est considéré comme pertinent pour l'interaction entre un utilisateur et une application, y compris l'utilisateur et l'application elle-même [22].

Noter définition qui est plus appropriée dans le cadre de notre projet est celle qui considère le contexte comme étant ” **Tout changement de valeur ou d'État qui engendre une fonctionnalité ou un changement de résultat ou de comportement de l'application** ”.

1.2.2 Catégories du contexte

Il existe de nombreuses façons de catégoriser les informations contextuelles. Deux méthodes de catégorisation typiques sont la catégorisation conceptuelle et la catégorisation opérationnelle [50].

Catégorisation conceptuelle

Les informations contextuelles sont généralement classées sous quatre catégories, Il s'agit du lieu, le temps, l'activité et l'identité [22].

Ces catégories de contexte ne répondent pas seulement aux questions de qui, quoi, quand et où, mais servent également d'indices pour d'autres sources d'information contextuelles. Par exemple, à partir de l'identité d'une personne, nous pouvons acquérir de nombreux éléments d'information connexes tels que des numéros de téléphone, des adresses, une date de naissance, des adresses électroniques, une liste d'amis, des relations avec d'autres personnes dans l'environnement etc.

Grâce à la localisation d'une entité, nous pouvons déterminer quels autres objets ou personnes se trouvent près de l'entité et quelle activité se déroule à proximité etc.

Catégorisation opérationnelle

Le contexte est divisé en deux parties, primaire et secondaire. Un contexte primaire est une valeur qui est collectée par le capteur lui-même. Un contexte secondaire est obtenu par le traitement de plusieurs contextes primaires [52] :

- **Contexte primaire :**

Toute information récupérée sans utiliser de contexte existant et sans effectuer aucune sorte d'opérations de fusion de données de capteurs (par exemple les relevés du capteur GPS comme la localisation).

- **Contexte secondaire :**

Toute information qui peut être calculée en utilisant le contexte primaire. Le contexte secondaire peut être calculé à l'aide d'opérations de fusion de données de capteurs ou d'opérations d'extraction de données telles que des appels de services Web (par exemple, identifier la distance entre deux capteurs en appliquant des opérations de fusion de données de capteurs sur deux valeurs brutes de capteurs GPS).

En outre, le contexte récupéré tel que des numéros de téléphone, des adresses, des adresses électroniques, liste d'amis à partir d'un fournisseur d'informations de contact, sur la base d'une identité personnelle en tant que contexte primaire peut également être identifié comme un contexte second.

Catégorisation opérationnelle		
	Contexte primaire	Contexte secondaire
Lieu	Localisation GPS Fixe, Localisation basée RFID	Localisation intérieur ou extérieur, Fixe ou en mouvement etc.
Temps	L'heure et la date réelle de l'horloge	Emploi du temps etc.
Activité	vitesse depuis l'accéléromètre	Course / marche / à l'arrêt etc.
Identité	Adresses Mail dans la base de donnés utilisateur	Liste connaissance / Amis etc.

FIG. 1.1 : Catégorisation du contexte

1.2.3 Caractéristique du contexte

Les caractéristiques du contexte sont fortement influencées et déterminées par son mode d'acquisition [27]. Un contexte peut être statique : les données ne changent pas au cours du temps et sont collectées au lancement de l'application, mais peuvent être changer manuellement (Profil de l'utilisateur, préférence, etc.), ou dynamique les données changent continuellement, ce sont les informations majoritairement acquis par les capteurs.

1.2.4 Acquisition du contexte

En fonction du domaine d'application, différentes méthodes ont été utilisées pour obtenir le contexte, on distingue trois méthodes d'acquisition [44] :

- **Contexte sensoriel :**

Ce type d'information est acquis par l'intermédiaire de capteurs physiques ou logiciels tels que la température, la pression, l'éclairage et le niveau de bruit etc.

- **Contexte dérivé :**

Ce type d'informations contextuelles peuvent être calculées à la volée. Les exemples les plus illustratifs sont l'heure et la date.

- **Contexte explicitement fourni :**

Par exemple, les préférences de l'utilisateur lorsqu'elles sont explicitement communiquées par l'utilisateur à l'application.

L'acquisition du contexte peut se faire en utilisant divers systèmes de détection, d'enregistrement et de positionnement comme le GPS, RFID et les réseaux de capteurs pour la détection de l'emplacement.

L'acquisition de contexte à partir de capteurs n'est cependant pas facile. Cette situation est due aux raisons principales suivantes : l'information peut être acquise à partir de différents capteurs et peut nécessiter une étape supplémentaire (interprétation) afin d'obtenir un contexte utile à une application.

De plus, les informations contextuelles peuvent être dynamiques par nature et ces changements doivent être saisis par l'infrastructure de détection, l'information contextuelle provient de sources hétérogènes et distribués et la technique de détection correspondante est généralement fixée à une source spécifique, c'est une mauvaise pratique d'un point de vue de l'ingénierie logicielle, car elle empêche la réutilisation du code d'application lorsque les capteurs changent.

1.2.5 Modèles de contexte

Un modèle de contexte est nécessaire pour définir et stocker les données contextuelles sous une forme exploitable par une machine. Pour développer des ontologies de contexte flexibles et utilisables qui couvrent le large éventail de contextes possible est une tâche difficile. Strang et Linnhoff-Popien 2004 [71] ont résumé les approches de modélisation des contextes les plus pertinents, qui sont basés sur les structures de données utilisées pour représenter et échanger des informations dans le système concerné [9, 42], on distingue six approches :

Modèles clés-valeurs :

Ces modèles représentent la structure de données la plus simple pour la modélisation du contexte. Ils sont fréquemment utilisés dans divers services d'infrastructure logicielle, où les paires clés-valeur sont utilisées pour décrire les capacités d'un service. La découverte de services est ensuite appliquée en utilisant des algorithmes de correspondance qui utilisent ces paires clés-valeur.

Modèle de présentation par balises :

Tous les modèles basés sur le balisage utilisent une structure de données hiérarchique constituée de balises avec des attributs et du contenu. Les profils représentent des modèles de schéma de balisage. Comme elle est formée de balises, elle utilise des langages dérivés du SGML (Standard Generic Markup Language) en particulier, le XML (eXtended Markup Language). Les entités du contexte sont exprimées sous forme de pages web qui peuvent être extraites par un URL. Un exemple de cette approche sont les profils qui décrivent les ensembles de données.

Modèles graphiques :

Le langage de modélisation unifié (UML) est également adapté à la modélisation du contexte. Il existe plusieurs approches où les aspects contextuels sont modélisés en utilisant UML. Cette approche consiste à modéliser les informations contextuelles selon un graphe conceptuel pour représenter les attributs contextuels et leurs relations.

Modèles orientés objet :

La modélisation du contexte en utilisant cette approche permet d'utiliser toute la puissance de l'orientation objet (encapsulation, réutilisation, héritage, etc.). Les approches existantes utilisent divers objets pour représenter différents types de contexte (tels que la température, l'emplacement, etc.) et encapsulent les détails du traitement et de la représentation du contexte. L'accès au contexte et la logique de traitement du contexte sont assurés par des interfaces définies [9].

Modèles basés sur la logique :

Les modèles basés sur la logique ont un haut degré de formalité. En général, des faits, des expressions et des règles sont utilisées pour définir un modèle de contexte. Un système basé sur la logique est ensuite utilisé pour gérer les termes susmentionnés et permet d'ajouter, de mettre à jour ou de supprimer de nouveaux faits. Le processus d'inférence (également appelé raisonnement) peut être utilisé pour dériver de nouveaux faits basés sur les règles existant dans les systèmes. Les informations contextuelles doivent être représentées de manière formelle sous forme de faits.

Modèles basés sur l'ontologie :

Les ontologies représentent une description des concepts et des relations. Par conséquent, les ontologies sont un instrument très prometteur pour la modélisation des informations contextuelles en raison de leur grande expressivité formelle et les possibilités d'appliquer des techniques de raisonnement ontologique. Différentes infrastructures logicielles sensibles au contexte utilisent des ontologies comme modèles de contexte. Les ontologies sont les modèles les plus expressifs et ce qui remplit la plupart de leurs exigences Strang et Linnhoff-Popien 2004 [71]. Korpipää et al. 2003 [35] présentent certaines exigences et certains objectifs après avoir conçu une ontologie de contexte qui sont :

- **La simplicité** : Les expressions et relations utilisées doivent être aussi simples que possible pour simplifier le travail des développeurs d'applications.
- **La flexibilité et l'extensibilité** : L'ontologie doit permettre l'ajout simple de nouveaux éléments de contexte et relations.
- **La généricité** : L'ontologie ne doit pas être limitée à un type particulier de contexte mais plutôt supporter différents types.
- **L'expressivité** : L'ontologie doit permettre de décrire autant d'états de contexte que possible dans des détails arbitraires.

1.3 La sensibilité au contexte

1.3.1 Définition

Dans l'ouvrage qui introduit pour la première fois le terme " context aware ", par Schilit et Theimer en 1994 [65] comme étant "être un logiciel qui s'adapte en fonction de son lieu d'utilisation, de la collection de personnes et d'objets à proximité, ainsi que des changements apportés à ces objets au fil du temps".

Brown 1998 [13] définit les applications sensibles au contexte comme des applications qui fournissent automatiquement des informations et/ou prennent des mesures selon le contexte actuel de l'utilisateur tel qu'il est détecté par les capteurs. Il adopte également une vision étroite de l'informatique contextuelle en déclarant que ses actions peuvent prendre la forme de la présentation d'informations à l'utilisateur, de l'exécution d'un programme en fonction du contexte, ou la configuration d'une présentation graphique en fonction du contexte.

Une définition qui convient dans la vision qu'on a sur la sensibilité au contexte pour créer une application est : **Un système est conscient du contexte s'il utilise le contexte pour fournir des informations et/ou des services pertinents à l'utilisateur, la pertinence dépendant de la tâche de l'utilisateur** [2].

1.3.2 Niveaux de sensibilité au contexte

La sensibilité au contexte peut être identifiée en trois niveaux basés sur l'interaction avec l'utilisateur [23] :

Personnalisation :

Il permet aux utilisateurs de définir leurs préférences, leurs goûts et leurs attentes à l'égard du système. Par exemple, les utilisateurs peuvent définir la température préférée dans une chambre de l'hôtel où le système de chauffage peut maintenir la température spécifiée dans toutes les pièces.

Sensibilité au contexte passive :

Le système surveille constamment l'environnement et offre les options appropriées aux utilisateurs pour qu'ils puissent agir. Par exemple, lorsqu'un utilisateur entre dans le restaurant de l'hôtel, le téléphone mobile l'avertit en lui proposant le menu du jour.

Sensibilité au contexte active :

Le système surveille de manière continue et autonome la situation et agit de manière autonome. Par exemple, dans un hôtel, si les détecteurs de fumée et les capteurs de température détectent un incendie dans une chambre intelligente, le système prévient automatiquement les pompiers et le résident de la chambre.

1.3.3 Les systèmes sensibles au contexte

Un système est dit sensible au contexte s'il est capable de s'adapter aux changements contextuels en temps réel. Selon M. Chebbi [16], il existe trois étapes indispensables pour qu'un système soit qualifié de « sensible au contexte » :

- Capture et stockage du contexte d'une façon transparente pour l'utilisateur du système.
- Interprétation du contexte capté afin de changer l'état de représentation des services qu'offre le système. Ces derniers seront capables d'être exploités par les utilisateurs. Ces tâches forment implicitement leurs préférences.
- Passage des informations interprétées à l'application afin de garantir l'exécution des fonctionnalités selon les changements du contexte sans l'intervention des utilisateurs.

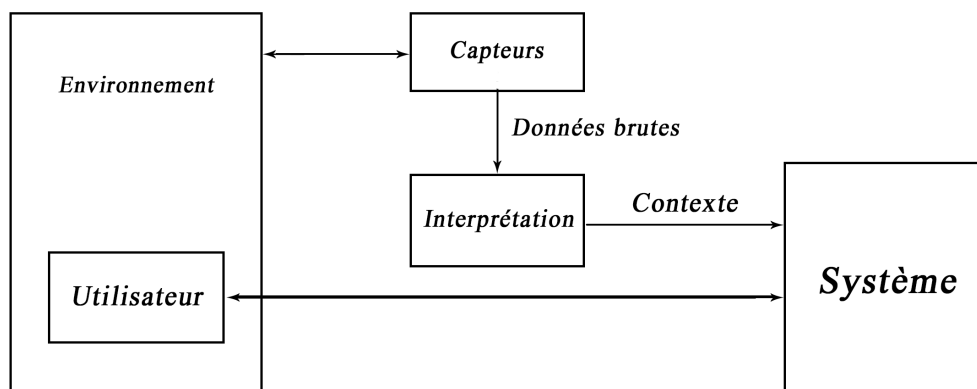


FIG. 1.2 : Principe des systèmes sensible au contexte

1.3.4 Architecture des systèmes sensibles au contexte

Les systèmes sensibles au contexte peuvent être mis en œuvre de nombreuses façons différentes en fonction de nombreuses considérations telles que les besoins individuels, le nombre d'utilisateurs, les types de dispositifs et les méthodes d'acquisition du contexte, etc.

L'architecture du système est nécessaire à la représentation et à la mise en œuvre du système, elle est une abstraction qui est utilisée pour généraliser les systèmes sans montrer le détail de l'implémentation.

Comme tout autre système, le système contextuel nécessite une architecture de système flexible. Les travaux existants illustrent l'évolution des architectures contextuelles, qui passent d'une application à usage spécifique à une application à usage général.

Inspiré par des concepts utilisés dans des outils d'interaction, Dey [22] a défini quelques abstractions qui aident à créer une interprétation de haut niveau du contexte, et qui supporte la séparation entre la gestion du contexte et l'application. Cette idée a été reprise par la majorité des travaux actuels dans le domaine de la sensibilité au contexte [9], proposant ainsi des architectures à plusieurs couches. Cependant, nous remarquons que toutes ces propositions se basent sur cinq couches principales : capture, interprétation, stockage, dissémination et application du contexte.

Capture du contexte

Composée d'une collection de capteurs matérielle ou logicielle qui peuvent générer une information contextuelle. Nous distinguons trois types de capteurs :

- **Capteurs physiques :**

Dispositifs matériels qui fournissent des données contextuelles. Par exemple : un GPS qui renvoie une localisation et un accéléromètre qui donne la vitesse.

- **Capteurs virtuels :**

Ce sont des capteurs virtuels qui fournissent des informations contextuelles en se basant sur d'autres applications ou services logiciels. Par exemple : il est possible de détecter l'emplacement d'une personne en consultant son carnet électronique de rendez-vous. Ces capteurs sont moins précis mais sont beaucoup moins coûteux que les capteurs physiques.

- **Capteurs logiques :**

Ces capteurs peuvent réutiliser les informations contextuelles des capteurs physiques et virtuels pour fournir une information contextuelle de plus haut niveau. Par exemple, un capteur logique peut fournir l'emplacement d'un caissier dans un grand magasin en analysant les différentes sessions ouvertes sur les caisses du magasin.

Interprétation du contexte

Cette couche sert à l'analyse et à la transformation des données brutes fournies par la couche de capture du contexte dans d'autres formats de haut niveau qui sont plus faciles à manipuler et à utiliser [15]. La couche d'interprétation est responsable du raisonnement et de l'interprétation des informations contextuelles. Les capteurs interrogés dans la couche sous-jacente renvoient le plus souvent des données techniques qui ne sont pas appropriées à utiliser par les concepteurs d'applications. Ainsi, cette couche élève les résultats de la première couche à un niveau d'abstraction plus élevé, les transformations comprennent des opérations d'extraction et de quantification. Par exemple, la position GPS exacte d'une personne n'est pas forcément utile pour une application, mais le nom de la pièce dans laquelle se trouve la personne peut l'être.

Stockage et gestion du contexte

Cette couche consiste à organiser les données déjà capturées et interprétées puis les stocker pour les utiliser dans les couches supérieures.

Dissémination du contexte

Cette couche assure une transparence totale de la communication avec l'application, donc assure la transmission des différentes informations contextuelles à cette dernière. Elle offre des moyens de communication standards pour notifier l'application des changements de contextes et leur transmission à l'application.

Application du contexte

Cette couche est responsable de l'extraction des informations des différentes sources de données attachées à l'application, elle doit aussi implémenter les réactions nécessaires aux changements du contexte. La couche est représentée par l'application qui offre ses services aux utilisateurs.

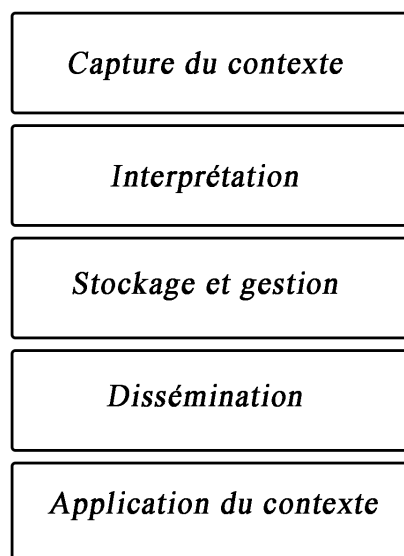


FIG. 1.3 : Architecture des systèmes sensible au contexte

1.4 Informatique ubiquitaire

1.4.1 Définition

L'informatique ubiquitaire, pervasive ou omniprésente, est le terme donné à la troisième ère de l'informatique moderne. La première ère était définie par l'ordinateur central, un seul grand ordinateur à temps partagé appartenant à une organisation et utilisé par de nombreuses personnes en même temps. Vient ensuite l'ère du PC, un ordinateur personnel détenu et utilisé principalement par une seule personne et qui lui est dédié.

La troisième ère, représentative de l'époque actuelle, se caractérise par l'explosion des petits ordinateurs portables, Ce qui donne lieu à un monde dans lequel chaque personne possède et utilise de nombreux ordinateurs. Chaque ère a entraîné l'intégration d'un nombre de plus en plus important d'ordinateurs dans la vie quotidienne.

Ce terme a été abordé pour la première fois par Marc Weiser en 1991 [74], signifie une informatique omniprésente, c'est-à-dire, qui existe partout. Ce terme est également utilisé pour décrire une disponibilité totale, donc les utilisateurs peuvent accéder à l'information et aux services souhaités, au moment voulu et dans n'importe quel endroit.

L'informatique ubiquitaire (ubicomputing) vise à créer un tout nouveau paradigme d'environnement informatique à presque tous ces égards [11]. Elle s'adresse à un ensemble hétérogène de dispositifs, notamment des ordinateurs invisibles intégrés dans des objets du quotidien tel que les voitures et les meubles, des dispositifs mobiles tels que les assistants numériques personnels (PDA) et les smart-phones, des dispositifs personnels tels que les ordinateurs portables, et de très grands dispositifs tels que les écrans muraux et les ordinateurs de table situés dans les environnements et les bâtiments que nous habitons.

Tous ces appareils ont des systèmes d'exploitation, des interfaces de réseau, des capa-

cités d'entrée et d'affichage. Certains sont conçus pour l'interaction avec l'utilisateur final, par exemple comme un affichage public dans une cafétéria, alors que d'autres dispositifs, comme les capteurs ne sont pas utilisés directement par les utilisateurs finaux. Le mode d'interaction va au-delà du modèle "un à un" qui prévaut pour les PC, pour passer à un modèle "plusieurs à plusieurs" où la même personne utilise plusieurs dispositifs, plusieurs personnes utilisent plusieurs dispositifs et plusieurs personnes peuvent utiliser le même dispositif.

L'interaction peut être implicite, invisible, ou par la détection d'interactions naturelles telles que la parole, le geste ou la présence : un large éventail de capteurs est nécessaire, des capteurs intégrés aux dispositifs ainsi que des capteurs intégrés à l'environnement. Des dispositifs de localisation, des caméras peuvent être utilisées pour détecter qui se trouvent dans un lieu et déduire ce qu'il fait. Ces informations peuvent être utilisées pour fournir à l'utilisateur des informations pertinentes dans un lieu spécifique ou pour l'aider à adapter son appareil à un environnement local ou d'adapter l'environnement à l'utilisateur.

Des environnements Ubicomp faisant appel à des techniques tels que celles décrites ci-dessus ont été créés pour un certain nombre de domaines d'application, notamment les chambres intelligentes, les salles de classe, les voitures, les hôpitaux, la maison, les voyages et les musées etc.

1.4.2 Environnements ubiquitaires

Un environnement ubiquitaire est un fonctionnement de plusieurs objets qui communiquent, se reconnaissent et se localisent automatiquement entre eux. Les objets interagissent entre eux sans une intervention de l'utilisateur [12].

1.4.3 Systèmes ubiquitaires

Le concept de système ubiquitaire, également connu sous le nom d'omniprésent, est des dispositifs informatiques miniaturisés intégrés de manière invisible dans notre environnement quotidien pour aider les gens dans leurs activités quotidiennes.

Ils existent depuis plus de 30 ans depuis l'article fondateur de Weiser [74]. Cependant, la technologie pour en faire une réalité n'est apparue que récemment. Les systèmes ubiquitaires Permettent d'automatiser certaines tâches quotidiennes grâce aux différents objets communicants disponibles [63], dotés de divers capteurs qui peuvent être portés sur le corps pour des applications de soins de santé, ou autre capteur utilisés pour surveiller l'environnement dans les zones de travail, à la maison, au bureau ou à l'extérieur, ou pour surveiller des machines ou des véhicules etc.

Les téléphones intelligents peuvent être utilisés pour la messagerie omniprésente, le courrier électronique et les communications vocales, mais ils comprennent également une variété d'outils de communication, mais ils intègrent aussi divers capteurs tels que des accéléromètres, des capteurs sonores et vidéo, et peuvent donc être utilisés comme des nœuds de détection omniprésents portés par des personnes.

1.4.4 Propriétés des systèmes ubiquitaires

Les besoins engendrés par l'intégration d'un nombre de plus en plus important d'objets communicants entraînent de nouveaux défis et propriétés indispensables [33] :

Distribution

Les systèmes distribués sont considérés comme le noyau des systèmes ubiquitaires, ils communiquent entre eux via des réseaux filaires ou sans fils en utilisant des infrastructures de communication permettant d'exploiter des services, les exécuter, et les échanger entre les différents dispositifs en cas de nécessité. Donc ils peuvent prendre en charge des nouveaux contextes et ressources sans que cela ne soit visible par un utilisateur [73].

Transparence

Les systèmes ubiquitaires nécessitent d'un minimum d'interventions des utilisateurs pour s'adapter seuls aux changements du contexte et à l'auto-apprentissage, il ne faut pas encombrer l'utilisateur avec des considérations qui ne le concernent pas [39].

Mobilité

La capacité d'accéder à des services ou à des applications sans tenir compte de la localisation physique, comportements et mouvements des utilisateurs [56], cela représente l'atout décisif dans les systèmes ubiquitaires.

Sensibilité aux Contexte

Les systèmes ubiquitaires utilisent la perception de l'environnement pour interagir plus naturellement avec l'utilisateur grâce aux capteurs de l'environnement physique, les capteurs logiques et la description des personnes sur les applications [73]. Ils signalent également l'importance de l'adaptation du système à ces changements [33]. Donc Les systèmes ubiquitaires sont des **systèmes sensibles aux contextes**.

Intelligence

Percevoir le contexte ne suffit pas, il faut utiliser efficacement les informations contextuelles pour les adapter au utilisateur. Exemple : Chambre d'hôtel intelligente, maison intelligente etc.

Pro-action

Proposer et préparer le traitement d'une demande utilisateur avant même que cette demande ne soit explicite [33] en fonction du contexte présent ou prédit. Exemple : Avec ces préférences choisir le menu dans un restaurant, ou choisir une activité.

Scalabilité

La mise à l'échelle permet de connaître les limitations du système lors d'une montée en charge du nombre de composants [46], il faut donc que les systèmes ubiquitaires soient capables de gérer un grand nombre d'équipements. Cela permet alors de développer des applications dont le cœur est indépendant du volume [3].

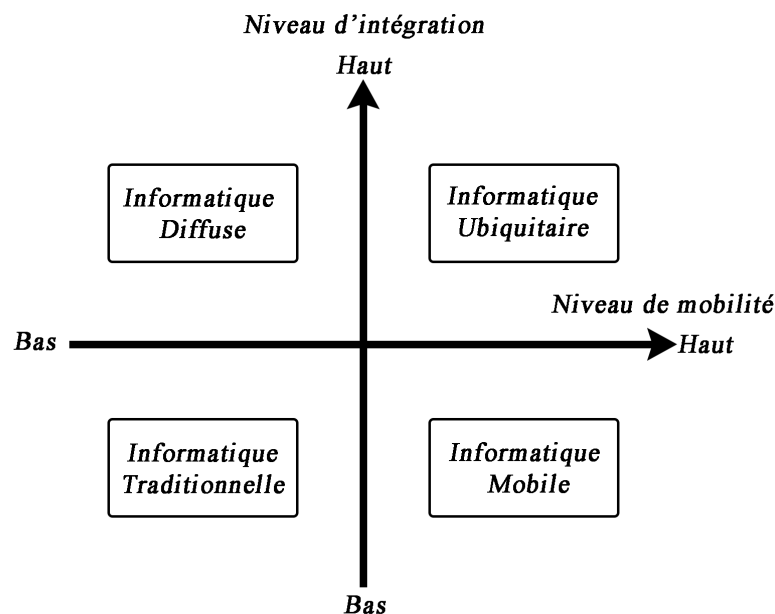


FIG. 1.4 : Comparaison de différents dispositifs informatiques [10]

1.4.5 Défis de l'informatique ubiquitaire

Le grand nombre d'équipements qui communiquent dans l'informatique ubiquitaire soulèvent de nombreux problèmes [33] :

Accessibilité

L'informatique ubiquitaire a été conçue dans le but d'être déployée partout et d'être accessible et maniables par la majorité des utilisateurs non spécialistes. Donc s'est un défi à relever pour les concepteurs.

Hétérogénéité

l'informatique ubiquitaire se caractérisent par la multiplicité d'objets du quotidien, étant aussi différents que nombreux, l'hétérogénéité se situe à plusieurs niveaux : domaines d'application, matériels, logiciels, protocoles réseaux et modèles d'interaction.

Limitation des réseaux

La connexion sans fil utilisée dans les équipements mobiles est souvent limitée en bande passante et instable par rapport à la connexion filaire, cela inclut le Bluetooth et le WiFi etc [6]. La vitesse lente de ces techniques par rapport au réseau filaire peut être une barrière pour exécuter de grandes applications et la gestion de données en temps réel.

La gestion du contexte

Les systèmes ubiquitaires introduisent la sensibilité au contexte en raison de la capacité des objets intelligents à le capturer. Le traitement et la gestion des informations contextuelles dans un environnement pervasif est un autre défi de taille.

1.5 Conclusion

Dans ce chapitre, nous avons présenté les principaux aspects du contexte et la sensibilité au contexte, y compris la catégorisation, l'acquisition et la modélisation du contexte. En plus des systèmes sensibles au contexte et leur architecture. La dernière partie est dédiée à l'informatique ubiquitaire, son environnement et les systèmes ubiquitaires qui sont un type de système sensible au contexte et leurs propriétés.

Dans le second chapitre, nous parlerons de l'Internet des objets, qui est le cœur de nos systèmes intelligents de capture du contexte.

Chapitre 2

Internet des objets

2.1 Introduction

Aujourd'hui, l'Internet est devenu omniprésent, a touché presque tous les coins du globe et affecte la vie humaine de manière invraisemblable. Nous entrons maintenant dans une ère de connectivité encore plus complète où une très grande variété d'appareils seront connectés au web [5]. En général, l'Internet des objets ou IoT est la mise en réseau d'objets physiques se connectant via l'Internet. L'IoT n'est pas une idée récente, car les appareils communiquent entre eux depuis plusieurs années. C'est un sujet émergent d'importance sociale, économique et technique.

L'IoT est un terme très descriptif d'une vision selon laquelle tout devrait être connecté à l'internet [55]. Il connecte différents objets et leur permet de partager des informations afin d'automatiser les processus pour les êtres humains et de rendre leur vie plus pratique.

Il y a plus de 14 milliards d'appareils numériques et électroniques interconnectés en fonctionnement dans le monde, l'équivalent de presque deux appareils pour chaque être humain sur terre. Les applications IoT ont été largement utilisées dans plusieurs domaines de la vie tels que la santé, l'automatisation industrielle et l'énergie etc.

Dans ce chapitre nous passerons en revues l'Internet des objets, et toutes les notions relatives au objets connectés et au Système IoT.

2.2 Internet des objets

Le terme Internet des objets a été inventé par Kevin Ashton dans sa présentation à Procter & Gamble (P&G) en 1999 [8]. Il le désigne comme un système où les objets sont connectés à Internet ou un réseau local.

Il existe plusieurs définitions, L'IEEE (Institut des ingénieurs électriciens et électroniciens) l'a défini comme un système constitué de réseaux d'actionneurs, de capteurs et d'objets intelligents dont le but est d'interconnecter "toutes" les choses, y compris les objets quotidiens et industriels, de manière à les rendre intelligentes, programmables et plus compétentes pour interagir avec les humains et entre elles [61].

L'UIT (Union Internationale des Télécommunications) le définit comme une infrastructure mondiale pour la société de l'information, qui permet de disposer de services évolués en interconnectant des objets grâce aux technologies de l'information et de la communication [57].

Une manière générique de décrire le processus de l'IoT était implicite dans la définition de l'informatique ubiquitaire de Mark Weiser [74].

Une définition globale est que Internet des objets est un paradigme dans lequel les objets équipés de capteurs, d'actionneurs et de processeurs communiquent entre eux pour servir un objectif utile [66].

2.3 Objets connectés

2.3.1 Définitions

Un objet connecté, dans le contexte de l’IoT, est une entité ou un objet physique doté d’un identifiant unique, d’un système intégré et de la capacité de transférer des données sur un réseau, ainsi que d’actionneurs permettant aux objets d’agir (par exemple, ouvrir ou fermer une porte, allumer ou éteindre la lumière, etc.) [75].

Une autre définition est que c’est un objet qui a une fonction mécanique et/ou électrique propre, il peut soit être conçu directement (connectable), soit il est déjà existant et la connectivité est rajoutée à posteriori [18].

2.3.2 Caractéristiques d’un objet connecté

Un objet connecté à trois caractéristiques principales [48] :

- La capacité à collecter les données par le biais de capteurs comme localisation, distance, température, etc.
- La capacité à transmettre, traiter l’information et enclencher une action si nécessaire.
- La capacité à s’interconnecter avec d’autres objets dans différents environnements.

2.3.3 Composants d’un objet connecté

Un objet connecté se compose de Trois éléments essentiels et d’une source d’énergie [18] :

Capteurs

Tous les objets connectés doivent être dotés d’un ou plusieurs capteurs pour collecter les données de l’environnement. Les capteurs sont des composants essentiels des objets intelligents. L’un des aspects les plus importants de l’Internet des objets est la connaissance du contexte, qui n’est pas possible sans la technologie des capteurs. Les capteurs de l’IoT sont principalement de petite taille, peu coûteux et consomment peu d’énergie, mais ils sont limités par des facteurs tels que la capacité de la batterie et la facilité de déploiement [66].

Les capteurs physiques dans l’IoT et l’électronique en générale sont divisés en trois types selon leurs sorties logiques, analogiques et numériques [70] :

- **Logique :**

C'est un capteur de type tout ou rien (TOR). Les capteurs délivrent donc un signal binaire (deux états possibles : 0 ou 1).

- **Analogique :**

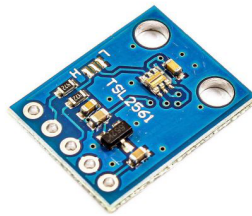
Ce capteur délivrent une valeur qui évolue dans le temps et qui peuvent prendre une infinité de valeurs, L'information à transmettre peut varier de manière continue, comme la température d'une pièce.

- **Numérique :**

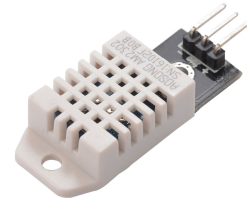
Ce capteur délivrent des informations qui peuvent être sous la forme d'un code numérique binaire (avec un nombre de bits définis (0 ou 1)). Une information qui ne prend qu'un nombre fini de valeurs est dite numérique.



(a) Capteur de mouvement
(Logique)



(b) Capteur de luminosité
(Analogique)



(c) Capteur de température
et d'humidité (Numérique)

FIG. 2.1 : Exemples de capteurs

Actionneurs

Un actionneur est un dispositif qui peut modifier l'environnement en convertissant l'énergie électrique en une forme d'énergie utile. Les éléments de chauffage ou de refroidissement, les haut-parleurs, les lumières, les écrans et les moteurs en sont des exemples [66].



(a) LED



(b) LCD



(c) Servomoteur

FIG. 2.2 : Exemples d'actionneurs

Module de connectivité

La connectivité de l'objet est assurée par un module qui va permettre la communication de l'objet vers un ou plusieurs réseaux. Les objets pourront d'une part remonter des informations telles que leur identité, leur état ou les données de capteurs, et d'autre part recevoir des informations telles que des données et/ou des commandes d'action [18].

Le WiFi, le Bluetooth, La connexion Ethernet ou GSM en sont des exemples.

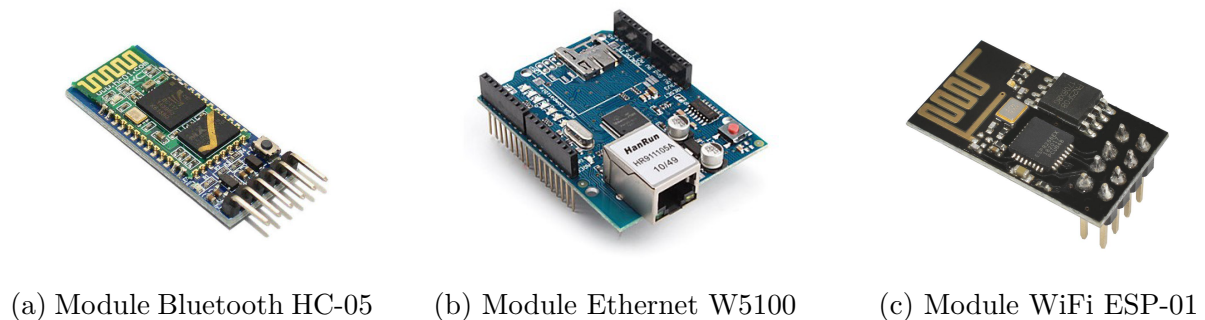


FIG. 2.3 : Exemples de Modules de connectivité

Sources d'énergie

Il existe plusieurs sources pour alimenter un objet connecté, on cite l'alimentation filaire pour les objets ayant accès à une prise de courant, les piles ou batteries pour ceux qui n'y ont pas accès ou de manière occasionnelle, les capteurs d'énergie par exemple le capteur solaire thermique [18].

Les objets passifs sans piles sont alimentés par les ondes électromagnétiques des lecteurs comme dans le cas du RFID ou du NFC.

2.4 Système IoT

2.4.1 Définition

Un système IoT est un système qui réunit de nombreux acteurs et composants technologiques. Il est composé d'objets connectés, de réseaux de communication sans fil et de plateformes de collecte, d'hébergement et/ou de traitement des données pour les utilisateurs finaux [18].

2.4.2 Composants d'un système IoT

Un système IoT se décompose en trois éléments [58] :

Objects

Il s'agit du matériel et des logiciels utilisés pour interagir avec le monde. Ils peuvent se connecter directement ou indirectement à Internet.

Passerelle

Le dispositif de passerelle IoT fournit une connexion et une traduction entre les appareils et le cloud. Il permet également aux appareils qui ne sont pas directement connectés à Internet d'utiliser les services du cloud.

Cloud

Le cloud est essentiel pour stocker, traiter et analyser les données de chaque appareil ou objet dans le système.

2.4.3 Protocoles de communication

Il existe plusieurs protocoles de communication pour le développement des systèmes d'IoT, quelques exemples :

CoAP

Le protocole d'application contraint (Constrained Application Protocol) est un protocole de couche d'application de demande/réponse synchrone qui a été conçu par l'Internet Engineering Task Force (IETF) pour cibler les appareils à ressources limitées. Il a été conçu en utilisant un sous-ensemble des méthodes HTTP, ce qui le rend interopérable avec HTTP [14].

CoAP fonctionne sur UDP pour que l'implémentation globale reste légère. Il utilise les commandes HTTP (GET, POST, PUT et DELETE) pour fournir des interactions orientées ressources dans une architecture client-serveur [32].

MQTT

Le Message Queue Telemetry Transport est un protocole de publication/abonnement qui fonctionne sur TCP. Il a été développé par L'International Business Machines Corporation (IBM) principalement comme un protocole client/serveur. Les clients sont des éditeurs/abonnés et le serveur agit comme un courtier auquel les clients se connectent via TCP [66].

API REST

Le Representational State Transfer n'est pas vraiment un protocole mais un style architectural. Il a été introduit par Roy Fielding en 2000 [24], et il est largement utilisé

depuis.

REST utilise les méthodes HTTP (GET, POST, PUT et DELETE) pour fournir un système de messagerie orienté ressources dans lequel toutes les actions peuvent être effectuées en utilisant simplement les commandes HTTP synchrone (demande/réponse). Il utilise l'en-tête Accept intégré de HTTP pour indiquer le format des données qu'il contient. Le type de contenu peut être XML ou JSON (JavaScript Object Notation) et dépend du serveur HTTP et de sa configuration [32].

REST est déjà un élément important de l'IoT car il est pris en charge par toutes les plates-formes commerciales de cloud M2M (Machine to Machine), exemple la base de données cloud de Google **Firestore**.

2.5 Modèles de communication d'Internet des objets

Les modèles de communication de l'Internet des objets sont les suivants [1] :

2.5.1 Périphérique à périphérique

Connexion et communication directe entre deux périphériques ou plus, plutôt que via un serveur intermédiaire. Ce modèle utilise de petits paquets d'informations pour la communication entre des appareils avec un débit de données relativement minimum comme dans la domotique.

2.5.2 Périphérique à cloud

Le Périphérique est directement relié à un service cloud, comme un fournisseur de services d'application, pour échanger des données et contrôler le trafic des messages. Cette méthode consiste à établir un lien entre le dispositif et le réseau IP, qui se connecte finalement au service cloud.

2.5.3 Périphérique à passerelle

La passerelle agit comme un intermédiaire entre le Périphérique et le service cloud, le modèle de passerelle entre le dispositif et la couche d'application (Application Layer Gateway ou ALG) se connectent par l'intermédiaire d'un service ALG en tant que canal pour étendre un service cloud.

2.5.4 Partage des données en amont (Back-End)

Il s'agit d'une architecture de communication qui permet aux utilisateurs d'exporter et d'analyser les données des objets intelligents à partir d'un service cloud, en combinaison avec des données provenant d'autres sources. Il permet également d'agréger et d'analyser les données recueillies à partir de flux de données de Périphérique individuels. Plus

concrètement, ce modèle facilitera l'échange d'informations de serveur à serveur (S2S) [36].

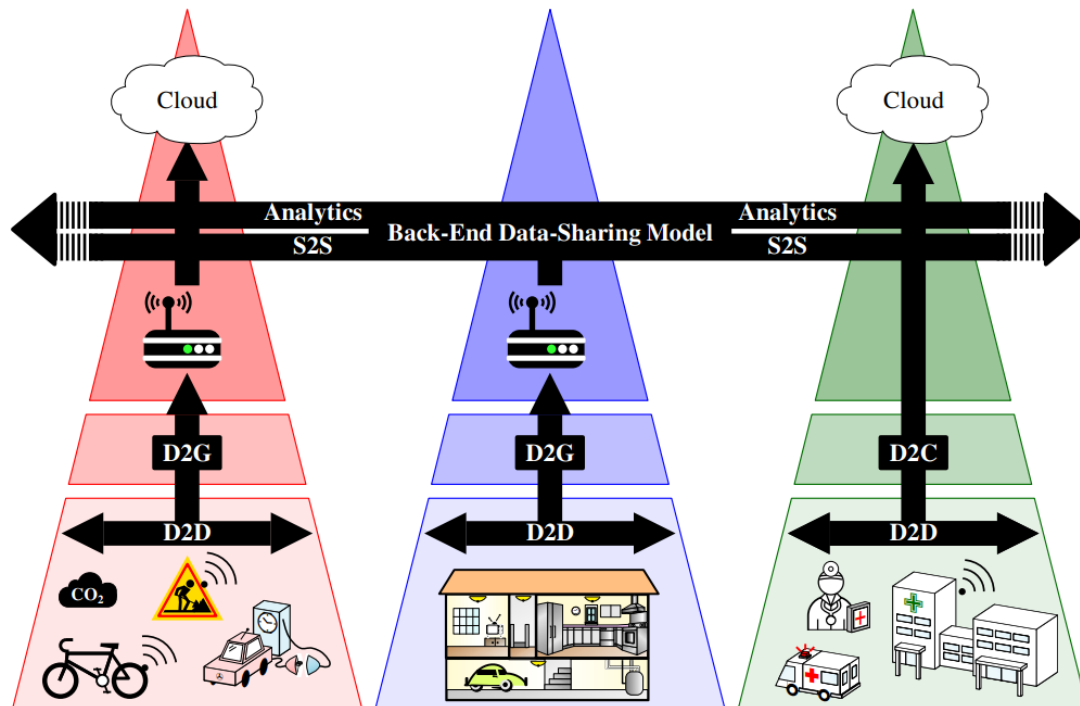


FIG. 2.4 : Modèles de communication de l'Internet des objets [36]

D2D : Périphérique à périphérique (Device to Device).

D2C : Périphérique à cloud (Device to Cloud).

D2G : Périphérique à passerelle (Device to Gateway).

S2S : Partage des données en amont (Server to Server).

2.6 Domaines d'application

Il existe une multitude de domaines d'application de l'IoT, On cite quelqu'un [54] :

2.6.1 Smart Home

L'Internet des objets est un terme qui pourrait être utilisé pour décrire tout ce qui est connecté à l'Internet. L'IoT est la prochaine étape de l'évolution des réseaux sans fil et des appareils connectés [29], à mesure que la taille des capteurs diminue et qu'ils migrent de nos smartphones vers d'autres objets du quotidien.

Une maison intelligente est une instance de ce paradigme, héritant de tous les aspects de la connectivité des appareils concernés. Une Smart Home est une résidence équipée de capteurs, de systèmes et de dispositifs qui peuvent être consultés, contrôlés et surveillés à distance, généralement via Internet. Elle peut intégrer divers services domestiques intelligents afin de fournir un environnement pratique, et de les aider à accomplir efficacement

leurs tâches ménagères. En général, les systèmes de maison connectée intelligente peuvent être classés en quatre catégories : systèmes de santé, sécurité, énergie et de divertissement.

Les services de soins et de santé se concentrent sur la fourniture de soins de santé mobiles et d'une aide à la remise en forme, et vise à offrir une vie saine et indépendante. Par rapport aux autres domaines, le service de soins de santé implique également l'utilisation de capteurs portables afin de permettre une surveillance éventuellement continue des paramètres des signaux corporels, même en dehors du domicile.

Les services de sécurité sont conçus pour surveiller, découvrir et contrôler les menaces à la sécurité et à la sûreté. Les systèmes énergétiques visent à assurer une consommation et une gestion efficaces de l'énergie dans la maison. Les systèmes de divertissement tendent à promouvoir le divertissement en maximisant le confort.



FIG. 2.5 : Smart Home [17]

2.6.2 Santé

L'Internet des objets de santé, IoHT en abrégé (H pour Healthcare), est un concept qui décrit des dispositifs identifiables de manière unique, connectés à l'Internet et capables d'interagir les uns avec les autres, utilisés dans le domaine médical.

Des exemples de l'utilisation de L'IoT dans ce domaine sont nombreux. Le suivi en temps réel via des dispositifs connectés peut sauver des vies en cas d'urgence médicale comme une crise d'asthme, une insuffisance cardiaque, un diabète, etc. L'IoT peut automatiser le flux de travail des soins aux malades avec l'assistance, la solution de mobilité des soins. Il permet aux appareils de recueillir des données importantes et de les transférer aux médecins pour un suivi en temps réel, tout en envoyant des notifications aux personnes sur les parties critiques via des applications mobiles et d'autres appareils liés.

2.7 Conclusion

Dans ce chapitre, nous avons abordé les concepts clés de l'internet des objets. Nous avons présenté les objets connectés et les systèmes IoT, leurs composants ainsi que leur architecture et leurs caractéristiques. On a aussi présenté des exemples de domaines d'applications de l'IoT.

Dans le chapitre suivant, nous parlerons des applications et des systèmes d'exploitation Mobiles.

Chapitre 3

Applications et Systèmes d'exploitation Mobiles

3.1 Introduction

Le développement de la technique mobile a rendu les appareils mobiles tels que les téléphones mobiles et les tablettes de plus en plus populaires ces dernières années. Les appareils mobiles sont devenus l'un des appareils électroniques les plus importants dans le monde d'aujourd'hui.

En outre, le développement de la capacité des appareils mobiles, qui étaient à l'origine utilisés pour appeler ou envoyer des SMS uniquement, a permis aux fournisseurs de développer de nouvelles fonctionnalités et des applications qui ont été ajoutées pour répondre à des besoins dans de nombreux domaines et activités telles que la recherche de données sur Internet, les opérations bancaires, la réservation, l'achat de billets ou le shopping.

Les appareils mobiles sont aussi de plus en plus fréquents à cause des divers systèmes d'exploitation mobiles tels que Android, IOS ou bien Windows Phone.

Dans ce chapitre on va définir ce qu'est l'informatique mobile, quelques concepts de base à propos des applications mobiles, le système Android, son architecture et le développement d'applications Android.

3.2 Applications mobiles

3.2.1 Définition

Les applications mobiles consistent en un logiciel ou ensemble de programmes qui s'exécutent sur un appareil mobile et effectuent certaines tâches pour l'utilisateur. L'application mobile est un nouveau segment de la technique globale de l'information et de la communication, qui a un large éventail d'utilisation pour son vaste domaine de fonctionnement comme les appels, la messagerie, la navigation, le chat, le réseau social, la communication, audio, vidéo, jeux, etc [30].

3.2.2 Types d'applications mobiles

Les applications mobiles sont devenues extrêmement populaires, aussi bien chez les consommateurs que dans les entreprises. Il existe trois types d'applications mobiles [45] :

Applications Natives

Les applications natives vivent sur l'appareil et sont accessibles par des icônes sur l'écran d'accueil. Elles sont développées spécifiquement pour un système d'exploitation donné et peuvent tirer pleinement parti de toutes les fonctionnalités de l'appareil. Les applications natives sont installées par le biais d'un magasin d'application (tel que Google Play ou l'App Store d'Apple).

Pour développer une application native, il faut utiliser la mémoire smartphone sans négliger les options liées au système d'exploitation concerné. De cette façon, le résultat se résume dans l'aboutissement d'applications mobiles avec des fonctions plus professionnelles.

Applications Web

Les applications Web ne sont pas de véritables applications, ce sont en fait des sites Web qui ressemblent à des applications natives, mais qui ne sont pas mises en œuvre en tant que telles. Ceux-ci sont exécutés par un navigateur et sont généralement rédigés en HTML5.

Une connexion Internet est essentielle pour utiliser ce type d'application. Il est donc possible d'accéder à autant de fonctions que possible sur un téléphone mobile, telles que la géolocalisation, l'accéléromètre ou la gestion des fonctions multitâches.

Applications hybride

Les applications hybrides sont à la fois des applications natives et des applications Web. Comme les applications natives, elles sont disponibles dans une boutique d'applications et peuvent tirer parti des nombreuses fonctionnalités disponibles sur les appareils. Et comme les applications web, elles reposent sur le HTML qui est dans un navigateur, à la différence que le dernier est intégré à l'application.

3.2.3 Domaines d'application mobile

Plusieurs secteurs d'application sont concernés par des applications mobiles. Parmi ces principaux secteurs sont la sécurité, les transports, le tourisme, l'environnement et les infrastructures, ainsi que les services publics, Etc [69].

Voici quelques exemples d'applications mobiles dans le domaine de l'hospitalité et l'hôtellerie :

Hilton Honors

En 2016, la marque hôtelière Hilton a été évaluée à 7,82 millions de dollars américains. La chaîne hôtelière classée AAA possède sa propre application mobile unique **The Hilton Honors** avec plus d'un million de téléchargements sur Google Play.

L'application permet aux utilisateurs de rechercher et de réserver une chambre dans l'un des hôtels Hilton ou de s'y faire conduire par Uber. Grâce à cette application, il est possible de demander des extras tels que de la nourriture, des boissons, des oreillers supplémentaires, etc.

L'application indique aux visiteurs le temps qu'il fait, et une carte GPS pour se rendre à l'hôtel. Elle permet également de contacter le personnel du Hilton.

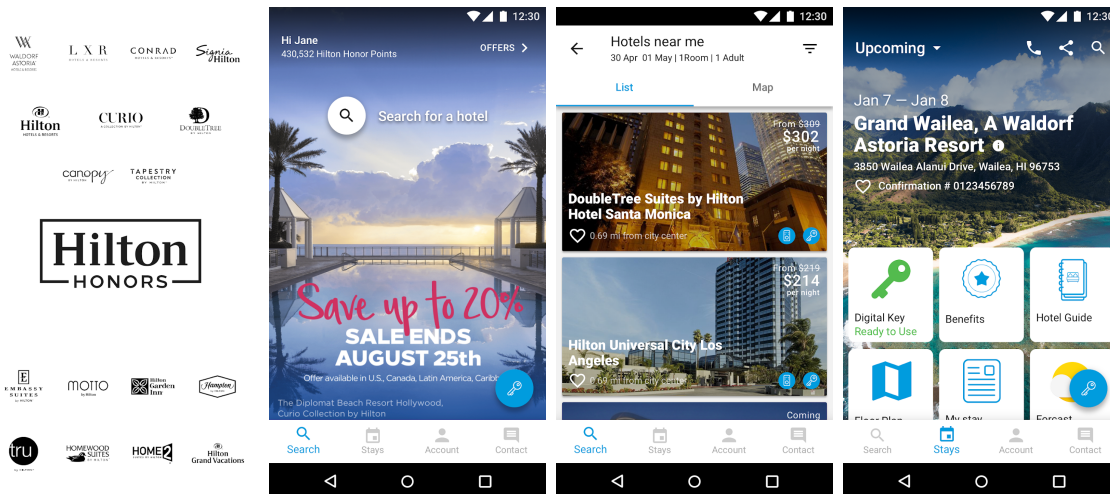


FIG. 3.1 : IHM application mobile **The Hilton Honors**

Marriott Bonvoy

En 2016, la marque Marriott International était estimée à 5,32 millions de dollars américains. Leur application principale Marriott International est considérée comme un service innovant et reçoit souvent des prix. Aujourd'hui, cette application compte plus d'un million de téléchargements sur Google Play.

L'application permet à ses utilisateurs non seulement de réserver une chambre à l'avance, mais aussi de gérer tous leurs déplacements à l'aide d'un appareil mobile. Les clients peuvent contacter directement un hôtel Marriott à partir de l'application ou choisir les éléments nécessaires dans la liste proposée. Avec son aide, les clients peuvent également consulter des cartes, des guides de ville et des équipements.



FIG. 3.2 : IHM application mobile **Marriott International**

3.2.4 Avantages et inconvénients d'une application mobile

Les application mobile n'ont pas que des avantage, sur tout du côté de leur conception et développement [43] :

Avantages

- Les applications mobiles offrent une solution beaucoup plus rapide que la navigation sur le Web mobile. Pour naviguer sur le Web, l'utilisateur doit ouvrir un navigateur, saisir une adresse URL et attendre que la page Web se charge. L'ouverture d'une application mobile ne prend qu'un moment, puisque la plupart des informations sont enregistrées dans l'application elle-même.
- Les applications réduisent les frais des SMS et des dépliants papier. Ils simplifient les échanges en informant les consommateurs en toute sécurité, immédiatement et directement.
- Les applications réduisent la charge de travail du personnel pour les demandes de données et les appels téléphoniques.

Inconvénients

- Le langage de programmation est l'un des inconvénients du développement d'applications mobiles la plupart du temps, les ingénieurs utilisent le langage de programmation Java pour composer des applications pour les appareils Android. Bien que de nombreux ingénieurs se concentrent sur la maîtrise de Java, cela ne sert qu'à travailler avec Android. IOS a connu des problèmes constants avec Java, ce qui entraîne des limitations indésirables pour les ingénieurs d'applications.
- Le développement pour deux ou plusieurs plates-formes à la fois est un défis, les ingénieurs d'applications mobiles doivent créer leurs applications dans un autre langage local pour chaque appareil.
- L'utilisation d'informations personnelles lors de l'utilisation d'applications mobiles malgré le fait que l'amélioration des applications Android et IOS simplifie des choses comme la banque en ligne, les recherches locales et les itinéraires, l'utilisation de données individuelles peut sembler très envahissante. La plupart des applications nécessitent l'accord de l'utilisateur pour accéder à l'information de son appareil. Dans tous les cas, si vous choisissez de ne pas permettre à l'application d'accéder à ces informations, il est courant que vous ne pouvez pas installer l'application sur votre appareil ou avoir des limitations dans les fonctionnalités.

3.3 Systèmes d'exploitation mobiles

3.3.1 Définition

Un système d'exploitation mobile est un logiciel système spécialement conçu pour fonctionner sur des appareils portables tels que les téléphones mobiles et les PDA. Il s'agit d'une plateforme où les programmes d'application fonctionnent sur des appareils mobiles, chaque système d'exploitation suit sa propre architecture [37].

3.3.2 Types des systèmes d'exploitation mobiles

Nous allons brièvement introduire quelques systèmes d'exploitation mobiles [37, 41] :

IOS

IOS (anciennement appelé iPhone OS) est un produit d'Apple Inc. Il dispose de la deuxième plus grande base installée au monde sur les smartphones. Il est un logiciel non libre (closed source) et exclusif, et s'appuie sur le système d'exploitation open source Darwin.

L'iPhone, iPod Touch, iPad et Apple TV seconde et troisième génération utilisent tous IOS, qui est un dérivé de macOS qui a quatre couches d'abstraction : Core OS , Core Services, Media et une couche Cocoa.



FIG. 3.3 : Logo Systèmes d'exploitation IOS

Android

Android (basé sur le noyau Linux) est un système d'exploitation mobile développé par Google Inc. En plus d'avoir la plus grande base installée au monde sur les smartphones, c'est aussi le système d'exploitation le plus populaire pour les ordinateurs à usage général (une catégorie qui comprend les ordinateurs de bureau et les appareils mobiles).

Tandis que le système d'exploitation Android est gratuit et open source, dans les appareils vendus, la majorité des logiciels fournis (y compris les applications Google et les logiciels installés par les fournisseurs) sont des logiciels exclusifs et des sources fermées.



FIG. 3.4 : Logo Systèmes d'exploitation Android

Windows 10 Mobile

Windows 10 Mobile (précédemment nommé Windows Phone) est un produit Microsoft. Il est un logiciel non libre (closed source) et exclusif. Il a la troisième plus grande base installée sur smartphone derrière Android et IOS. Ce système d'exploitation dispose d'une interface utilisateur inspirée par Metro Design Language de Microsoft.

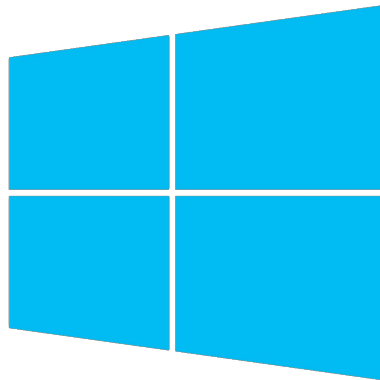


FIG. 3.5 : Logo Systèmes d'exploitation Windows 10 Mobile

Tizen

Tizen est un système d'exploitation pour des appareils tels que les smartphones, les tablettes, l'infodivertissement embarqué (In-vehicle infotainment ou IVI) , smart TVs.

C'est un système open source (le SDK était cependant fermé et exclusif) qui vise à fournir une expérience utilisateur cohérente sur tous les appareils.

Les éléments les plus importants de Tizen sont le noyau Linux et WebKit runtime. Actuellement, Tizen est le 4ème système d'exploitation mobile en termes de part de marché.



FIG. 3.6 : Logo Systèmes d'exploitation Tizen

Sailfish OS

Sailfish OS est un système d'exploitation Open source de la société Jolla, basé sur le noyau Linux. Ce système d'exploitation, en raison du modèle commercial de Jolla, de ses alliances avec divers partenaires et de la conception intentionnelle des composants internes du système d'exploitation, est capable d'adopter dans plusieurs couches des logiciels tiers, y compris ceux de Jolla.



SAILFISH OS

FIG. 3.7 : Logo Systèmes d'exploitation Sailfish OS

3.4 Système d'exploitation Android

3.4.1 Histoire du système d'exploitation Android

La plateforme Android open source a été initialement développée par Android Inc, à Palo Alto, Californie, Etats-Unis, en 2003. Plus tard, en 2005, Android a été vendu à Google. En 2007, Google en collaboration avec OHA (Open handset alliance), a lancé l'AOSP (Android Open-Source plate-forme) [59, 41].

L'objectif de l'OHA est de développer une norme ouverte pour les appareils mobiles, d'innover dans le domaine de la téléphonie mobile et de fournir une meilleure expérience au consommateur à moindre coût.

Android possède son propre système d'exploitation qui est basé sur le noyau Linux 2.6, son intergiciel, ses applications mobiles clés et ses applications, et le système d'exploitation Android possède sa propre machine virtuelle appelée DVM, qui est utilisée pour exécuter les applications Android.

3.4.2 Versions d'Android

En 2003, le développement du système d'exploitation Android a été lancé par Android Inc. Il est mis à jour sur une base régulière. La première version commerciale d'Android 1.0 est sortie en septembre 2008.

Depuis la sortie initiale, Android a été développé en permanence par Google et l'Open Handset Alliance. Les mises à jour du système d'exploitation Android se concentrent sur les nouvelles fonctions et les correctifs de bugs.

Le tableau montrent comment la version Android a évolué. Il indique aussi que les versions de base 1.0 et 1.1 ne possèdent pas de nom de code et que les versions postérieures d'Android ont un nom de dessert, et que la version 9 pie arrête le système de nom de code dessert après Android et commence un système appelé Android 10, Android 11, Android 12 [41].

Num	Version	Nom	Date De Sortie
1	1.0	Pas de Nom	September 23, 2008
2	1.1	Pas de Nom	February 9, 2009
3	1.5	Cupcake	April 27, 2009
4	1.6	Donut	September 15, 2009
5	2.0-2.1	Eclair	October 27, 2009
6	2.2-2.2.3	Froyo	May 20, 2010
7	2.3-2.3.7	GingerBread	December 6, 2010
8	3.0-3.2.6	Honeycomb	February 22, 2011
9	4.0-4.04	Ice Cream Sandwich	October 18, 2011
10	4.1-4.3.1	Jelly Bean	July 9, 2012
11	4.4-4.4.4	KitKat	October 31, 2013

12	5.0-5.1.1	Lollipop	November 4, 2014
13	6.0-6.0.1	Marshmallow	October 2, 2015
14	7.0-7.1.2	Nougat	August 22, 2016
15	8.0-8.1	Oreo	August 21, 2017
16	9	Pie	August 6, 2018
17	10	Android 10	September 3, 2019
18	11	Android 11	September 8, 2020
19	12	Android 12	October 4, 2021

TAB. 3.2 : Versions du Système d'exploitation Android

3.4.3 Architecture Android

Le système d'exploitation Android contient une pile de couches de composants logiciels. Les principales couches d'éléments Android sont [59] :

Couche Linux Kernel

La couche inférieure du système d'exploitation Android est la couche du noyau Linux. Cette couche n'interagit pas directement avec le développeur, elle contient de nombreux pilotes de périphériques matériels et est responsable de la gestion de l'énergie, de la mémoire, des périphériques et de l'accès aux ressources.

Couche Libraries et Android Runtime

En haut du noyau Linux se situe cette couche qui est divisé en deux éléments principaux : Android Runtime et Android Library.

Les bibliothèques natives contiennent WebKit, Open Gl, FreeType, SQLite, Media, runtime library C (libc), etc. Les bibliothèques sont écrites en C ou C++. Pour afficher le HTML sur le moteur du navigateur, on utilise la bibliothèque WebKit. Pour afficher du contenu graphique 2D ou 3D à l'écran, on utilise Open Gl. Surface Manager est utilisée pour gérer l'affichage du dispositif. SQLite est la base de données relationnelle utilisée pour le stockage des données et disponible pour toutes les applications.

Couche Application Framework

La couche Android Framework existe au-dessus de la couche library native et du runtime Android. Cette couche contient les API Android telles que l'interface utilisateur (UI), la téléphonie, les ressources, les emplacements, les fournisseurs de contenu (données) et les gestionnaires de paquets.

Cette couche comprend aussi un ensemble de classes et d'interfaces pour le développement d'applications Android.

Couche Application

Les applications se trouvent en dessus de la couche framework. Toutes les applications, par exemple l'accueil, les contacts, les paramètres, les jeux, les navigateurs, sont écrites en langage de programmation Java et utilisent un framework qui utilise les librairies et le runtime. L'utilisateur de l'appareil Android interagit directement avec cette couche.

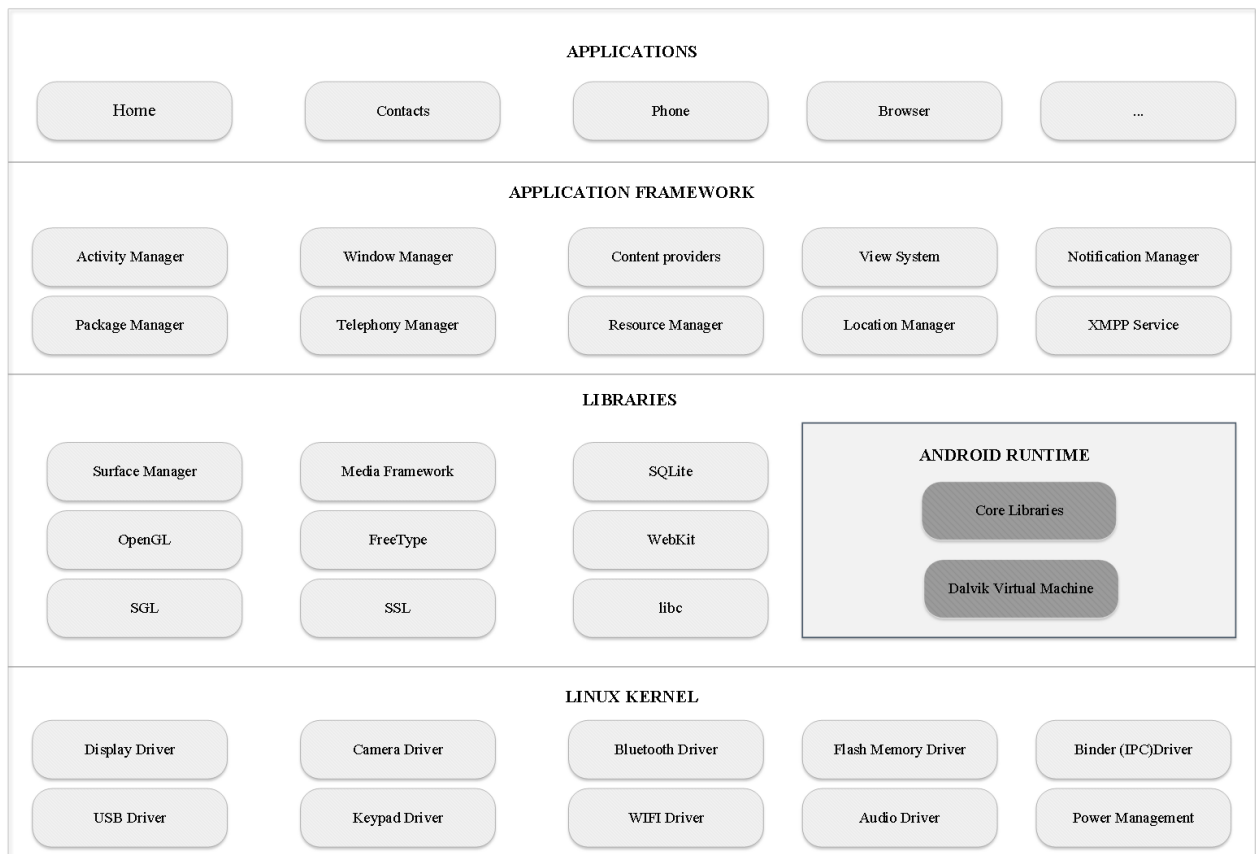


FIG. 3.8 : Architecture des couches Android

3.5 Développement d'applications Android

Les étapes de développement d'applications mobiles peuvent varier en fonction du projet ou des techniques établies. Les applications mobiles sont le résultat de plusieurs activités réalisées, telles que l'attribution des rôles au groupe de travail, la définition des objectifs et des activités, la mise en œuvre d'un ensemble de bonnes pratiques pour le travail d'équipe et la collaboration, l'établissement du calendrier des activités, la gestion des risques, entre autres. En fin de compte, les directives doivent être adaptées en fonction des ressources disponibles et des exigences du client [28].

3.5.1 Composants Principaux d'Android

Les composants d'application sont les éléments essentiels d'une application Android. Chaque composant est un point différent par lequel le système peut entrer dans l'application. Tous les composants ne sont pas des points d'entrée réels pour l'utilisateur, et certains dépendent les uns des autres, mais chacun existe en tant qu'entité propre et joue un rôle spécifique. Chacun d'eux est un seul bloc de construction qui permet de définir le comportement général de l'application. Les sous-sections ci-dessous représentent quatre types de composantes d'application : les activités, les services, les fournisseurs de contenu et les intentions [60].

Activités

Une activité représente un écran unique avec une interface utilisateur. Par exemple, une application d'email peut avoir une activité pour rédiger un email, et une autre activité pour lire des emails. Bien que les activités collaborent afin de créer une expérience utilisateur cohérente dans l'application de messagerie, chacune est indépendante des autres. Ainsi, une autre application peut lancer n'importe laquelle de ces activités (si l'application de messagerie l'autorise).

Services

Un service est un composant qui s'exécute en arrière-plan pour effectuer des opérations de longue durée ou pour effectuer des travaux pour des processus distants. Le service ne fournit aucune interface utilisateur. Un autre composant, tel qu'une activité peut lancer le service et le laisser fonctionner ou s'y lier afin d'interagir avec lui.

Content Providers

Un Content Providers gère un ensemble partagé de données d'application. Vous pouvez stocker les données dans le système de fichiers, une base de données SQLite, sur le Web ou n'importe quel autre emplacement de stockage permanent auquel votre application peut accéder. Grâce au Content Providers, d'autres applications peuvent interroger, voire modifier les données (si le Content Providers le permet).

Intents

Intent ou intentions est un mécanisme permettant de décrire une action spécifique, telle que "prendre une photo" ou "appeler à la maison". Dans Android, presque tout passe par des Intents, ce qui vous donne de nombreuses occasions de remplacer ou de réutiliser des composants. Par exemple, on a Intent pour envoyer un e-mail. Si votre application doit envoyer du courrier, vous pouvez faire appel à cette Intent.

3.5.2 Outils de développement d'application Android

Les outils pour construire, administrer, tester, lancer et mettre à jour une application mobile sont principalement [19] :

Android SDK

L'Android Software Development Kit (Android SDK) contient les outils nécessaires pour la création, la compilation et le déploiement des applications Android.

Débogueur ADB

Le SDK Android contient un débogueur nommé **Android debug bridge** ou également ADB, qui permet de connecter un appareil Android virtuel ou réel, dans le but de gérer l'appareil ou déboguer l'application.

IDE Android Developer Tools et Android Studio

Google propose deux environnements de développement intégrés (IDE) pour développer de nouvelles applications :

- **Eclipse**

Éclipse IDE est un environnement de développement intégré libre (le terme Eclipse désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, il est principalement écrit en Java, grâce à des bibliothèques spécifiques, il est également utilisé pour écrire des extensions.

- **Android Studio**

Android Studio est l'environnement de développement intégré (IDE) officiel pour le développement de la plateforme Android. Cet environnement est libre et disponible gratuitement sous la licence Apache.

Android Studio était en phase de preview en accès anticipé à partir de la version 0.1 en mai 2013, puis est entré en phase bêta à partir de la version 0.8 qui a été publiée en juin 2014. La première version stable a été lancée en décembre 2014, depuis la version 1.0. Basé sur le logiciel IntelliJ IDEA de Jet Brains.

Android Studio est conçu spécifiquement pour le développement Android. Il est disponible pour Windows et Mac.OS X et Linux, et a remplacé Eclipse Android Development Tools (ADT) en tant qu'IDE principal de Google pour le développement d'applications natives Android [68].

3.6 Conclusion

Dans ce chapitre, nous avons présenté les principaux concepts des applications mobiles, leurs systèmes d'exploitation les plus populaires, certains de leurs avantages et désavantages et quelques exemples d'applications d'hôtellerie. On a aussi présenté Android en détail ainsi que ses composants et outils de Développement.

Le chapitre suivant fera l'objet de la conception et les étapes à suivre pour réaliser notre système.

Chapitre 4

Analyse et Conception

4.1 Introduction

Afin d'aboutir à une meilleure organisation et une bonne maîtrise du travail et ainsi être capable de déployer de meilleures applications et systèmes, il est nécessaire de suivre une approche méthodologique stricte. C'est pourquoi nous avons choisi le langage de modélisation UML.

Dans ce chapitre nous allons procéder à l'analyse et la conception de notre système, qui se compose d'une application mobile pour l'hôtel, une application pour la gestion de l'hôtel et un système pour la capture des informations contextuelles (une chambre intelligente).

4.2 Processus unifié

Le processus unifié est un processus de développement logiciel qui nous permet de transformer les besoins des utilisateurs en logiciels. C'est un standard industriel des auteurs de l'UML. Il est connu sous le nom de "Unified Process" ou UP.

Le projet UML était de fournir à la fois un langage visuel "UML" et un processus d'ingénierie logicielle "UP".

UP est basée sur le travail de processus effectué à Ericsson (l'approche d'Ericsson 1967), Rational (le processus objectif rationnel 1996 à 1997) et d'autres sources de meilleures pratiques. Il s'agit donc d'une méthode pragmatique et éprouvée de développement de logiciels qui intègre les meilleures pratiques de ses prédécesseurs [7]. Il comporte quatre phases [C4] :

4.2.1 Inception

La première étape sert à la préparation de la base du projet, y compris la préparation de l'analyse de rentabilité, la définition de la portée et des limites du projet, la description des exigences clés et la solution d'architecture potentielle. Ainsi que les compromis de conception, la détermination des risques et l'élaboration du calendrier initial du plan de projet avec les principales étapes et les estimations de coûts.

4.2.2 Elaboration

Dans cette deuxième phase, il faut saisir la majorité des exigences du système (par exemple, sous la forme de cas de modélisation UML topologiques), effectuer une analyse des risques identifiés et élaborer un plan de gestion des risques afin de réduire ou d'éliminer leur impact sur le calendrier et le produit final, établir la conception et l'architecture (par-exemple, à l'aide de diagrammes de classe de base, de diagrammes de paquets et de diagrammes de déploiement), et établir un plan (calendrier, estimation des coûts et objectifs atteignables) pour la prochaine phase.

4.2.3 Construction

C'est la phase la plus longue et la plus importante du processus unifié. Au cours de cette phase, la conception du système est finalisée et affinée et le système est construit à l'aide de la base créée au cours de la phase précédente. La phase de construction se compose de plusieurs itérations, chaque itération devant produire une version exécutable du système. La dernière itération de cette phase produit un système complet à déployer au cours de la phase de transition.

4.2.4 Transition

La phase finale du projet qui permet la prestation du nouveau système aux utilisateurs finaux. La phase de transition comprend aussi la migration des données des anciens systèmes ainsi que la formation des utilisateurs.

4.3 Langage de modélisation UML

UML est un langage graphique et textuel pour préciser, visualiser, construire et documenter les artefacts dans les systèmes logiciels. Il permet aux développeurs et aux clients de mieux comprendre le logiciel, le système ou le produit à développer.

La modélisation UML est basée sur deux phases : l'analyse des besoins et la conception. Lors de la phase de spécification des besoins, nous soulignerons les besoins et les acteurs concernés par le système. A l'étape de la conception, nous fournirons une série de diagrammes importants comme, le diagramme d'activité, le diagramme de classes, le diagramme de séquence, le diagramme de collaboration ou le diagramme de communication, le diagramme de machine à états, le diagramme de composants et le diagramme de déploiement. Ces diagrammes traduiront la solution et fournirons des descriptions détaillées des résultats de l'analyse [72].

4.4 Analyse

L'analyse commence avec la spécification des besoins fonctionnels du système, l'identification des acteurs et de leurs différentes tâches.

4.4.1 Identification des acteurs

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) en interaction directe avec le système étudié. Un acteur peut consulter ou changer directement l'état du système, en transmettant ou en recevant des messages qui peuvent transporter des données [62].

Les acteurs qui participeront à notre système sont :

Visiteur

Désigne la personne qui utilise l'application mobile sans avoir de compte, ou qu'il ne s'est authentifié, il a le droit de se connecter ou de s'inscrire (s'authentifier).

Client

Désigne la personne qui a un compte dans l'application et qui s'est authentifié.

Résident

Désigne la personne qui a une réservation et qui a fait le check-in de l'hôtel, il a scanné un QR-code donné par le réceptionniste, et donc peut accéder à toutes les fonctionnalités de l'application mobile.

Réceptionniste

Désigne la personne qui réceptionne les résidents, il est donc responsable de la gestion des réservations, des check-in et des check-out.

Gérant de l'hôtel

Désigne la personne responsable de la gestion de l'hôtel et de son bon fonctionnement.

Base de données

Désigne notre base de données, C'est l'élément le plus important vu qu'il constitue non seulement notre base, mais aussi le moyen de communication entre les différents éléments de notre système.

On a opté pour une base de données Cloud, vu le besoin d'utiliser une base de données en temps réel pour le bon fonctionnement de notre système, et la bonne gestion des informations contextuelles captées.

Cet acteur interagit avec la majorité de nos cas d'utilisation, on a donc décidé de ne pas l'inclure dans le diagramme de cas d'utilisation, pour une clarté visuelle, et le considérer comme un élément interne du système.

4.4.2 Diagramme de cas d'utilisation

Un cas d'utilisation modélise un service rendu par le système. Celle-ci exprime les interactions acteur/système et apporte une valeur ajoutée « notable » à l'acteur concerné [62].

Ce diagramme vise à déterminer les exigences fonctionnelles du système à l'étude. Voici le diagramme de cas d'utilisation de notre application :

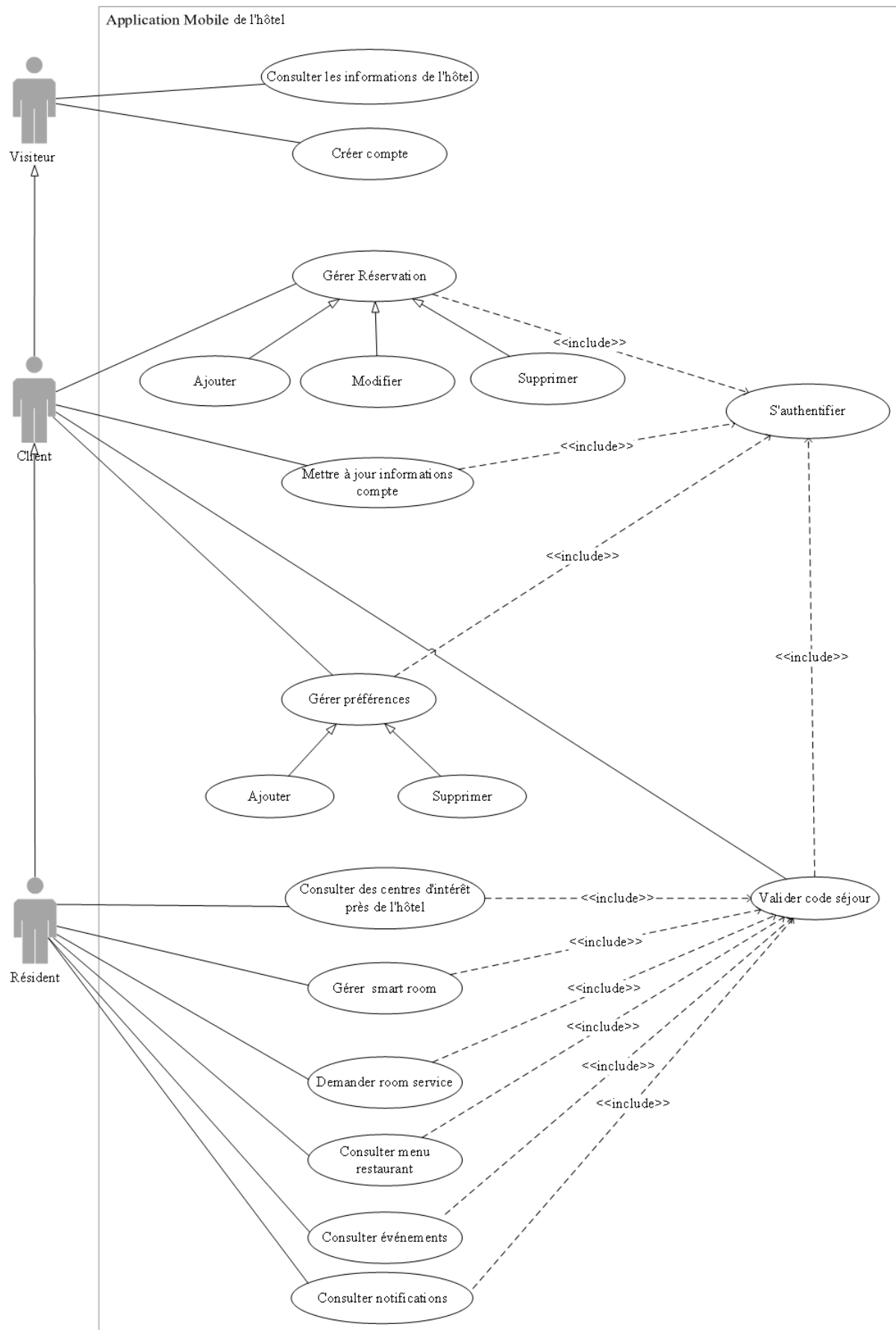


FIG. 4.1 : Diagramme de cas d'utilisation de l'application mobile

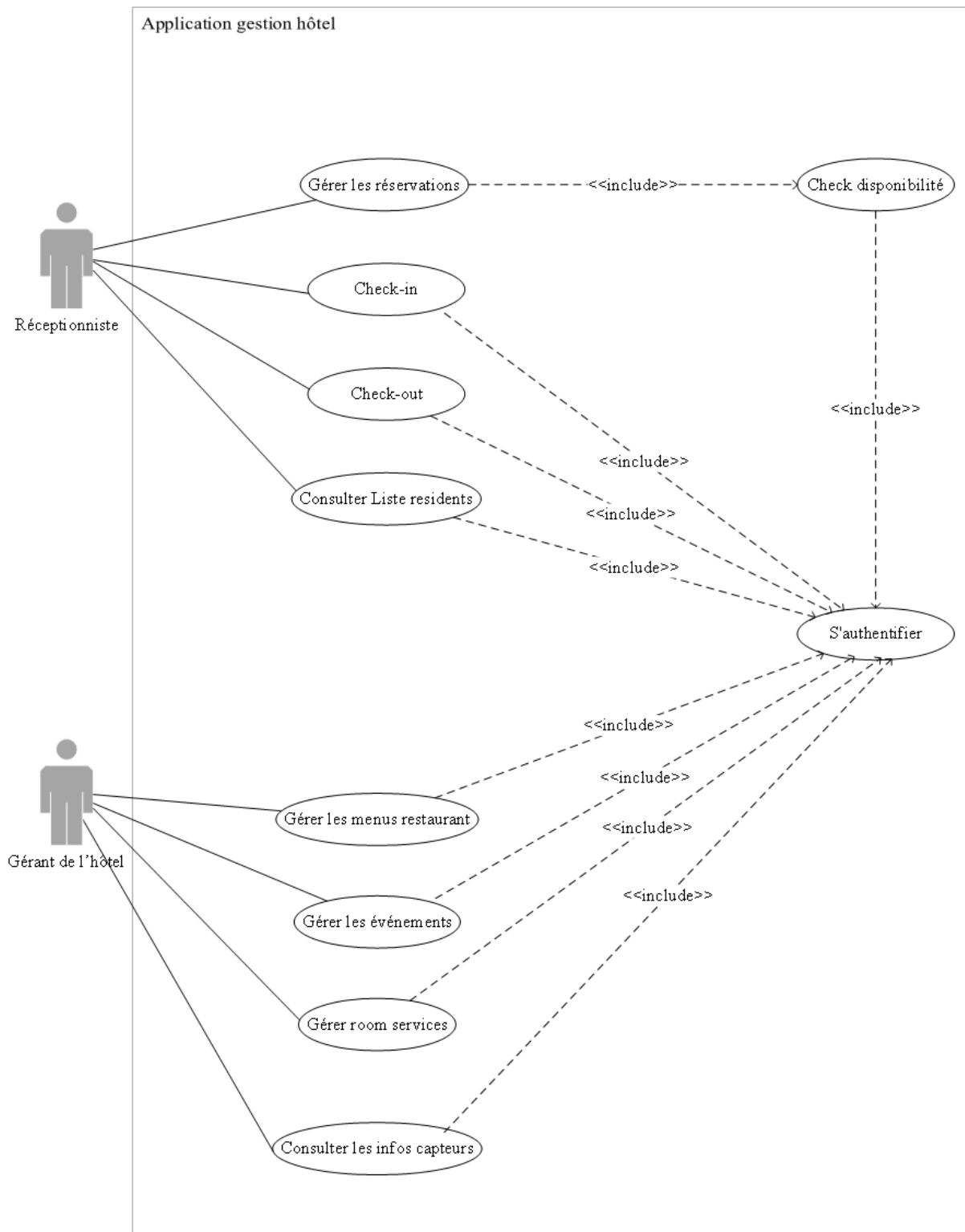


FIG. 4.2 : Diagramme de cas d'utilisation de l'application gestion de l'hôtel

Voici le diagramme du fonctionnement de notre système (maquette) pour la capture des informations contextuelles, on parlera en détail de sa conception dans la section qui suit.

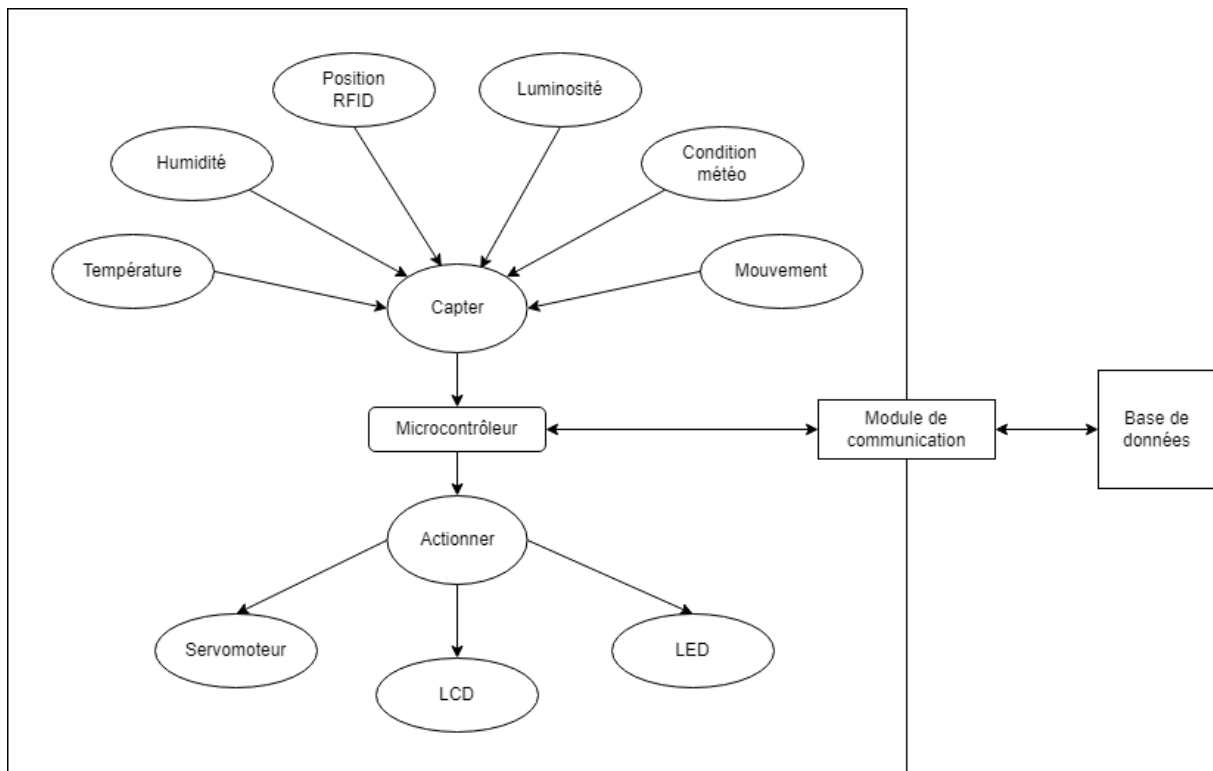


FIG. 4.3 : Diagramme du fonctionnement de notre système de capture

4.4.3 Spécification des cas d'utilisation

Notre spécification des cas se compose en deux étapes, la première est un tableau qui donne une description des cas d'utilisation de nos systèmes. Et la deuxième étape est le diagramme de séquence, Il décrit les objets concernés par le scénario et l'ordre des messages échangés entre les objets nécessaires à la réalisation des fonctionnalités.

Le diagramme de séquence est représenté que pour les cas d'utilisation complexes, les cas basique n'ont qu'une description textuelle.

- *Cas d'utilisation commun :*

S'authentifier

Description textuelle du cas d'utilisation :

Cas d'utilisation	S'authentifier
Acteurs	Client, Résident, réceptionniste et gérant de l'hôtel.
Objectifs	Vérifier l'autorisation pour accéder au système.
Pré-condition	Compte existant.
Post-condition	Autorisation d'accès accordée.
Scénario nominal	1- L'utilisateur demande l'accès à l'application. 2- Le système affiche le formulaire d'authentification. 3- L'utilisateur remplit le formulaire (nom d'utilisateur et mot de passe). 4- Le système vérifie la validité du mot de passe et ouvre une session utilisateur.
Scénario alternatif	4.a- Le nom de l'utilisateur et/ou le mot de passe sont erronés, Le système affiche une notification d'erreur « Le nom ou mot de passe est incorrect ». • Le scénario nominal reprend au point 2.

TAB. 4.2 : Description du cas d'utilisation S'authentifier

Diagramme de séquence du cas d'utilisation :

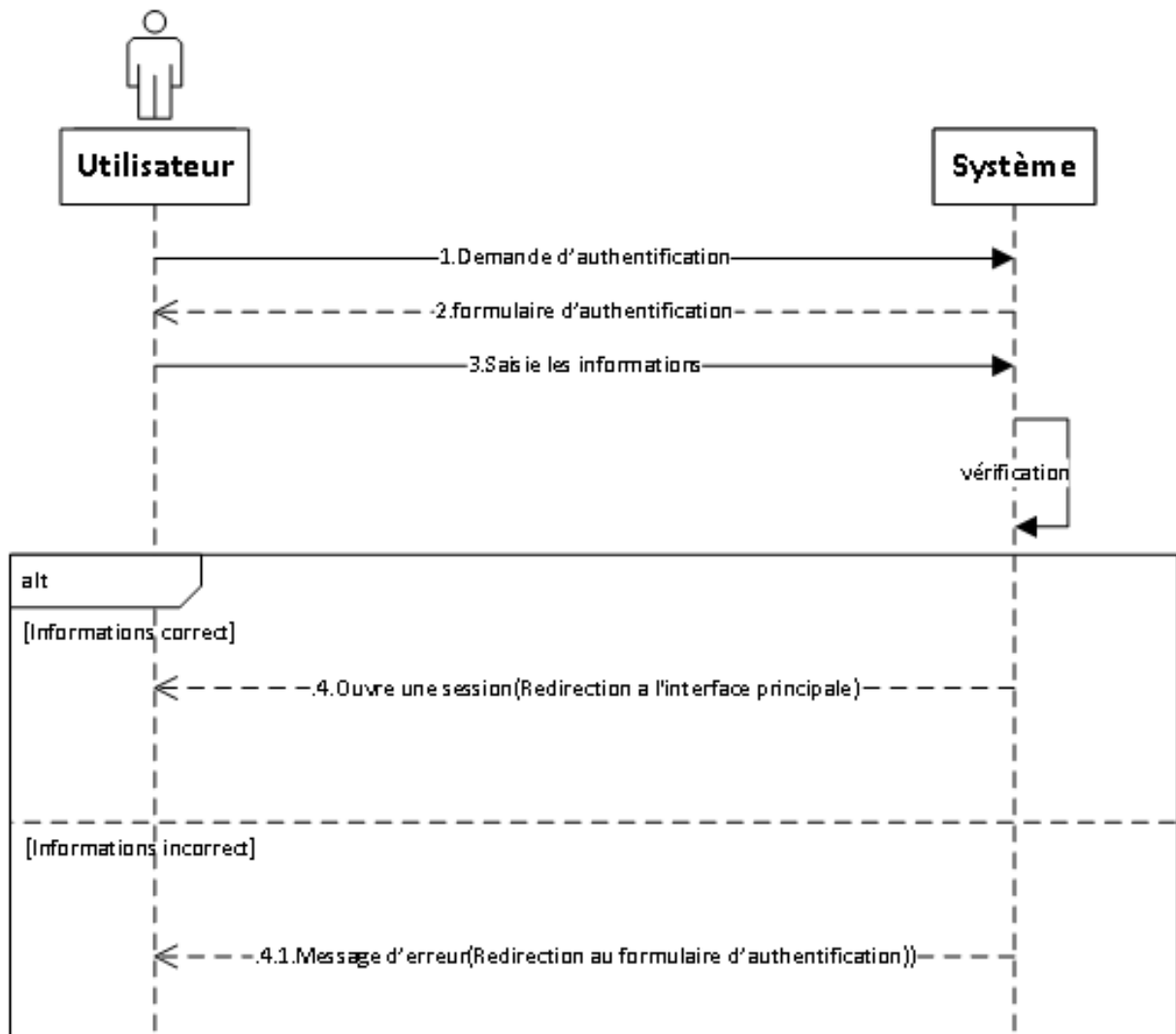


FIG. 4.4 : Diagramme de séquence S'authentifier

- **Application Mobile de l'hôtel :**

Consulter les informations de l'hôtel

Description textuelle du cas d'utilisation :

Cas d'utilisation	Consulter les informations de l'hôtel
Acteurs	Visiteur
Objectifs	Voir les informations relatives à l'hôtel, ainsi que les services offerts et les événements.
Pré-condition	Ouvrir l'application mobile de l'hôtel.
Post-condition	La consultation de la présentation de l'hôtel avec succès.

Scénario nominal	1- Le visiteur ouvre l'application de l'hôtel. 2- Le système affiche les informations relatives à l'hôtel, ainsi que les activités, services, événements qu'offre l'hôtel.
Scénario alternatif	/

TAB. 4.4 : Description du cas d'utilisation Consulter les informations de l'hôtel

Créer compte

Description textuelle du cas d'utilisation :

Cas d'utilisation	Créer compte
Acteurs	Visiteur
Objectifs	Créer un compte pour pouvoir bénéficier de l'application.
Pré-condition	Ouvrir l'application mobile de l'hôtel.
Post-condition	Un nouveau compte est ajouté.
Scénario nominal	1- L'utilisateur demande le formulaire de création. 2- Le système affiche le formulaire. 3- L'utilisateur remplit les champs nécessaires et valide. 4- Le système vérifie les données saisies. 5- Le système enregistre le nouvel utilisateur et affiche un message de succès de l'opération.
Scénario alternatif	4.a- L'utilisateur saisie des données manquantes ou invalides. 4.a-1- Le système affiche un message d'erreur. 4.a-2- Reprise de l'étape 3 du scénario nominal. 4.b- Le compte existe déjà. 4.b-1- Le système demande à l'utilisateur de modifier l'identifiant ou se connecter. 4.b-2- Reprise de l'étape 3 du scénario nominal.

TAB. 4.6 : Description du cas d'utilisation Créer compte

Diagramme de séquence du cas d'utilisation :

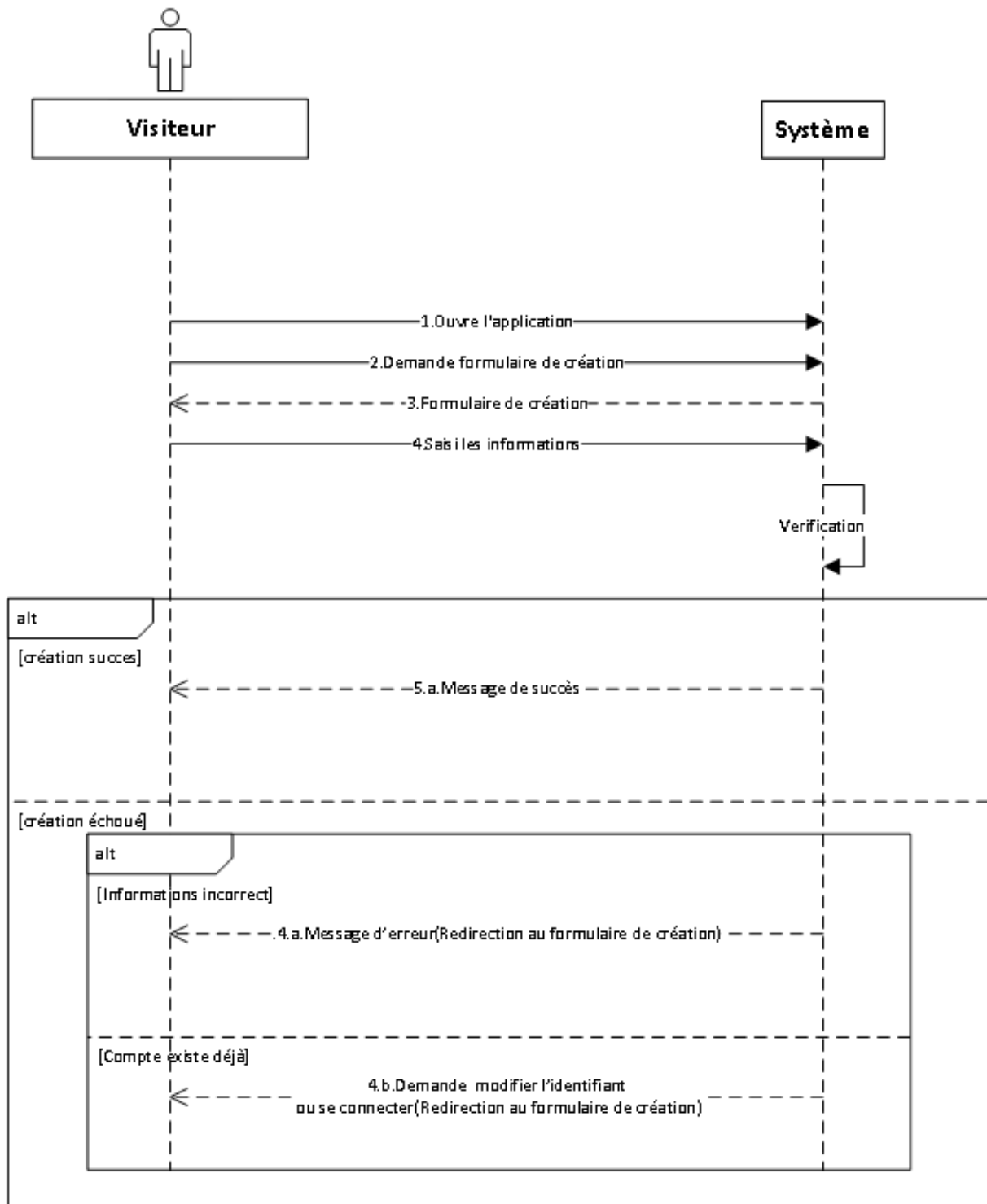


FIG. 4.5 : Diagramme de séquence Créer compte

Réservation de chambre

Description textuelle du cas d'utilisation :

Cas d'utilisation	Réservation de chambre
Acteurs	Client
Objectifs	Réserver une chambre dans l'hôtel et modifier ou supprimer cette Réservation.
Pré-condition	S'authentifier.
Post-condition	La réservation d'une chambre d'hôtel ou sa modification ou sa suppression est effectuée avec succès.
Scénario nominal	1- Le client ouvre l'application de l'hôtel et s'authentifie. 2- Le visiteur demande le formulaire de réservation. 3- Le système affiche le formulaire. 4- L'utilisateur remplit les champs nécessaires et valide. 5- Le système vérifie les données saisies, et affiche un message de succès de l'opération.
Scénario alternatif	5.1- Le système affiche un message d'erreur lorsque l'utilisateur fournit des données incomplètes ou erronées. • Le scénario nominal reprend au point 3.

TAB. 4.8 : Description du cas d'utilisation Réservation de chambre

Diagramme de séquence du cas d'utilisation :

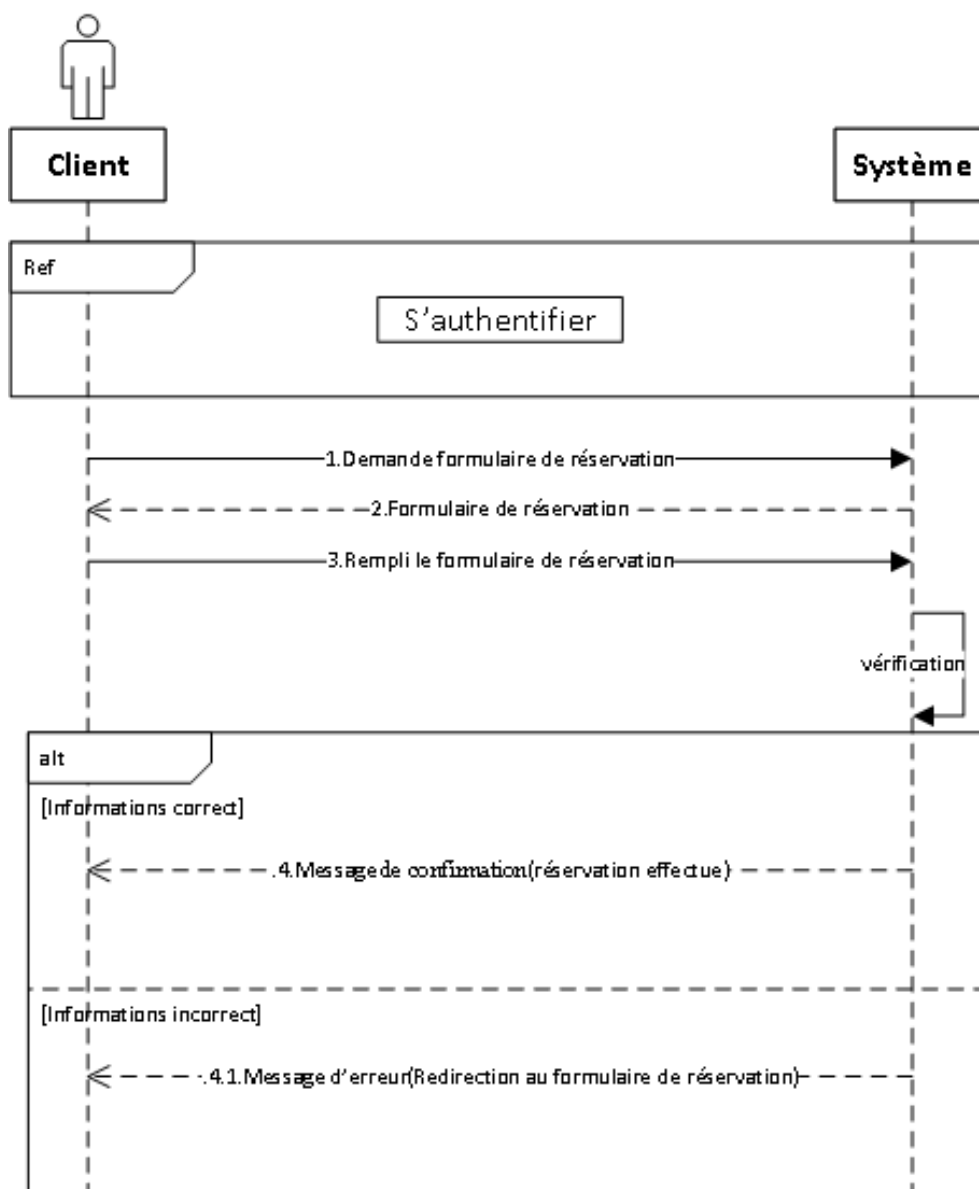


FIG. 4.6 : Diagramme de séquence Réservation de chambre

Mettre à jour informations compte

Description textuelle du cas d'utilisation :

Cas d'utilisation	Mettre à jour informations compte
Acteurs	Client
Objectifs	Mettre à jour (modifier) les informations personnelles du client.
Pré-condition	S'authentifier.
Post-condition	Le compte est modifié avec succès.

Scénario nominal	1- L'utilisateur demande l'interface compte. 2- Le système affiche l'interface. 3- L'utilisateur choisi l'information qui veut modifier. 4- Le système affiche le formulaire de modification. 5- L'utilisateur remplit le champ. 6- Le système vérifie les donnes, affiche le message confirmation et enregistre la mise à jour.
Scénario alternatif	6.1- Le système affiche un message d'erreur lorsque l'utilisateur fournit des données incomplètes ou erronées. • Le scénario nominal reprend au point 4.

TAB. 4.10 : Description du cas d'utilisation Mettre à jour informations compte

Diagramme de séquence du cas d'utilisation :

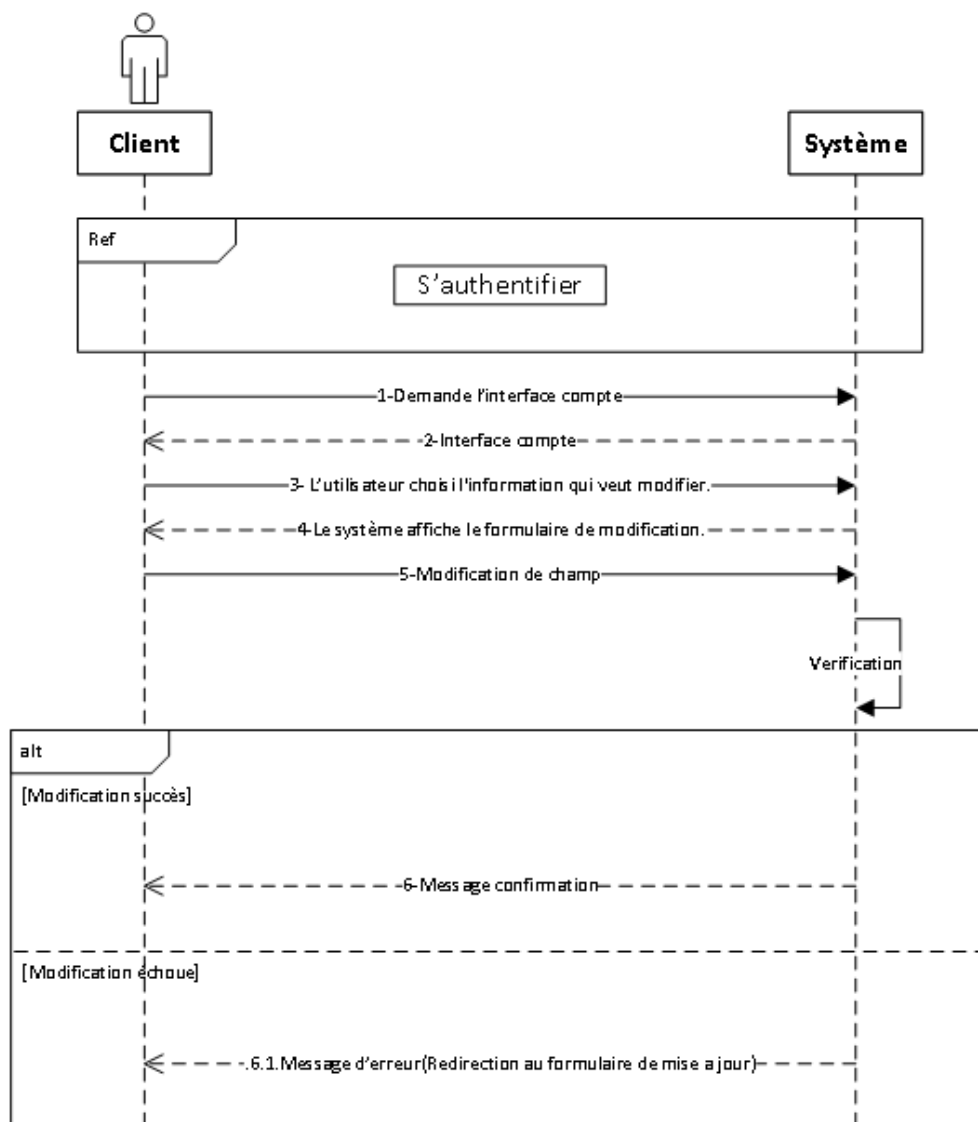


FIG. 4.7 : Diagramme de séquence Mettre à jour informations compte

Gérer préférences

Description textuelle du cas d'utilisation :

Cas d'utilisation	Gérer préférences
Acteurs	Client
Objectifs	Permet à un Client d'ajouter ou de supprimer ses préférence.
Pré-condition	S'authentifier.
Post-condition	Les préférence d'un résident sont ajouter ou supprimer.
Scénario nominal	1- L'utilisateur demande l'interface de Gestion de préférences. 2- Le système affiche l'interface. 3- L'utilisateur choisi ses préférences dans la liste. 4- Le système enregistre et affiche un message de confirmation.
Scénario alternatif	/

TAB. 4.12 : Description du cas d'utilisation Gérer préférences

Valider code séjour

Description textuelle du cas d'utilisation :

Cas d'utilisation	Valider code séjour
Acteurs	Client
Objectifs	Scanner un QR-code donner lors du check-in pour valider une réservation et être un résident.
Pré-condition	S'authentifier. Avoir un QR-code de validation.
Post-condition	Accès au fonctionnalités d'un résident.
Scénario nominal	1- L'utilisateur demande l'interface de validation. 2- Le système affiche l'interface. 3- L'utilisateur scanne un QR-code. 4- Le système vérifie le QR-code, affiche le message confirmation.
Scénario alternatif	4.1- Le système affiche un message d'erreur lorsque l'utilisateur fournit un QR-code erronées ou invalide. • Le scénario nominal reprend au point 3.

TAB. 4.14 : Description du cas d'utilisation Valider code séjour

Diagramme de séquence du cas d'utilisation :

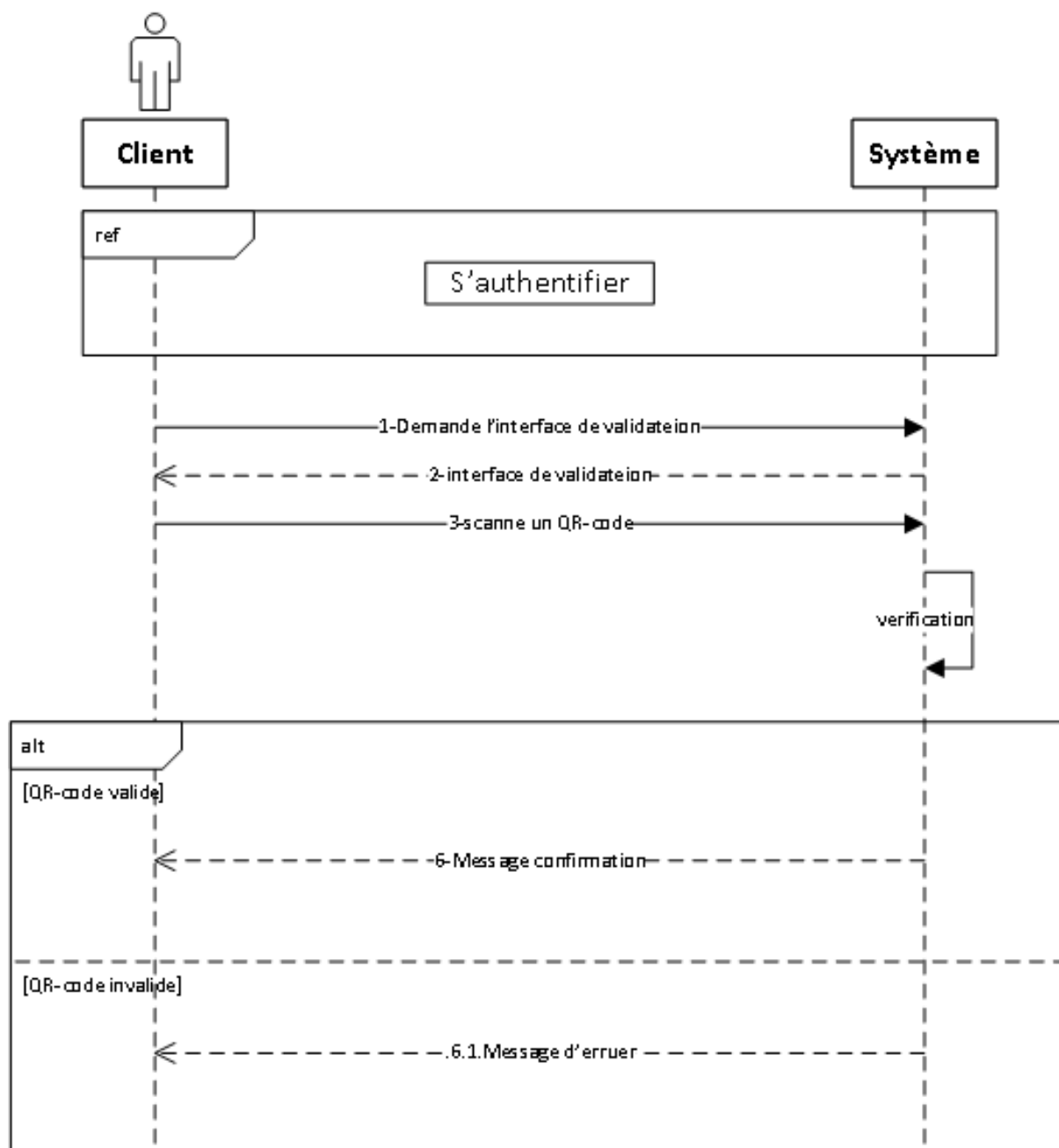


FIG. 4.8 : Diagramme de séquence Valider code séjour

Consulter des centres d'intérêt près de l'hôtel

Description textuelle du cas d'utilisation :

Cas d'utilisation	Consulter des centres d'intérêt près de l'hôtel
Acteurs	Résident

Objectifs	Voir dans une carte les centres d'intérêt près de l'hôtel et sa position grasse au GPS.
Pré-condition	S'authentifier. Check-in est effectuée. Valider un code séjour.
Post-condition	Les centres d'intérêt sont consulté avec succès.
Scénario nominal	1- L'utilisateur demande l'interface de consultation des centres d'intérêt. 2- Le système affiche une carte avec les centres d'intérêt. 3- L'utilisateur choisi un centre d'intérêt. 4- Le système affiche les informations de ce dernier.
Scénario alternatif	/

TAB. 4.16 : Description du cas d'utilisation Consulter des centres d'intérêt près de l'hôtel

Diagramme de séquence du cas d'utilisation :

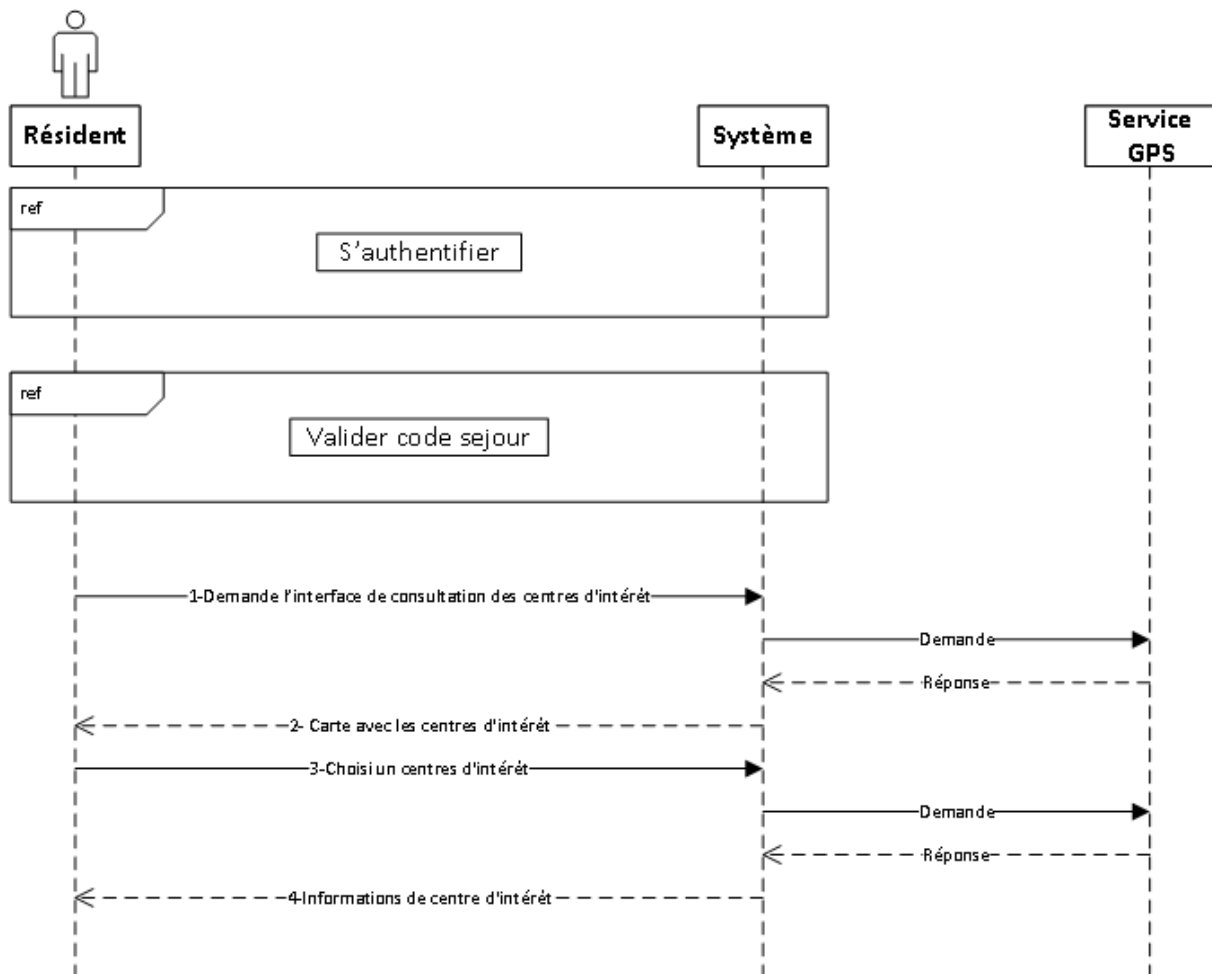


FIG. 4.9 : Diagramme de séquence Consulter des centres d'intérêt près de l'hôtel

Gérer Smart Room

Description textuelle du cas d'utilisation :

Cas d'utilisation	Gérer Smart Room
Acteurs	Résident
Objectifs	Interagir avec la chambre de l'hôtel et la contrôler.
Pré-condition	S'authentifier. Check-in est effectuée. Valider un code séjour.
Post-condition	La chambre de l'hôtel est contrôler par son résident.
Scénario nominal	1- L'utilisateur demande l'interface de Smart Room. 2- Le système affiche l'interface. 3- L'utilisateur choisi une des plusieurs options de contrôle possible.
Scénario alternatif	/

TAB. 4.18 : Description du cas d'utilisation Gérer smart Room

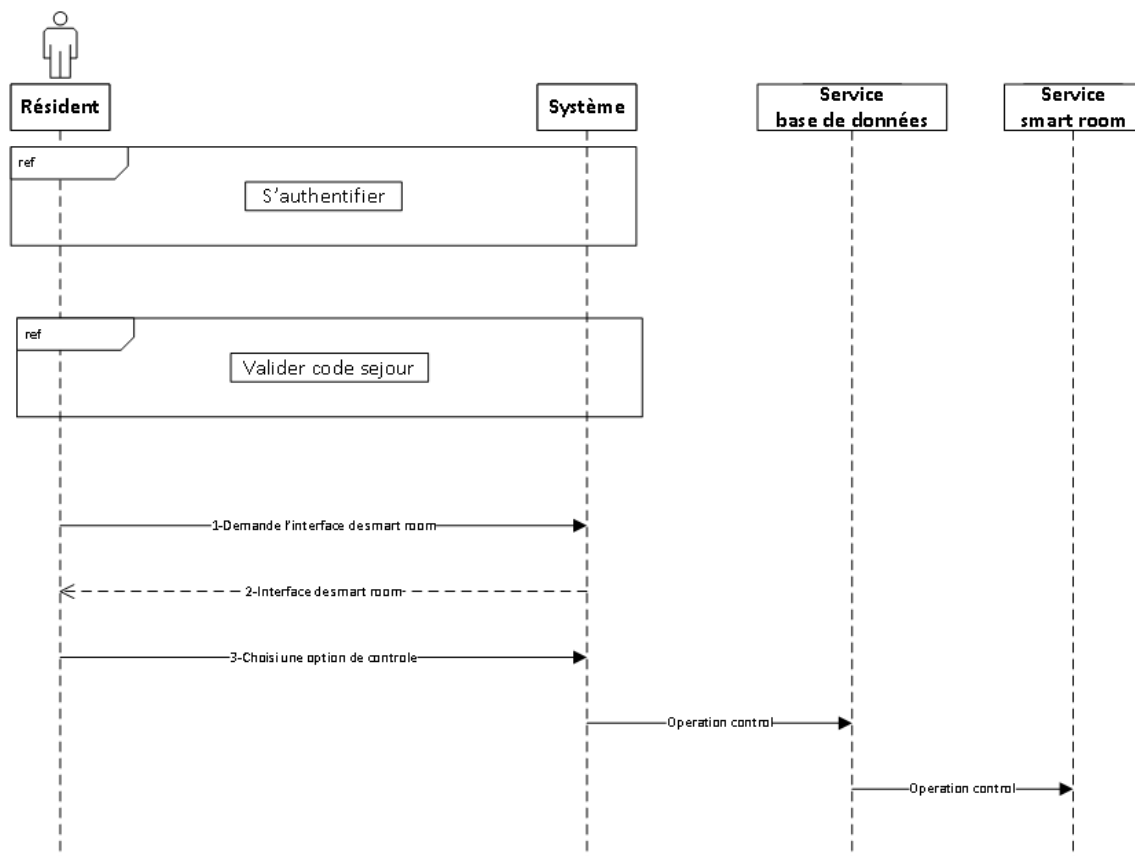


FIG. 4.10 : Diagramme de séquence Gérer Smart Room

Demander Room service

Description textuelle du cas d'utilisation :

Cas d'utilisation	Demander Room service
Acteurs	Résident
Objectifs	Demander des services en chambre des repas, des petits déjeuners, des boissons etc (demandes et livraisons).
Pré-condition	S'authentifier. Check-in est effectuée. Valider un code séjour.
Post-condition	La demande est envoyer avec succès.
Scénario nominal	1- L'utilisateur demande l'interface Room service. 2- Le système affiche l'interface. 3- L'utilisateur choisi un ou plusieurs services. 4- Le système affiche le message de confirmation. 5- L'utilisateur confirme la demande. 6- Le système vérifie, affiche le message de succès.
Scénario alternatif	5.1- L'utilisateur annule la demande. • Le scénario nominal reprend au point 2.

TAB. 4.20 : Description du cas d'utilisation Demander Room service

Diagramme de séquence du cas d'utilisation :

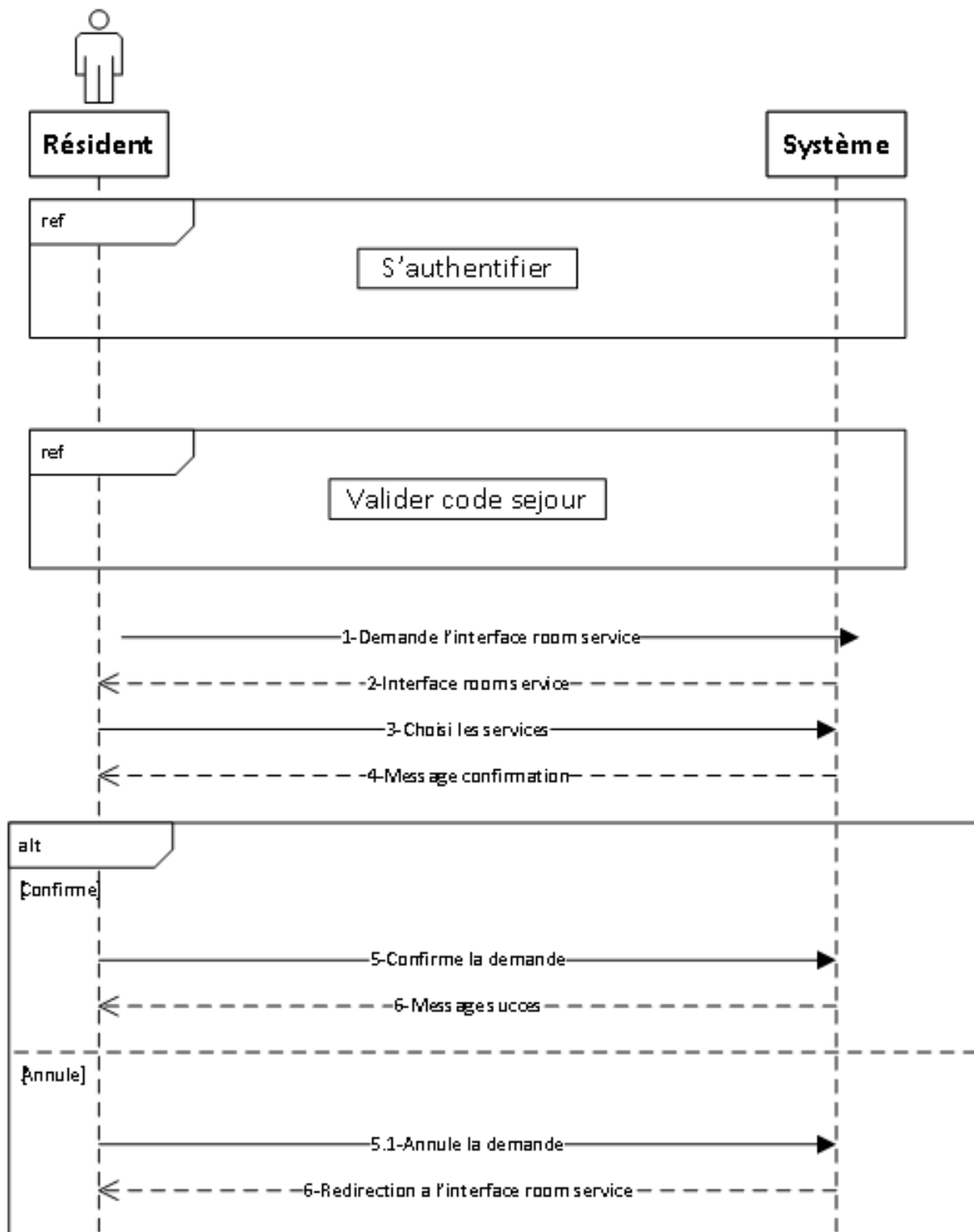


FIG. 4.11 : Diagramme de séquence Demander Room service

Consulter Menu restaurant

Description textuelle du cas d'utilisation :

Cas d'utilisation	Consulter Menu restaurant
Acteurs	Résident
Objectifs	Voir les informations du Menu restaurant.
Pré-condition	S'authentifier. Check-in est effectuée. Valider un code séjour.
Post-condition	Le Menu est consulté avec succès.
Scénario nominal	1- L'utilisateur demande l'interface de menu restaurant. 2- Le système affiche l'interface du menu. 3- L'utilisateur choisi un plat dans la liste du menu. 4- Le système affiche les informations sur se plat.
Scénario alternatif	/

TAB. 4.22 : Description du cas d'utilisation Consulter Menu restaurant

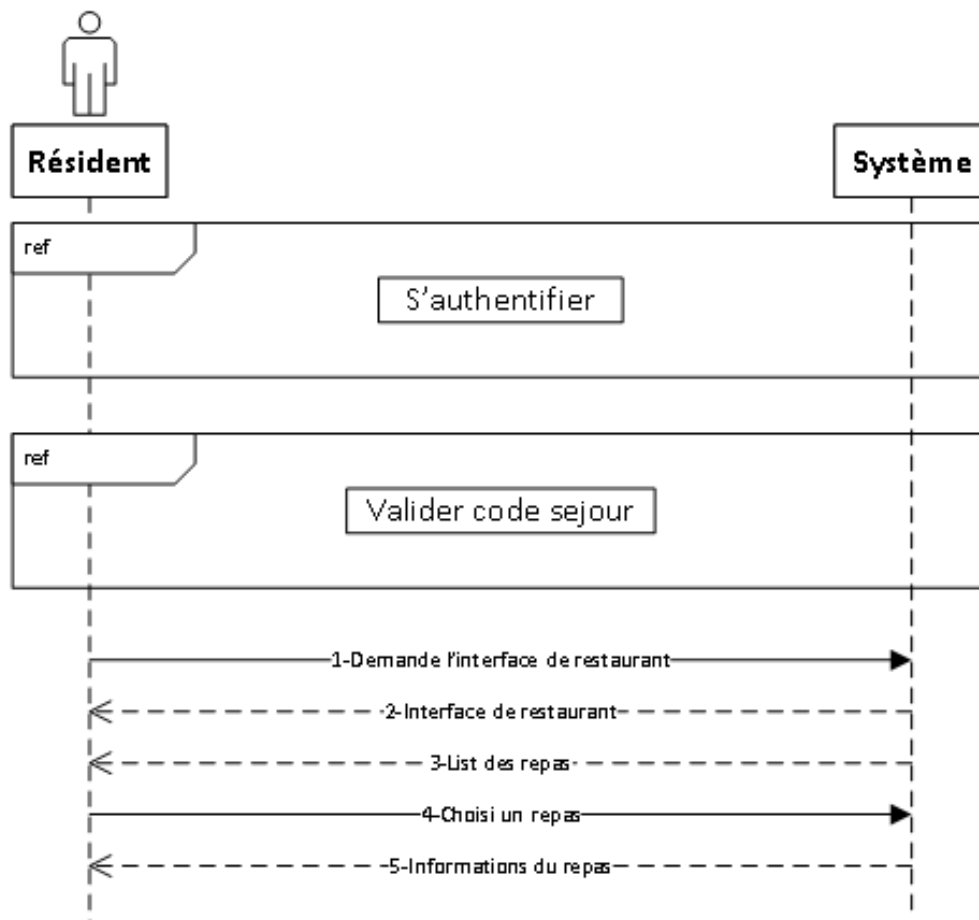


FIG. 4.12 : Diagramme de séquence Consulter Menu restaurant

Consulter événements

Description textuelle du cas d'utilisation :

Cas d'utilisation	Consulter événements
Acteurs	Résident
Objectifs	Voir les informations relatives aux évènements, la date et le lieu.
Pré-condition	S'authentifier. Check-in est effectuée. Valider un code séjour.
Post-condition	L'évènement est consultée avec succès.
Scénario nominal	1- L'utilisateur demande l'interface des évènements. 2- Le système affiche l'interface des évènements. 3- L'utilisateur choisi un évènement qui l'intéresse. 4- Le système affiche les informations de l'évènement.
Scénario alternatif	/

TAB. 4.24 : Description du cas d'utilisation Consulter événements

Diagramme de séquence du cas d'utilisation :

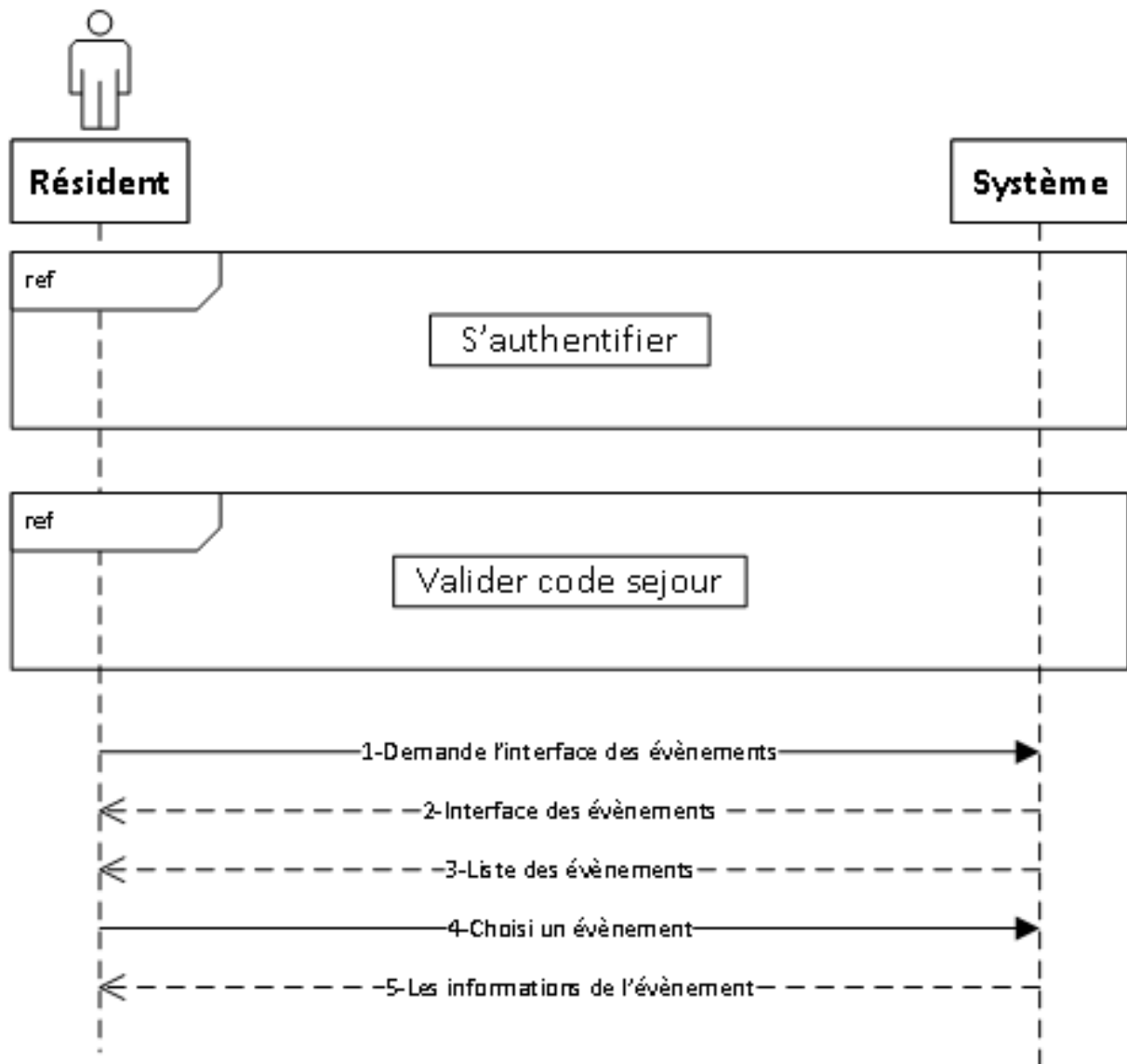


FIG. 4.13 : Diagramme de séquence Consulter événements

• Application gestion de l'hôtel :

Gestion des réservations

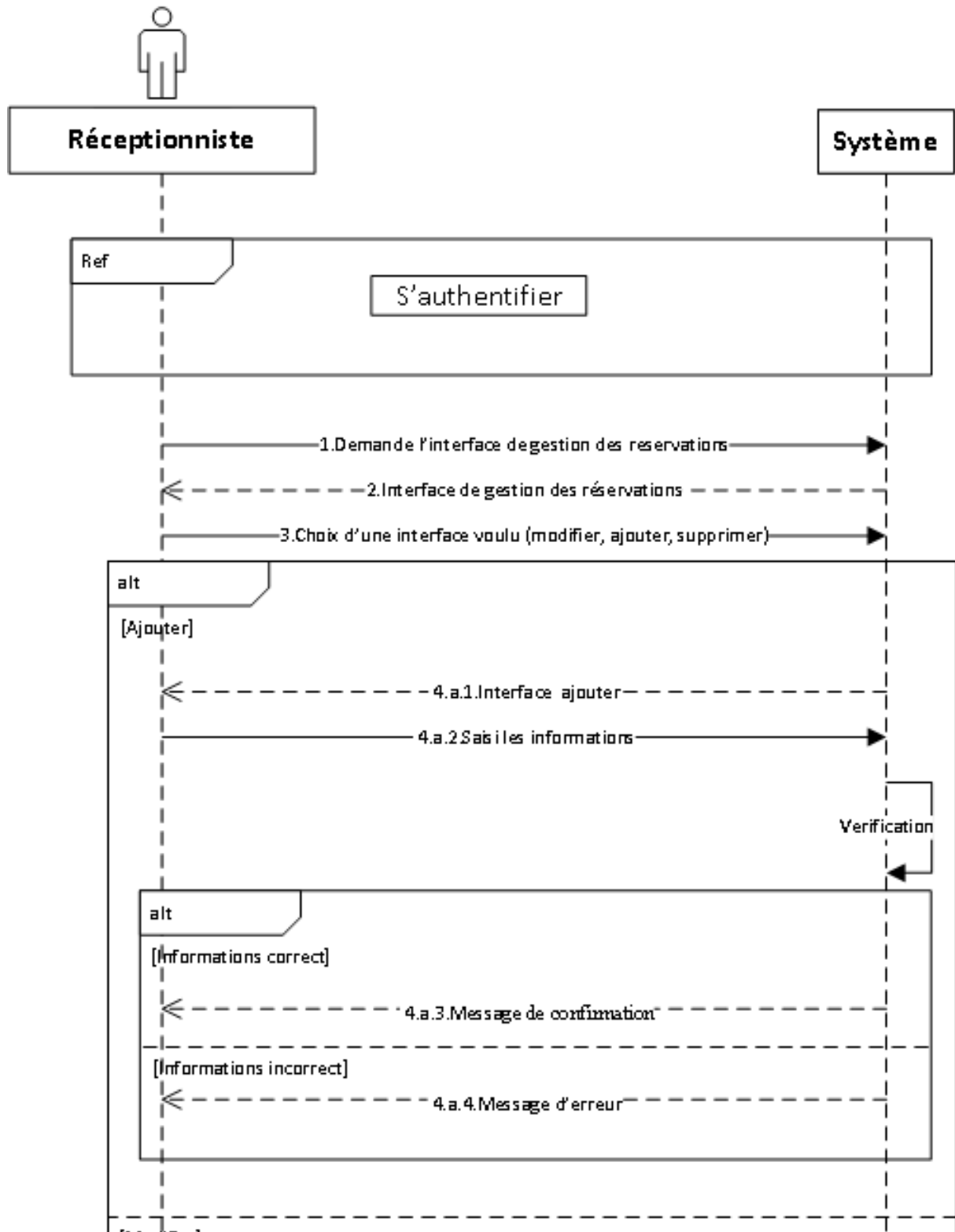
Description textuelle du cas d'utilisation :

Cas d'utilisation	Gestion des réservations
Acteurs	Réceptionniste
Objectifs	Permet à l'utilisateur de gérer les réservation des clients.
Pré-condition	S'authentifier.

Post-condition	La gestion des réservations est effectuée avec succès.
Scénario nominal	1- L'utilisateur demande l'ajout, modifie ou supprime une réservation. 2- Le système affiche l'interface voulu.
Scénario alternatif	/
Cas	Ajouter réservation
Objectifs	Permet à l'utilisateur de réserver une chambre.
Scénario nominal	1- L'utilisateur demande l'ajout d'une réservation. 2- Le système affiche le formulaire de réservation. 3- L'utilisateur remplit le formulaire et enregistre. 4- Le système enregistre et affiche un message de confirmation.
Scénario alternatif	3.1- Le système affiche un message d'erreur lorsque l'utilisateur fournit des données incomplètes ou erronées. • Le scénario nominal reprend au point 2.
Cas	Modifier réservation
Objectifs	Permet à l'utilisateur de modifier une réservation.
Scénario nominal	1- L'utilisateur demande la modification d'une réservation. 2- Le système affiche le formulaire de modification. 3- L'utilisateur choisit une réservation, remplit le formulaire et enregistre. 4- Le système enregistre et affiche un message de confirmation.
Scénario alternatif	3.1- Le système affiche un message d'erreur lorsque l'utilisateur fournit des données incomplètes ou erronées. • Le scénario nominal reprend au point 2.
Cas	Supprimer réservation
Objectifs	Permet à l'utilisateur de Supprimer une réservation.
Scénario nominal	1- L'utilisateur demande la suppression d'une réservation. 2- Le système affiche l'interface de suppression. 3- L'utilisateur choisit la réservation, clic supprimer et enregistre. 4- Le système enregistre et affiche un message de confirmation.
Scénario alternatif	4.1- L'utilisateur annule la suppression. 5.1- Le système affiche un message d'erreur. • Le scénario nominal reprend au point 2.

TAB. 4.26 : Description du cas d'utilisation Gestion des réservations

Diagramme de séquence du cas d'utilisation :



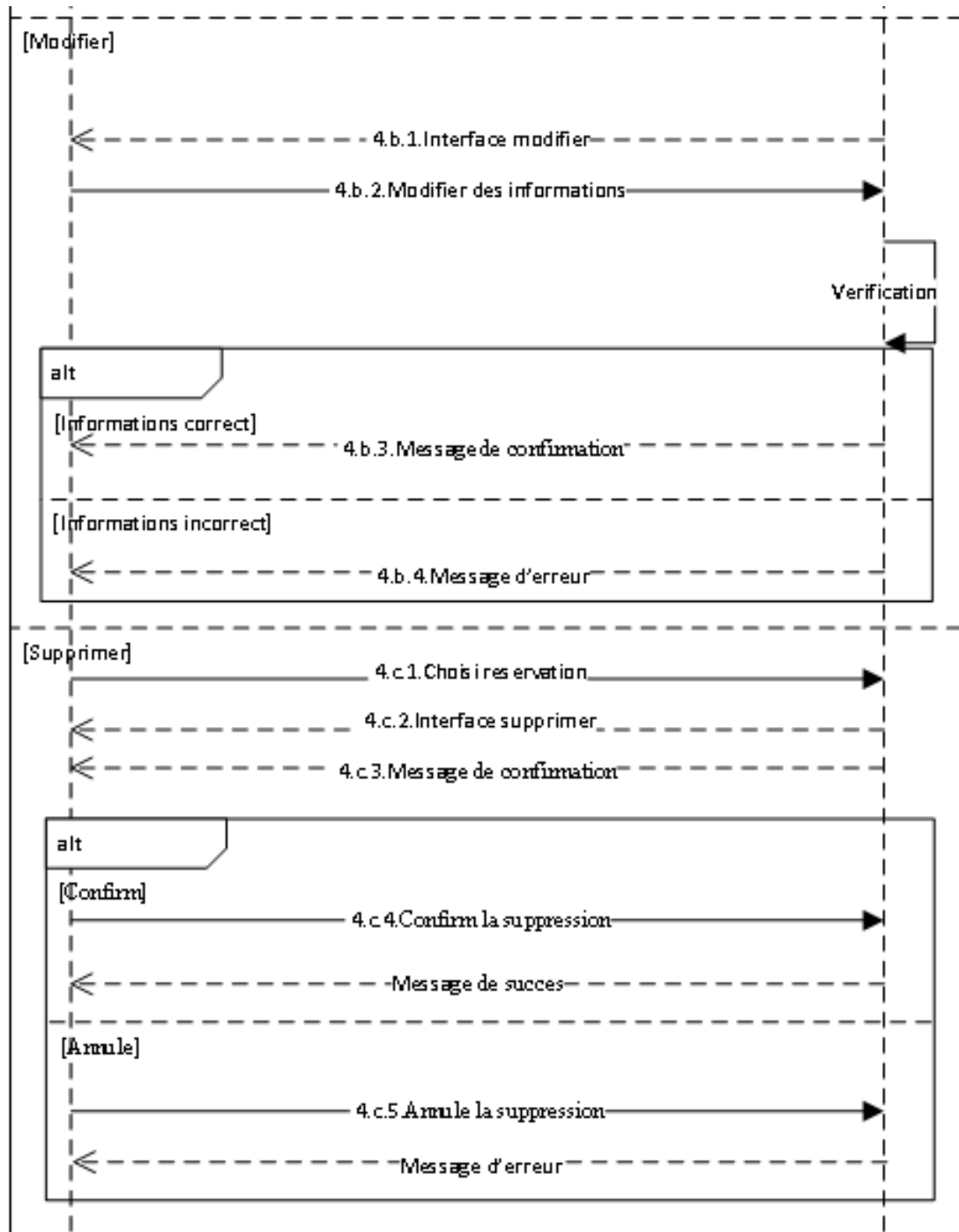


FIG. 4.14 : Diagramme de séquence Gestion des réservations

Check de disponibilité

Description textuelle du cas d'utilisation :

Cas d'utilisation	Check de disponibilité
Acteurs	Réceptionniste
Objectifs	Permet à l'utilisateur de voir si une chambre est disponible.

Pré-condition	S'authentifier.
Post-condition	La réponse concernant la disponibilité est affichée.
Scénario nominal	1- L'utilisateur demande au système de vérifier la disponibilité d'une chambre. 2- Le système affiche le résultat de la requête concernant la disponibilité.
Scénario alternatif	/

TAB. 4.28 : Description du cas d'utilisation Check de disponibilité

Check-in d'un client

Description textuelle du cas d'utilisation :

Cas d'utilisation	Check-in
Acteurs	Réceptionniste
Objectifs	Permet à l'utilisateur de sélectionner une réservation d'un client, de saisir ses informations, et donner à ce dernier un code de validation de séjour.
Pré-condition	S'authentifier.
Post-condition	Le check-in est fait avec succès.
Scénario nominal	1- L'utilisateur demande au système l'interface de check-in. 2- Le système affiche l'interface voulu. 3- L'utilisateur sélectionne une réservation existante. 4- L'utilisateur saisi les informations nécessaires et enregistre. 5- Le système affiche un QR-code, enregistre et confirme.
Scénario alternatif	4.1- Le système affiche un message d'erreur lorsque l'utilisateur fournit des données incomplètes ou erronées. • Le scénario nominal reprend au point 3.

TAB. 4.30 : Description du cas d'utilisation Check-in

Diagramme de séquence du cas d'utilisation :

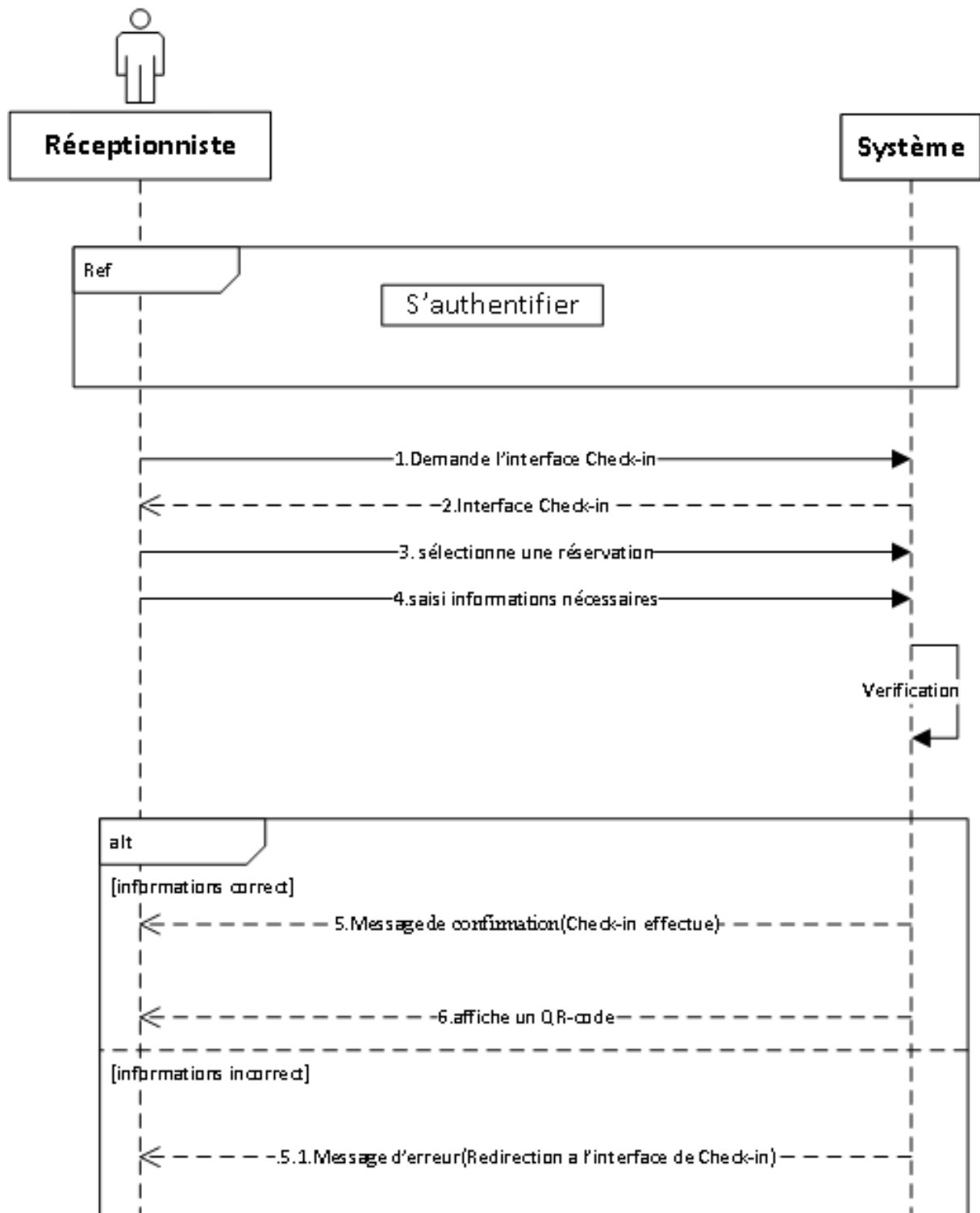


FIG. 4.15 : Diagramme de séquence Check-in

Check-out d'un client

Description textuelle du cas d'utilisation :

Cas d'utilisation	Check-out
Acteurs	Réceptionniste
Objectifs	Permet à l'utilisateur de supprimer un résident, retirer le contrôle de sa chambre et la rendre disponible.
Pré-condition	S'authentifier.
Post-condition	Le check out est fait avec succès.
Scénario nominal	1- L'utilisateur demande au système l'interface check-out. 2- Le système affiche l'interface voulu. 3- L'utilisateur saisi choisi le Résident et clic sur check-out. 4- Le système affiche enregistre et confirme.
Scénario alternatif	/

TAB. 4.32 : Description du cas d'utilisation Check-out

Diagramme de séquence du cas d'utilisation :

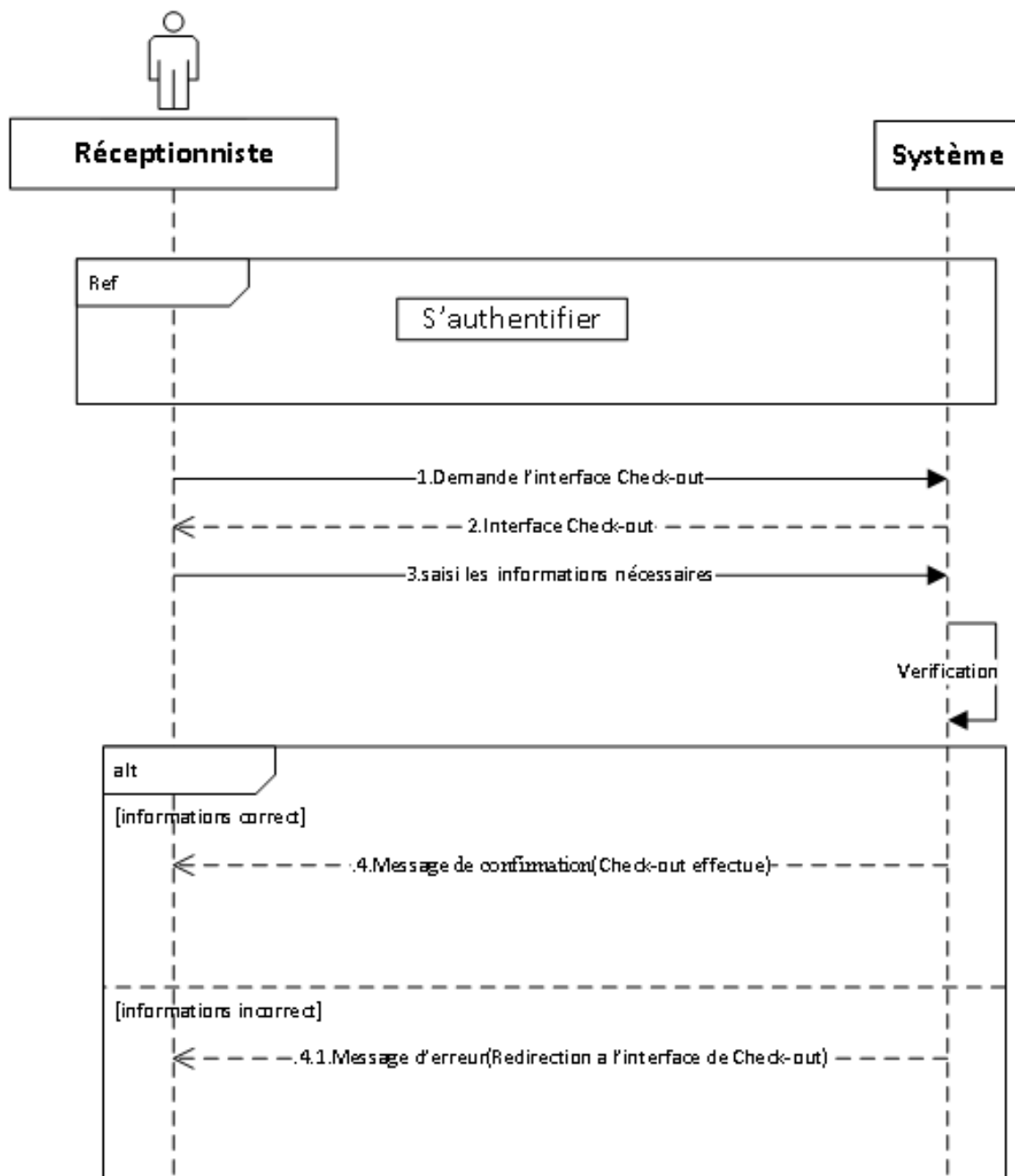


FIG. 4.16 : Diagramme de séquence Check-out

Consulter liste résidents

Description textuelle du cas d'utilisation :

Cas d'utilisation	Consulter liste résidents
Acteurs	Réceptionniste
Objectifs	Voir la liste des résidents dans l'hôtel et leur détail.
Pré-condition	S'authentifier.
Post-condition	Le liste des résidents est consulté avec succès.
Scénario nominal	1- L'utilisateur demande l'interface liste résidents. 2- Le système affiche l'interface de la liste des résidents.
Scénario alternatif	/

TAB. 4.34 : Description du cas d'utilisation Consulter liste résidents

Consulter les informations Capteurs

Description textuelle du cas d'utilisation :

Cas d'utilisation	Consulter les informations Capteurs
Acteurs	Gérant de l'hôtel
Objectifs	Récupérer les informations captées des différents capteurs physiques.
Pré-condition	S'authentifier.
Post-condition	Les informations captées sont exploitées.
Scénario nominal	1- L'administrateur demande la liste des informations Captées. 2- Le système affiche la liste des informations.
Scénario alternatif	/

TAB. 4.36 : Description du cas d'utilisation Consulter les informations Capteurs

Gestion du menu restaurant

Description textuelle du cas d'utilisation :

Cas d'utilisation	Gestion du menu restaurant
Acteurs	Gérant de l'hôtel
Objectifs	Saisir le menu des repas prévus pour une journée.
Pré-condition	S'authentifier.
Post-condition	Le menu est saisi.
Scénario nominal	1- Le gérant demande au système d'afficher le formulaire du menu. 2- Le système affiche le formulaire. 3- Le gérant remplit le formulaire avec les différents repas prévus pour le jour et enregistre. 4- Le système vérifie les informations, enregistre et affiche un message de confirmation.
Scénario alternatif	4.1- L'une des informations remplies est erronée ou un champ vide, le système affiche message d'erreur. • Le scénario nominal reprend au point 2.

TAB. 4.38 : Description du cas d'utilisation Gestion du menu restaurant

Diagramme de séquence du cas d'utilisation :

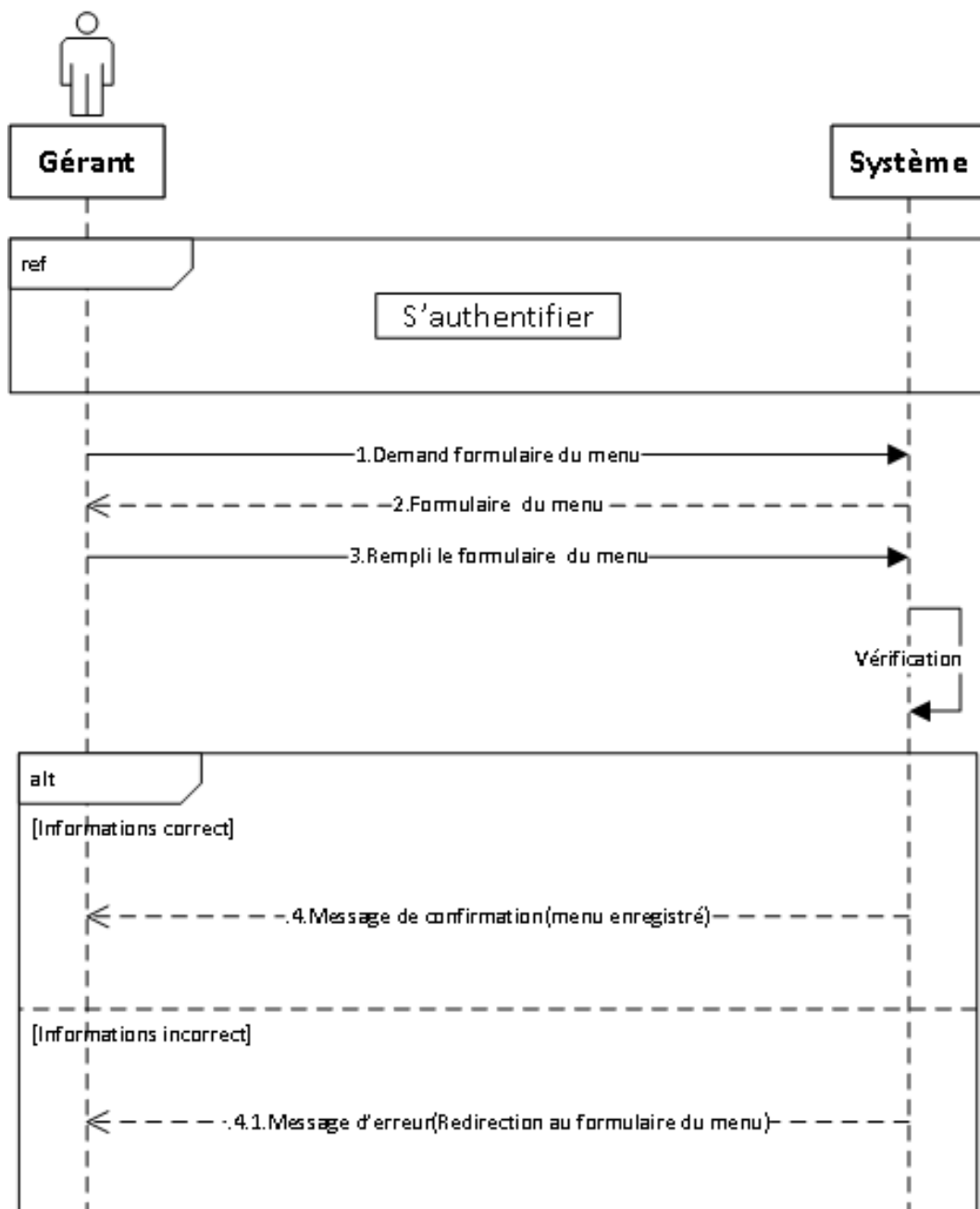


FIG. 4.17 : Diagramme de séquence Gestion du menu restaurant

Gestion des événements

Description textuelle du cas d'utilisation :

Cas d'utilisation	Gestion des événements
Acteurs	Gérant de l'hôtel
Objectifs	Saisir la liste des événements qui se dérouleront au sein de l'hôtel dans une journée ou les générer automatiquement.
Pré-condition	Le gérant de l'hôtel s'authentifie.
Post-condition	La liste est saisie.
Scénario nominal	1- Le gérant demande au système d'afficher le formulaire des événements. 2- Le système affiche le formulaire. 3- Le gérant remplit le formulaire avec les différents évènements qui auront lieu à l'hôtel ou les génère automatiquement et enregistre. 4- Le système vérifie les informations, enregistre et affiche un message de confirmation.
Scénario alternatif	4.1- L'une des informations remplies est erronée ou un champ est vide, le système affiche message d'erreur. • Le scénario nominal reprend au point 2.

TAB. 4.40 : Description du cas d'utilisation Gestion des événements

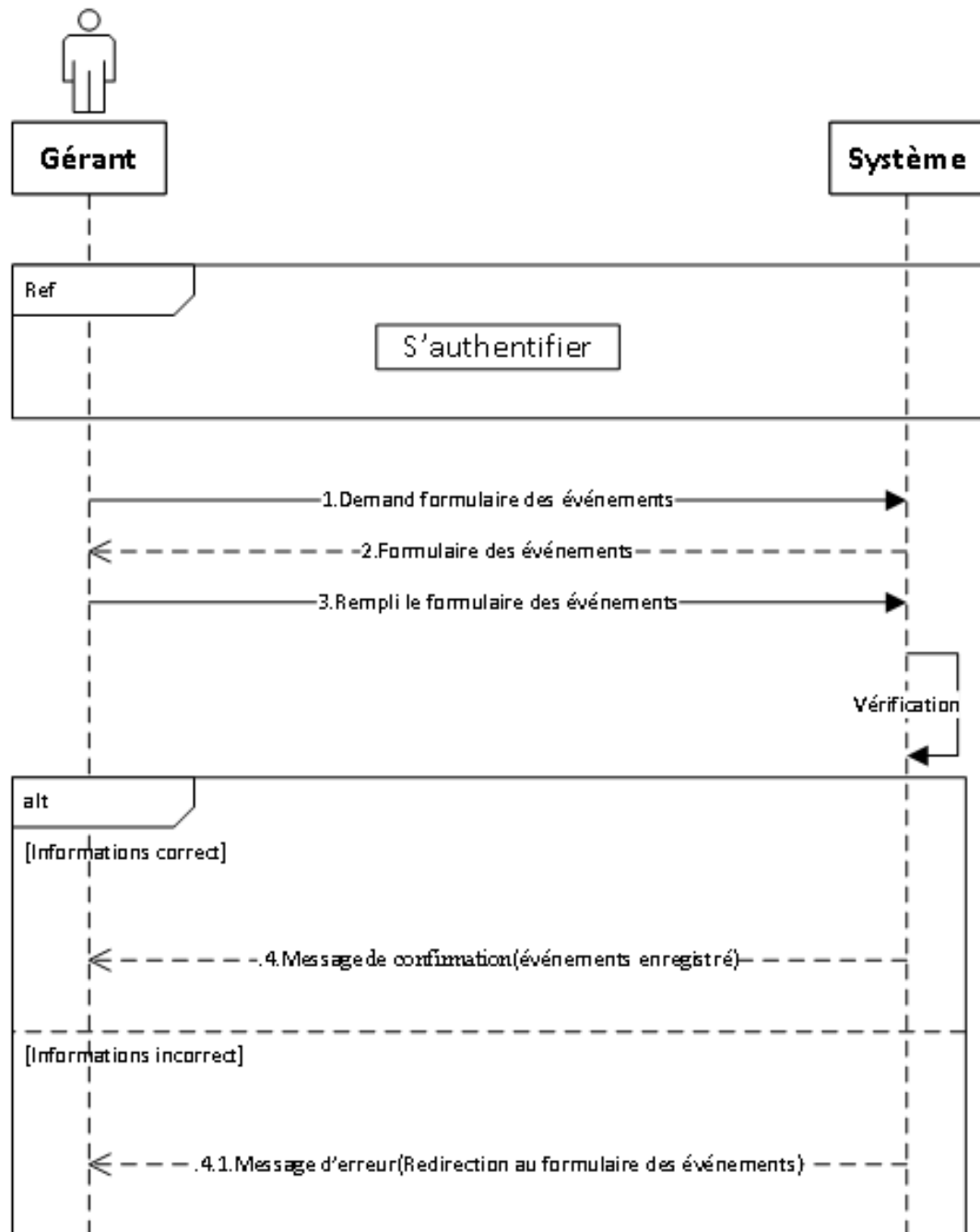


FIG. 4.18 : Diagramme de séquence Gestion des événements

Gestion du Room Service

Description textuelle du cas d'utilisation :

Cas d'utilisation	Gestion du Room Service
Acteurs	Gérant de l'hôtel

Objectifs	Permettre au maître d'hôtel (gérant) de veiller au bon déroulement du service en chambre des repas, des petits déjeuners, des boissons etc.
Pré-condition	S'authentifier.
Post-condition	La gestion du Room service est effectuée avec succès.
Scénario nominal	1- L'acteur demande l'interface de la gestion du Room service. 2- Le système affiche l'interface. 3- L'acteur valide les demandes existantes. 4- Le système vérifie les informations, enregistre et affiche un message de confirmation.
Scénario alternatif	/

TAB. 4.42 : Description du cas d'utilisation Gestion du Room Service

4.5 Conception

La conception est la dernière étape avant l'implémentation de l'application, c'est donc la réalisation du modèle qui sera mis en œuvre. Similaire à la phase d'analyse, UML utilise différents diagrammes pour la conception, Dans notre cas, nous allons établir le diagramme de classe participantes et le diagramme de séquence détaillé.

4.5.1 Diagramme de classes participantes

Ce diagramme effectue la jonction entre, d'une part, les cas d'utilisation, les modèles de la couche métiers et l'interface avec l'utilisateur. Il semble particulièrement important pour guider la phase de production du livrable final. Il se compose de 3 éléments (MVC) [51] :

- Dialogue : sont celles qui permettent les interactions entre les utilisateurs et l'application.
- Contrôle : contiennent la dynamique de l'application Elles font le lien entre les classes dialogue et les classes métier. Elles permettent de contrôler la cinématique de l'application, l'ordre dans lequel les choses doivent se dérouler.
- Métier : ou entités représentent les objets métier. Elles proviennent directement du modèle du domaine (mais peuvent être complétées en fonction des cas d'utilisation).

S'authentifier

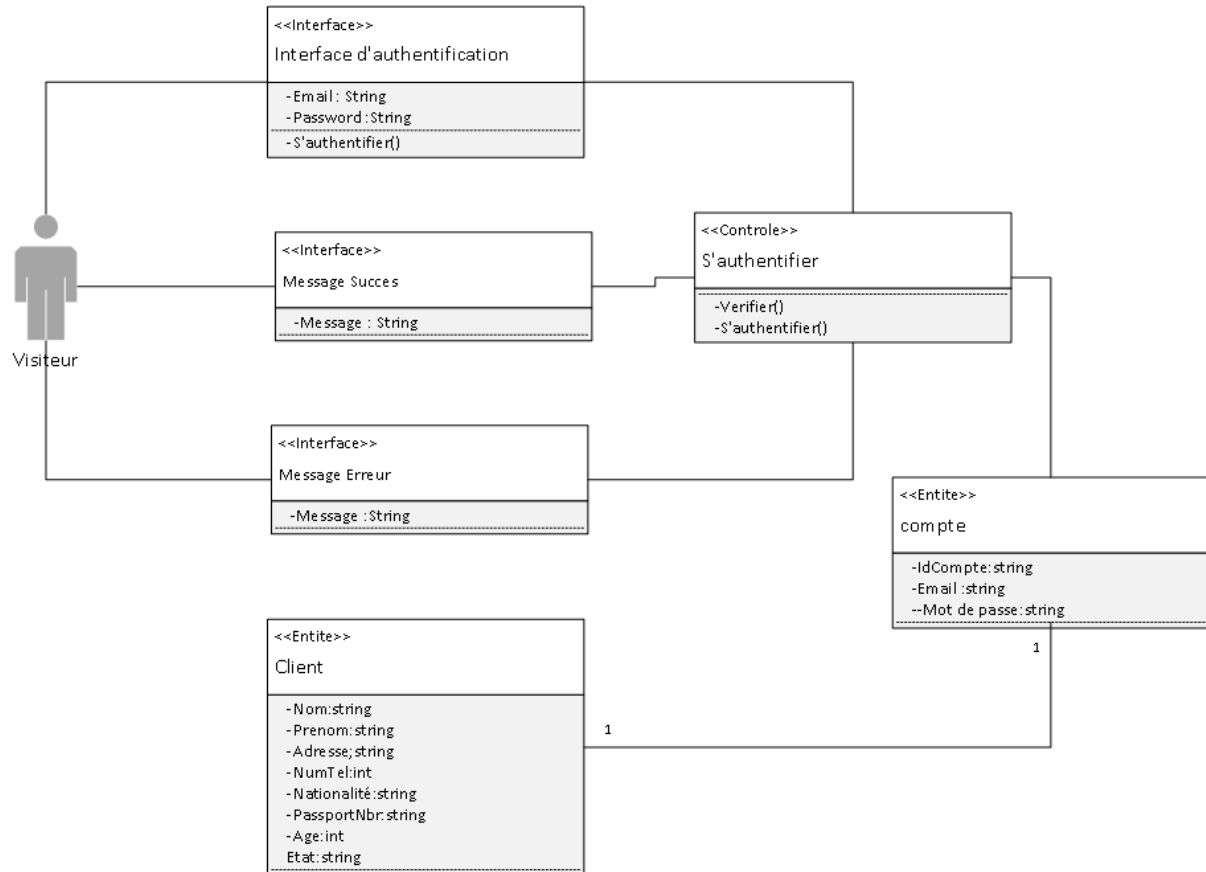


FIG. 4.19 : Diagramme de classes participantes S'authentifier

Consulter les informations de l'hôtel

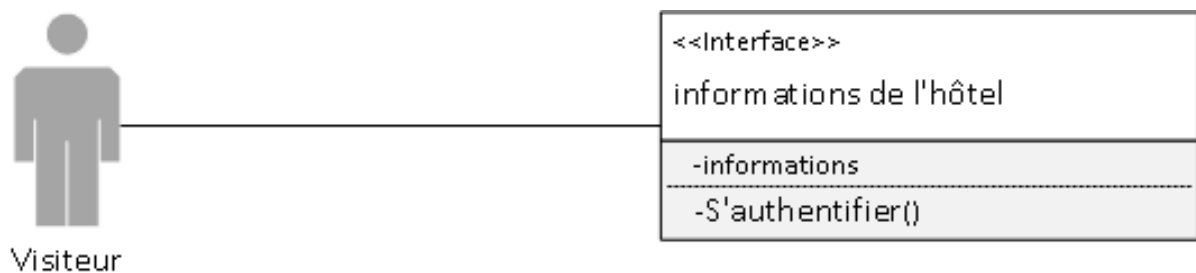


FIG. 4.20 : Diagramme de classes participantes Consulter les informations de l'hôtel

Créer compte

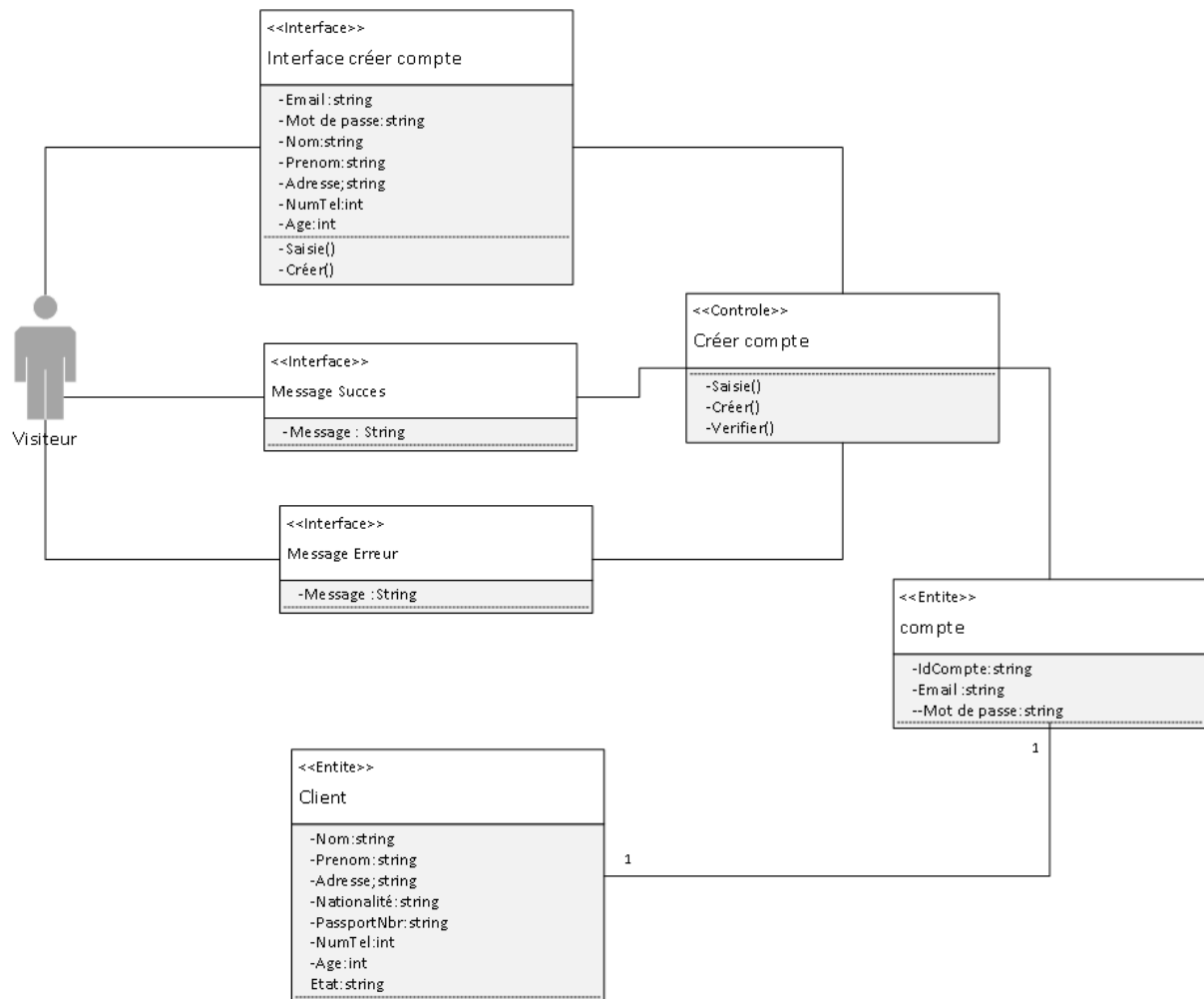


FIG. 4.21 : Diagramme de classes participantes Créer compte

Réservation de chambre

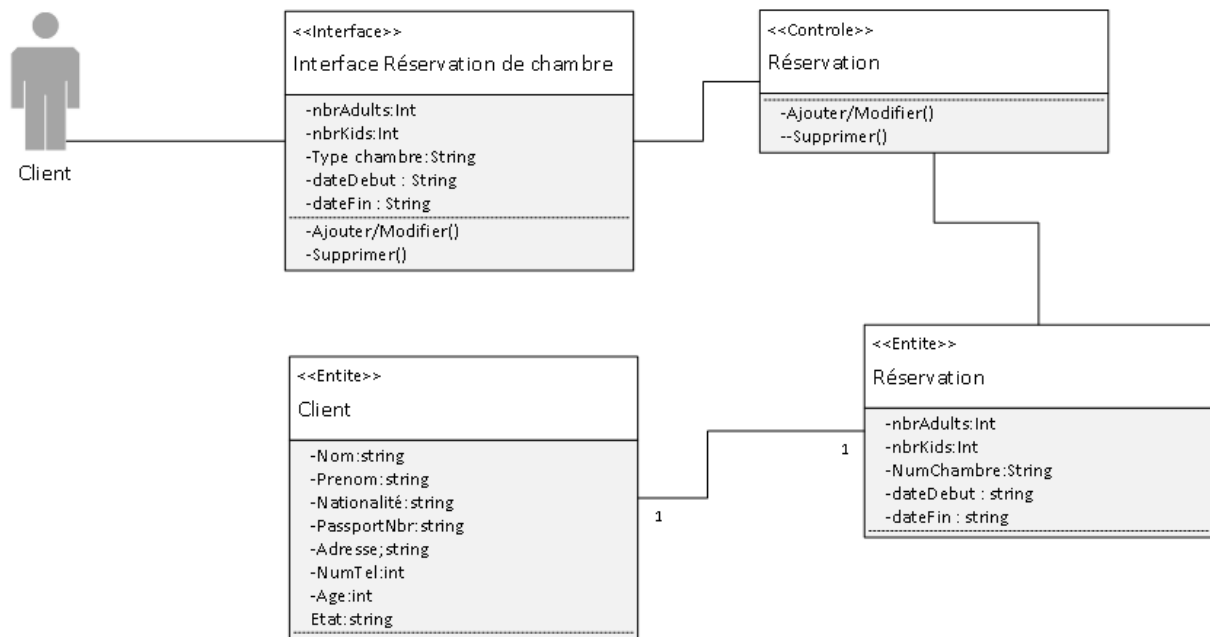


FIG. 4.22 : Diagramme de classes participantes Réservation de chambre

Mettre à jour informations compte

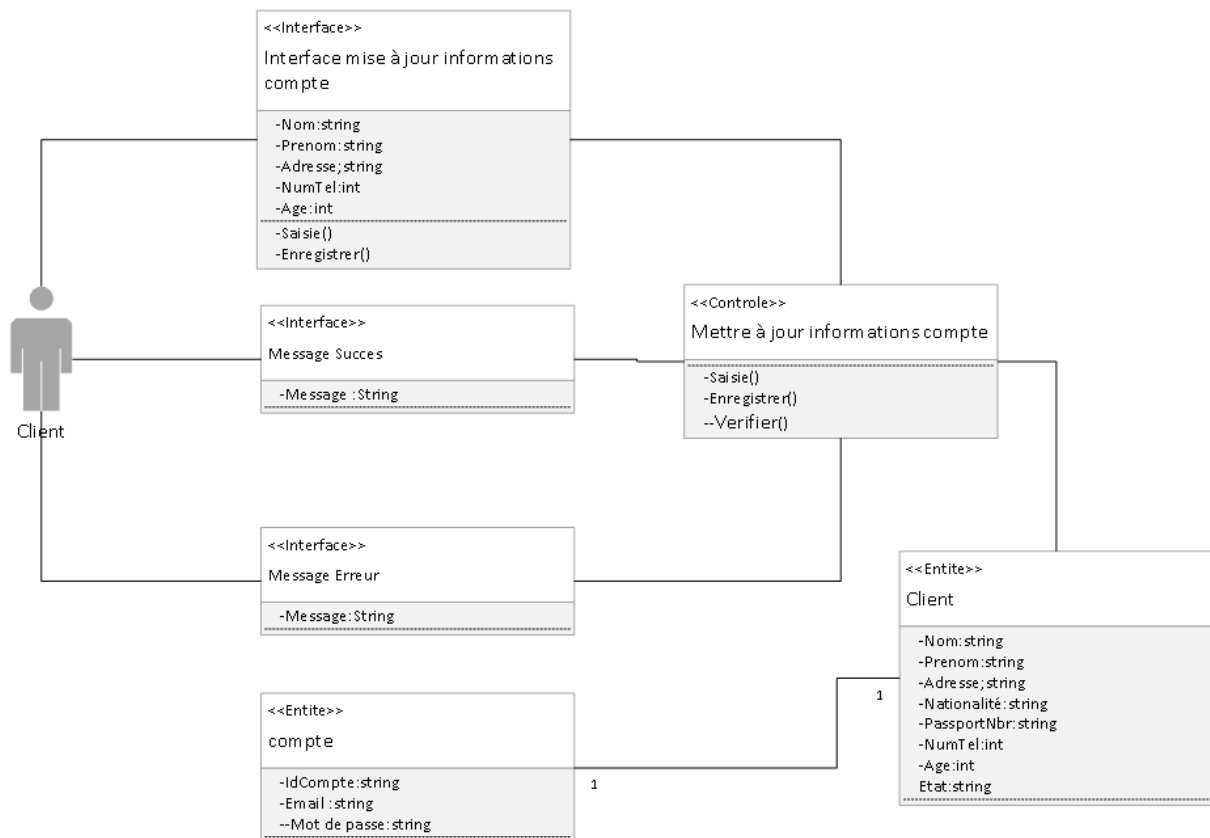


FIG. 4.23 : Diagramme de classes participantes Mettre à jour informations compte

Gérer préférences

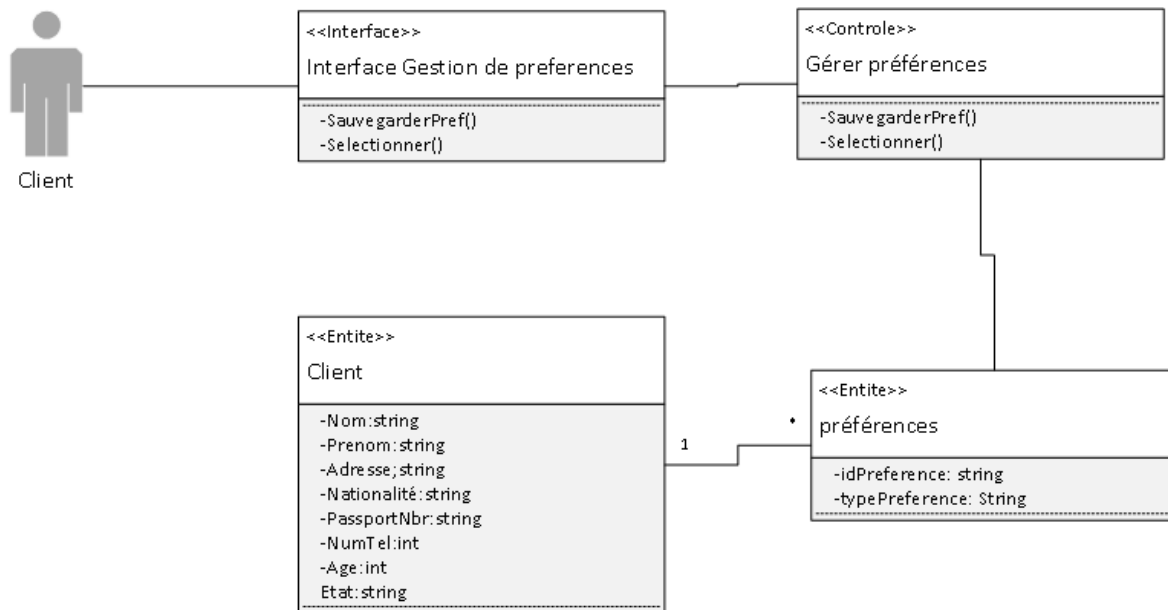


FIG. 4.24 : Diagramme de classes participantes Gérer préférences

Valider code séjour

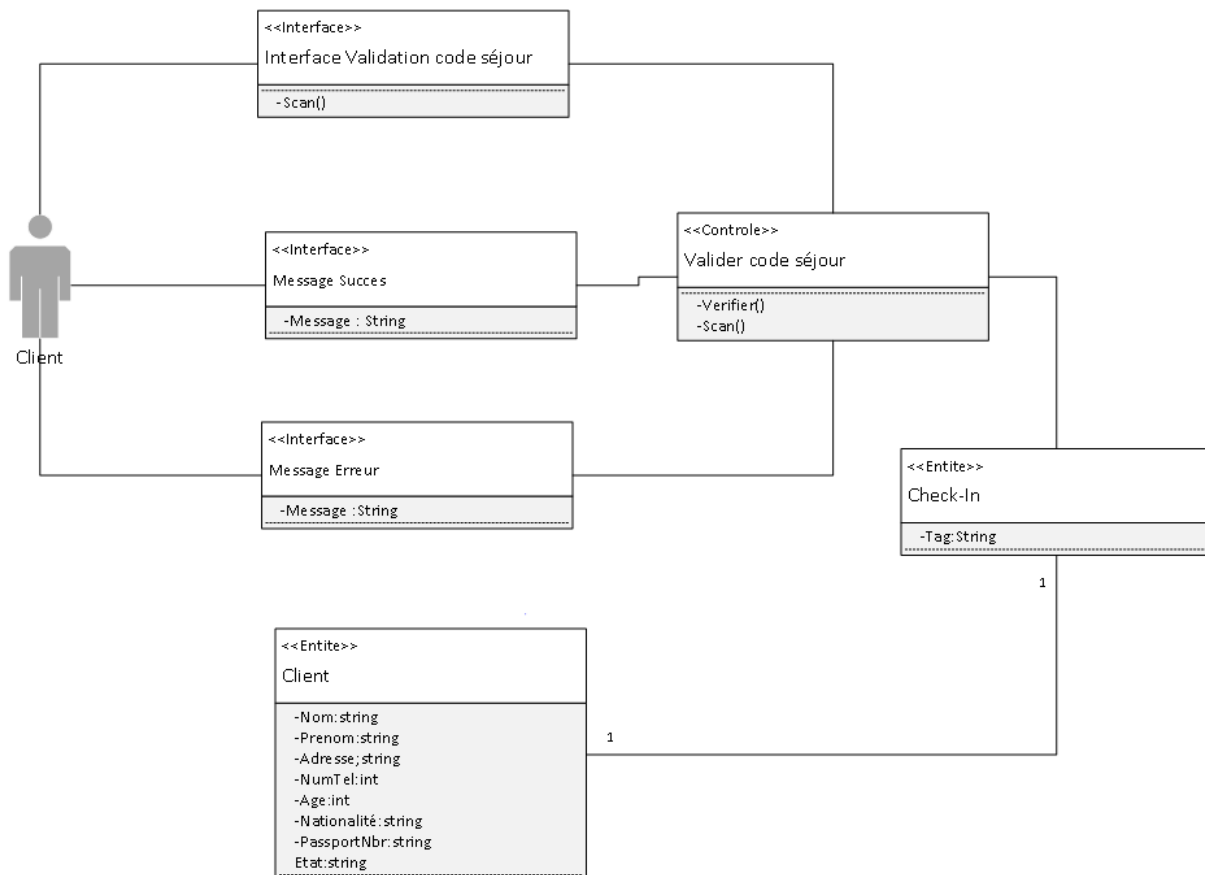


FIG. 4.25 : Diagramme de classes participantes Valider code séjour

Consulter des centres d'intérêt près de l'hôtel

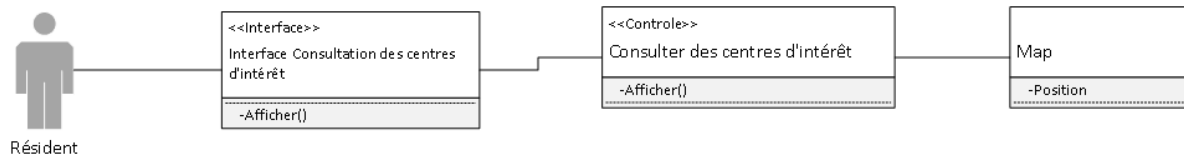


FIG. 4.26 : Diagramme de classes participantes Consulter des centres d'intérêt près de l'hôtel

Gérer Smart Room

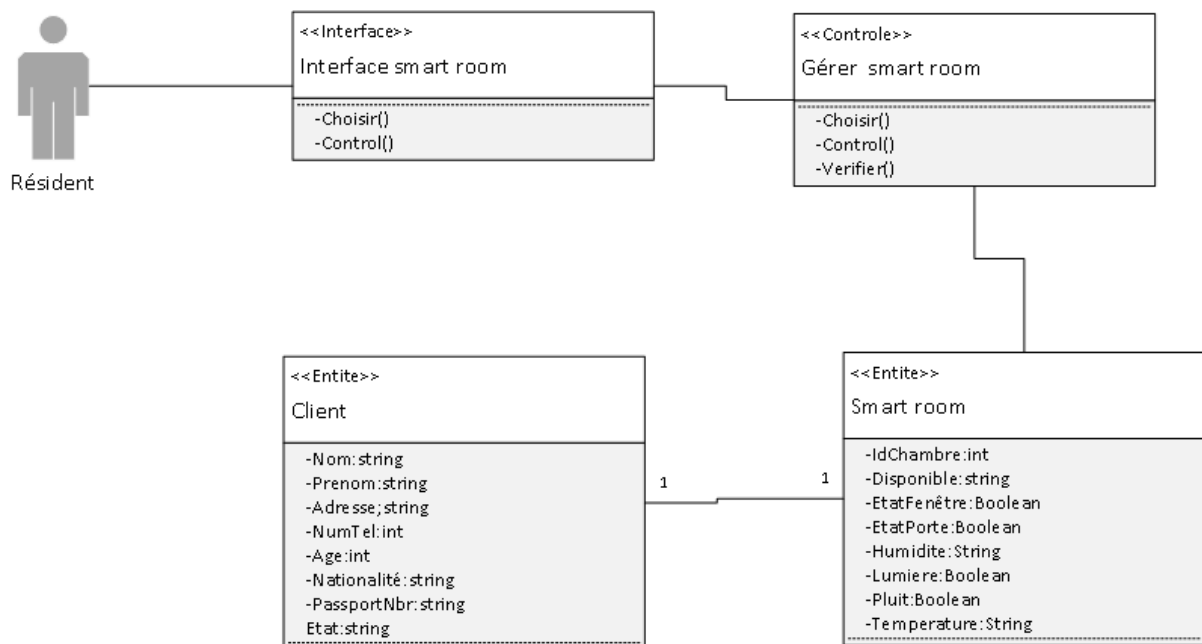


FIG. 4.27 : Diagramme de classes participantes Gérer Smart Room

Demander Room service

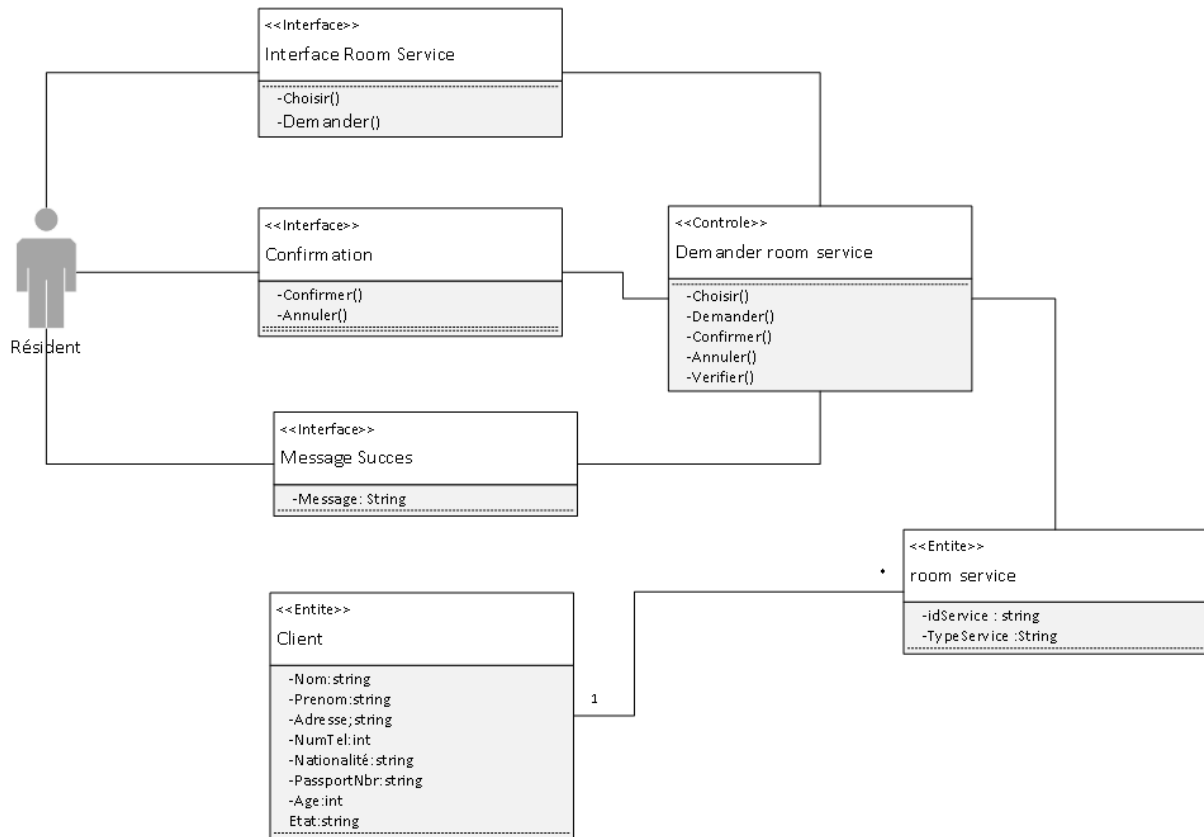


FIG. 4.28 : Diagramme de classes participantes Demander Room service

Consulter Menu restaurant

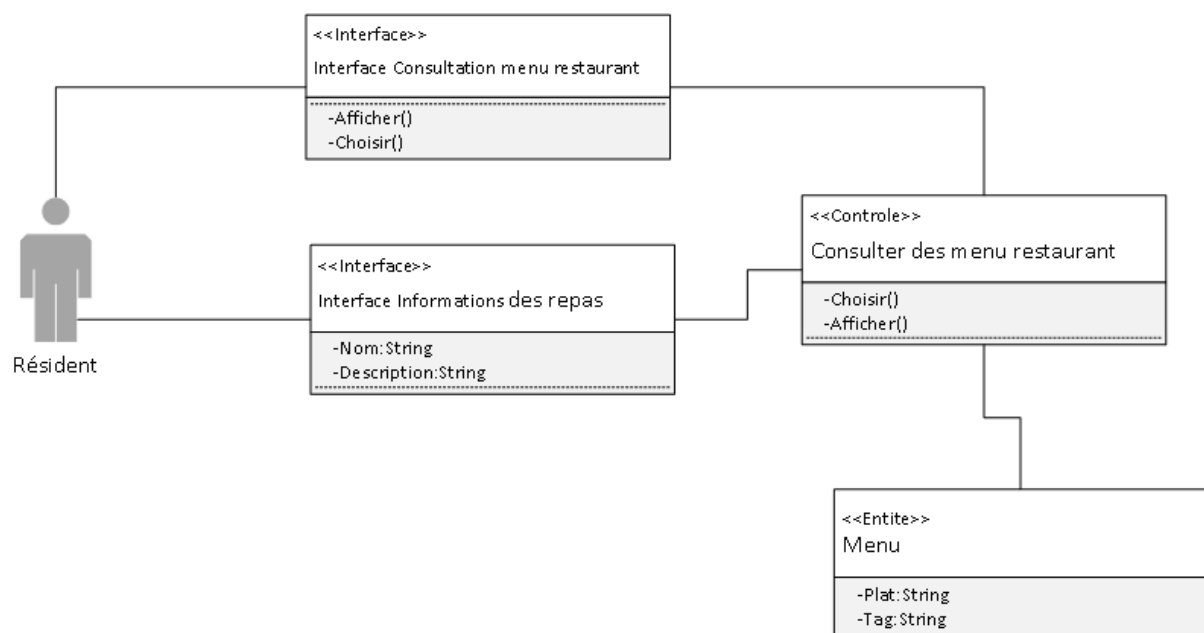


FIG. 4.29 : Diagramme de classes participantes Consulter Menu restaurant

Consulter événements

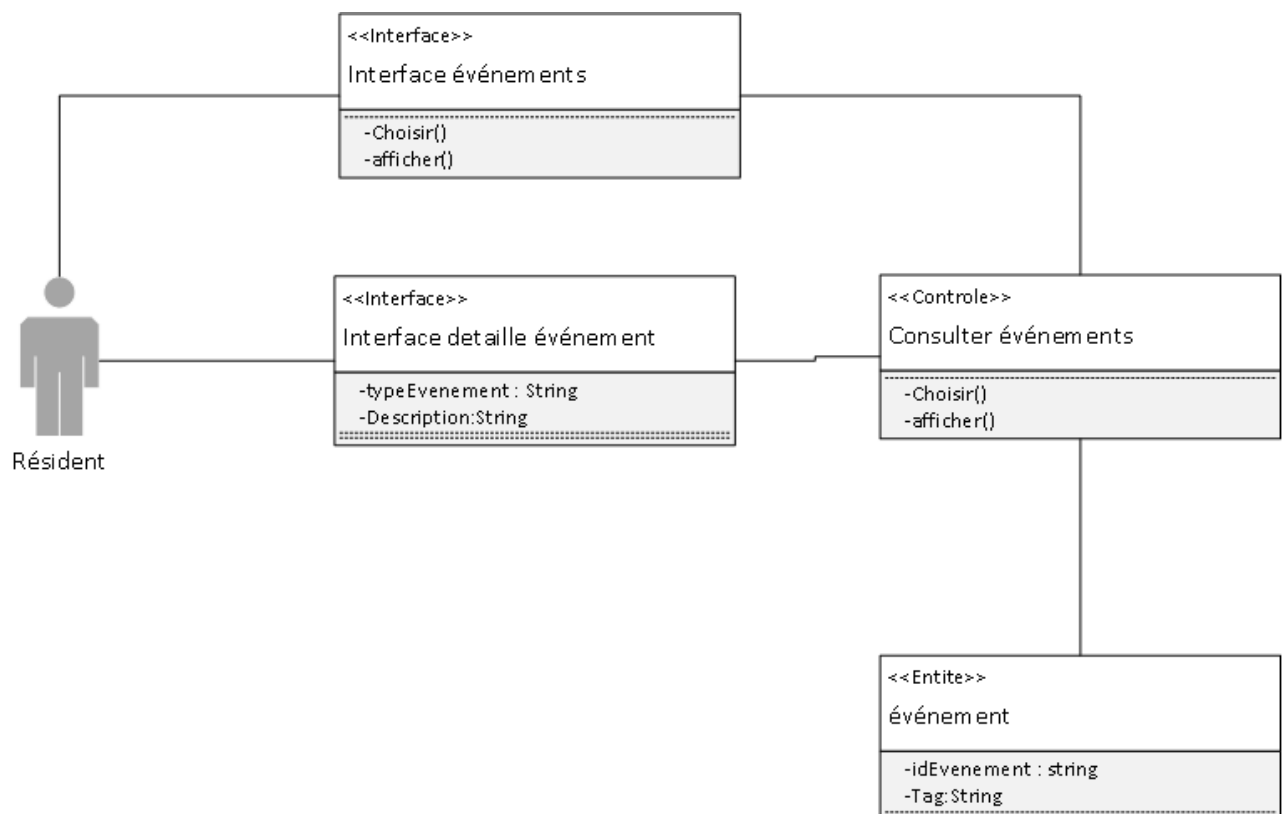


FIG. 4.30 : Diagramme de classes participantes Consulter événements

Gestion des réservations

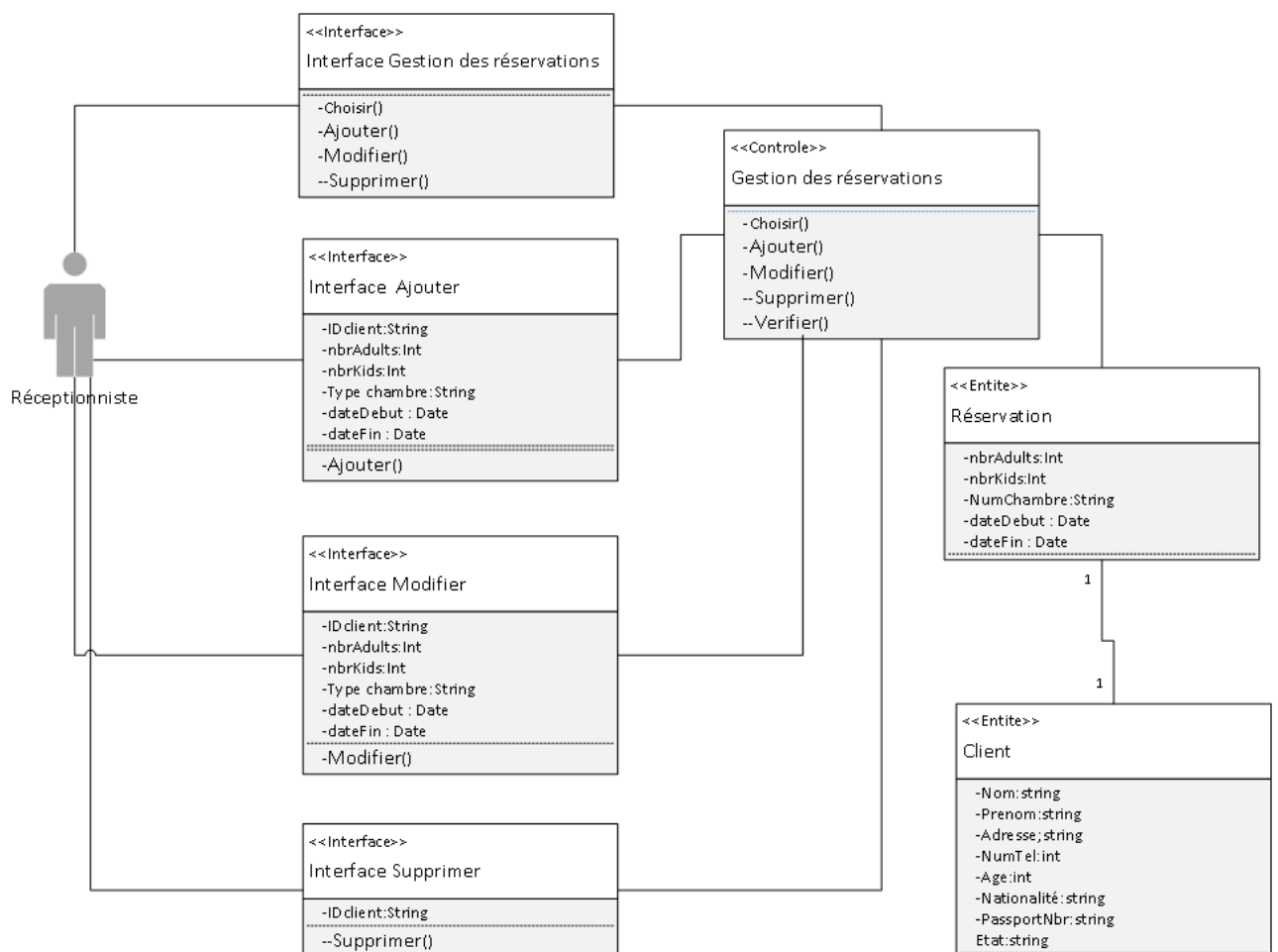


FIG. 4.31 : Diagramme de classes participantes Gestion des réservations

Check-in d'un client

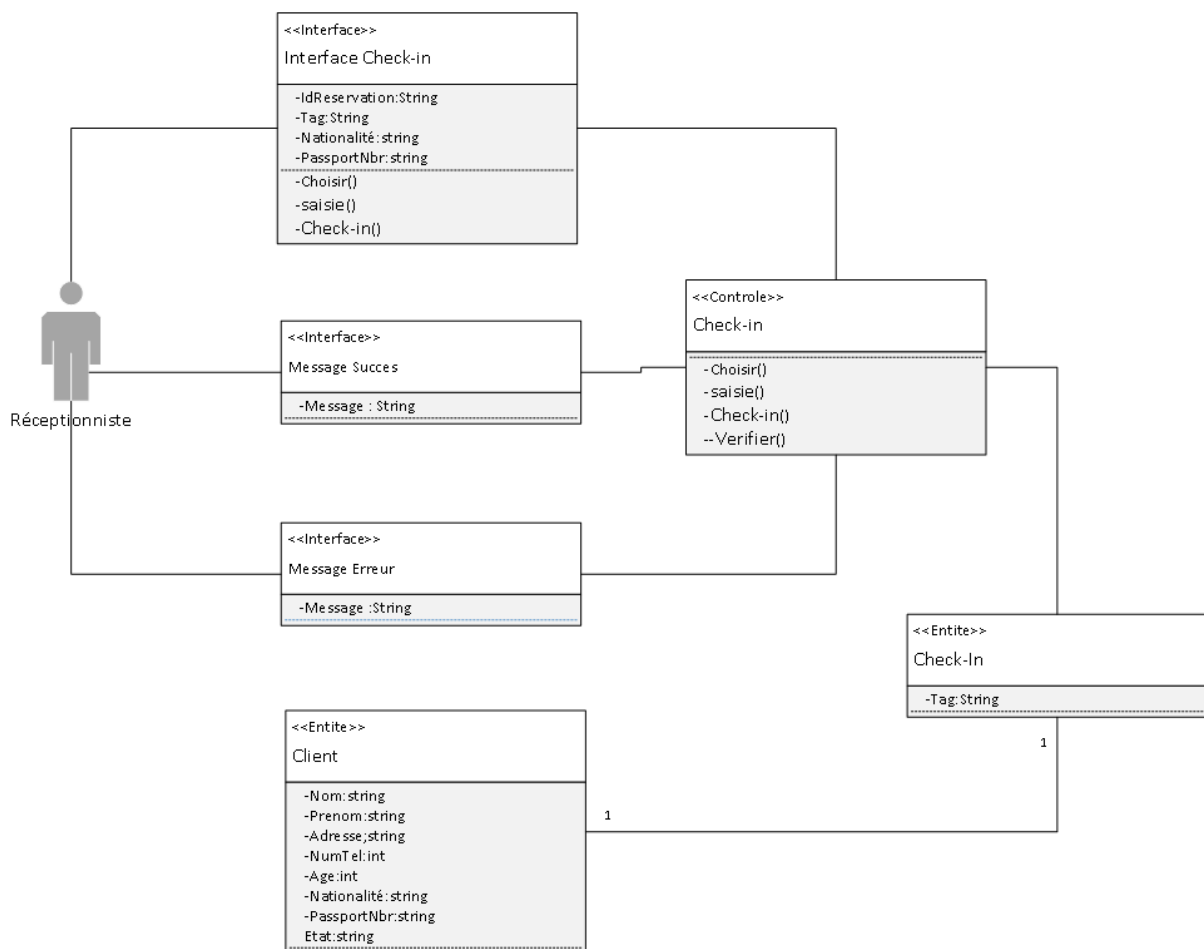


FIG. 4.32 : Diagramme de classes participantes Check-in

Check-out d'un client

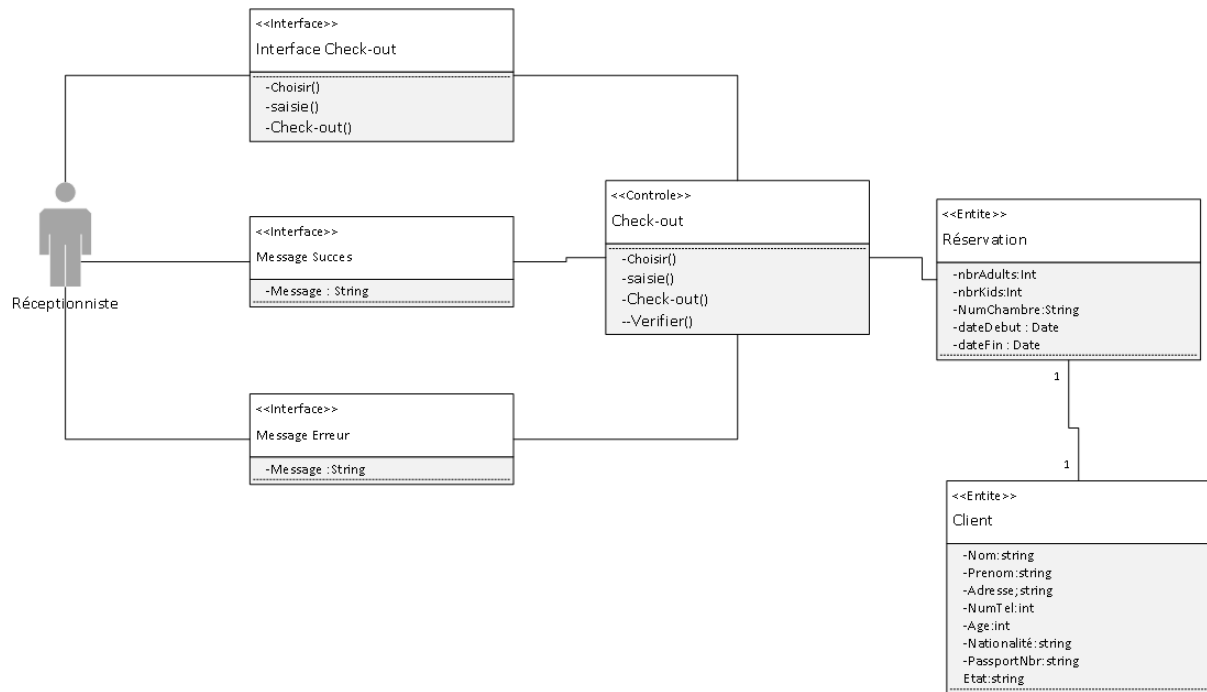


FIG. 4.33 : Diagramme de classes participantes Check-out

Consulter liste résidents

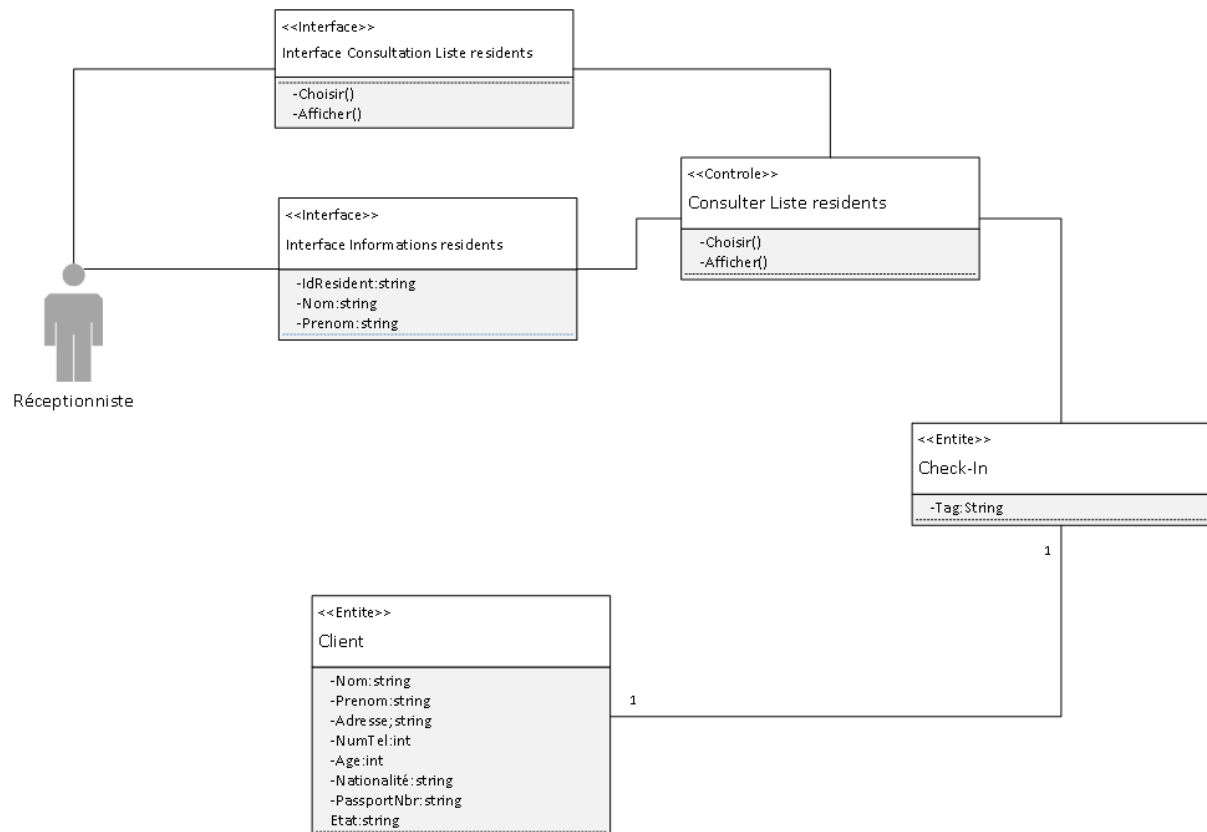


FIG. 4.34 : Diagramme de classes participantes Consulter liste résidents

Consulter les informations Capteurs

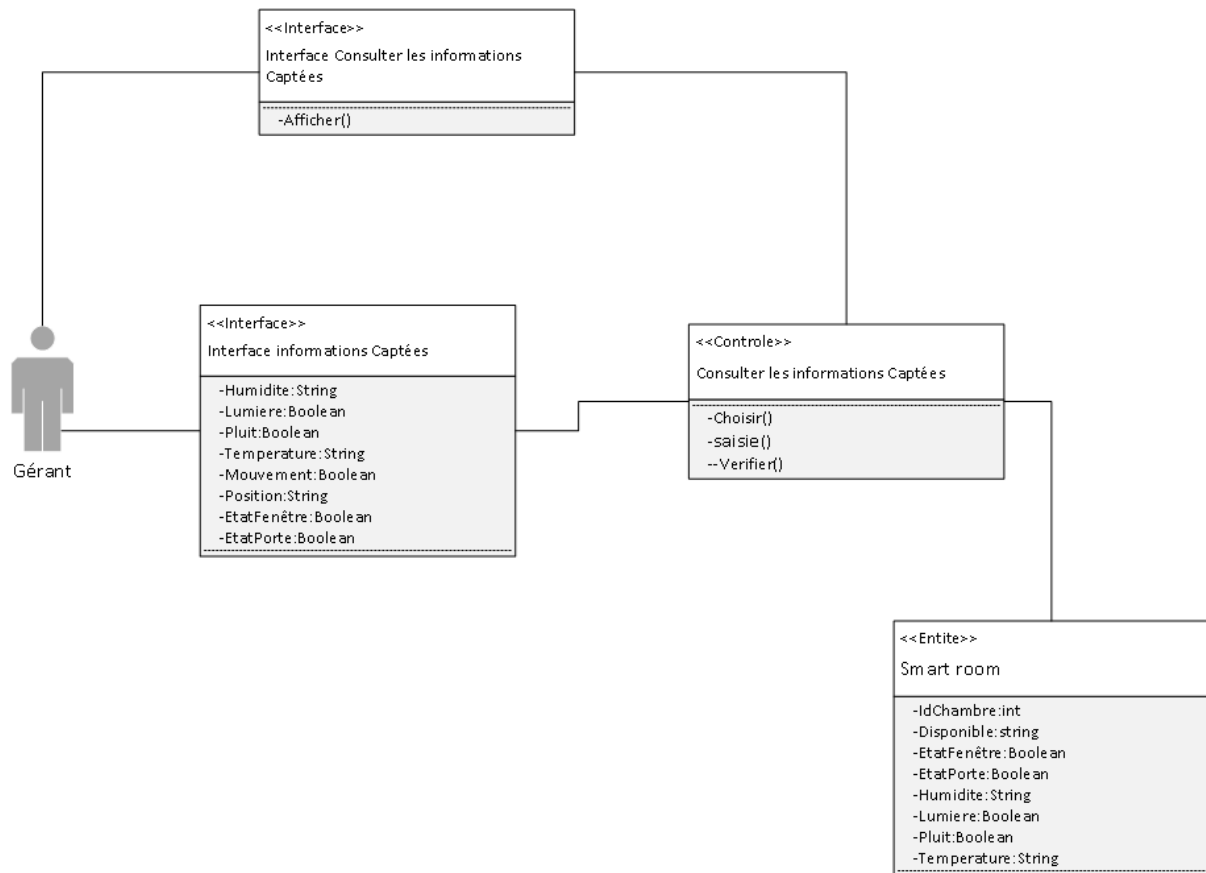


FIG. 4.35 : Diagramme de classes participantes Consulter les informations Capteurs

Gestion du menu restaurant

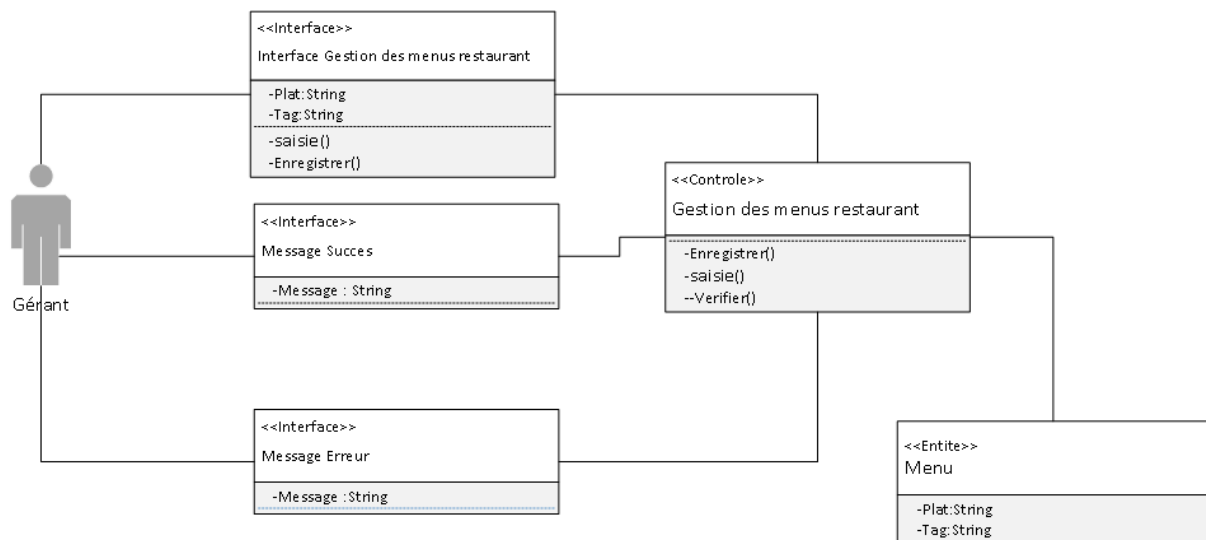


FIG. 4.36 : Diagramme de classes participantes Gestion du menu restaurant

Gestion des événements

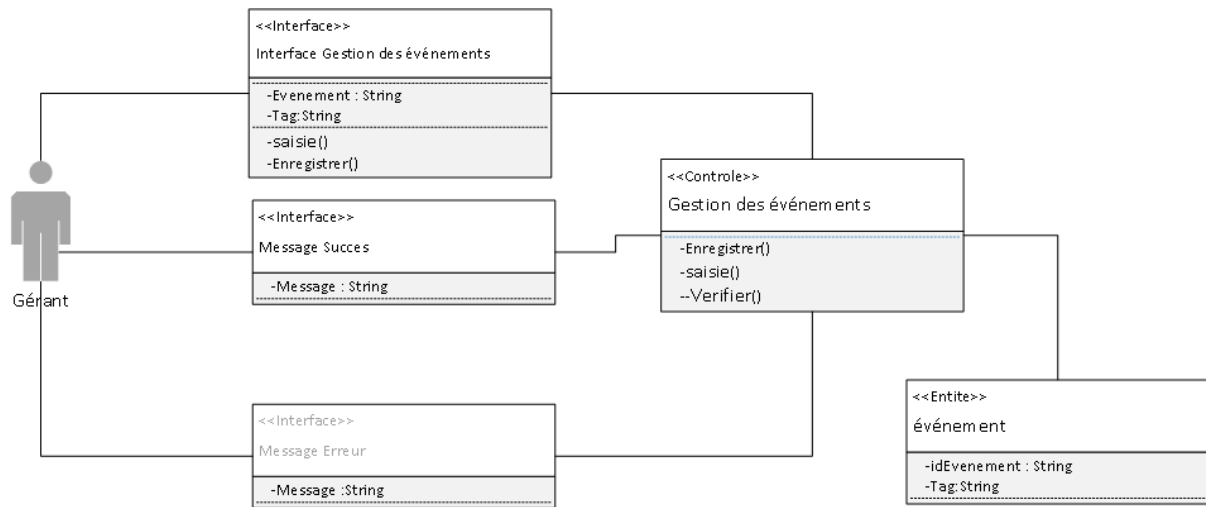


FIG. 4.37 : Diagramme de classes participantes Gestion des événements

Gestion du Room Service

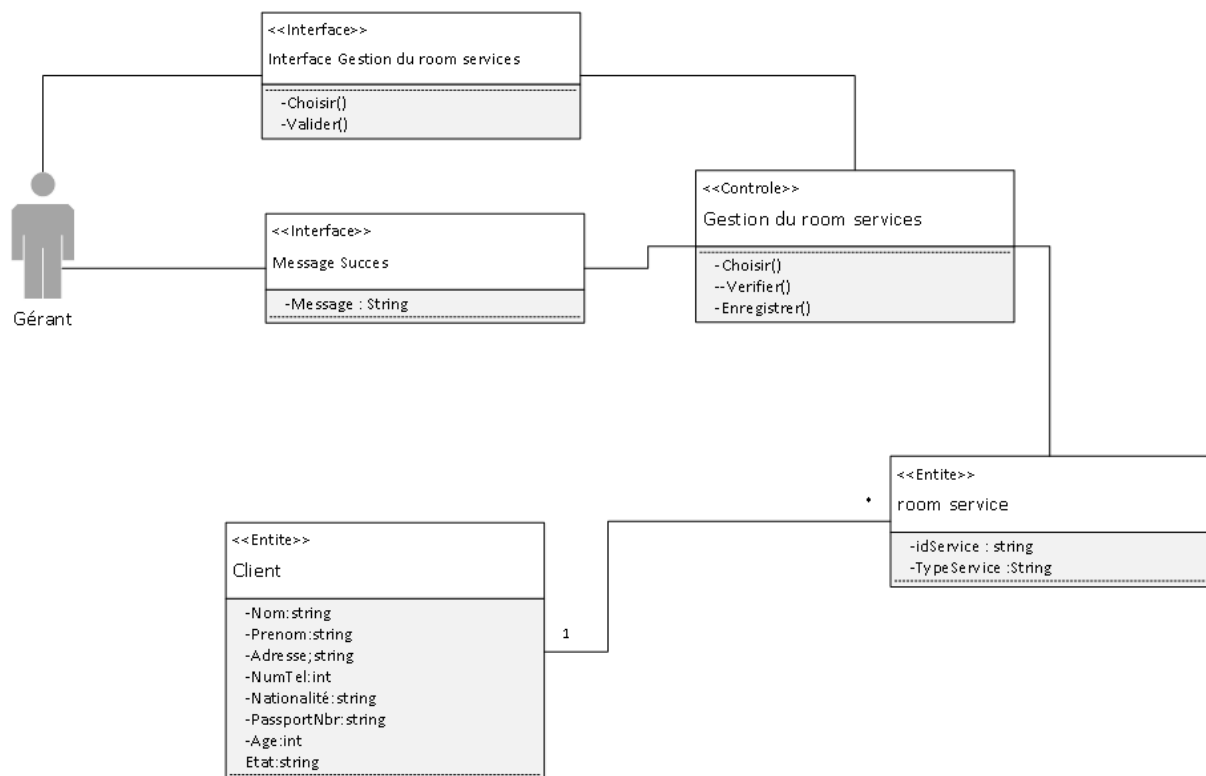


FIG. 4.38 : Diagramme de classes participantes Gestion du Room Service

4.5.2 Diagramme de séquence détaillé

Ce diagramme décrit comment les objets et les instances échangent des messages dans un ordre particulier, c'est un type de diagramme d'interaction, car il décrit comment et dans quel ordre plusieurs objets fonctionnent ensemble. Dans les interactions, les objets sont des lignes de vie [25].

S'authentifier

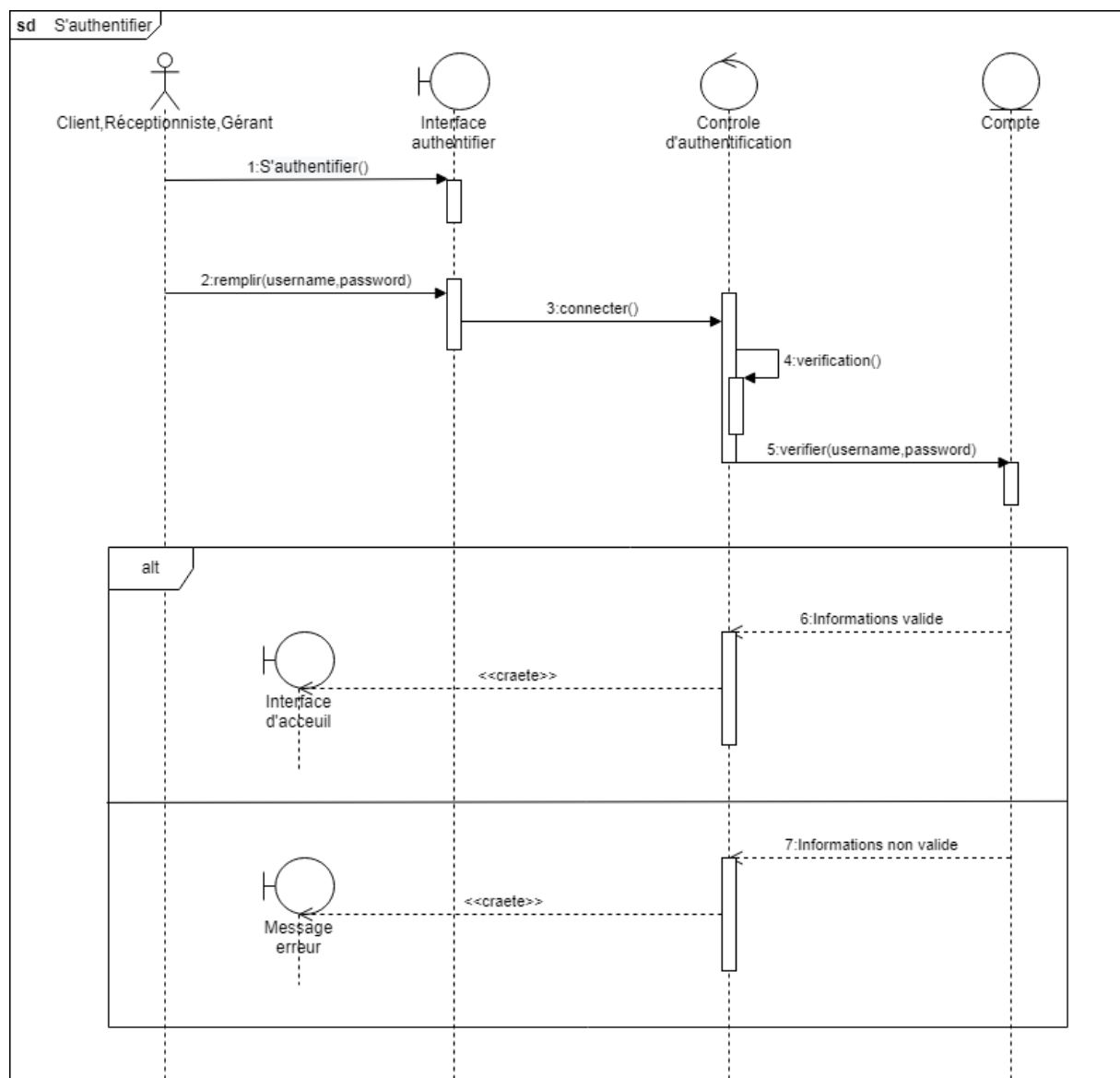


FIG. 4.39 : Diagramme de séquence détaillé S'authentifier

Créer compte

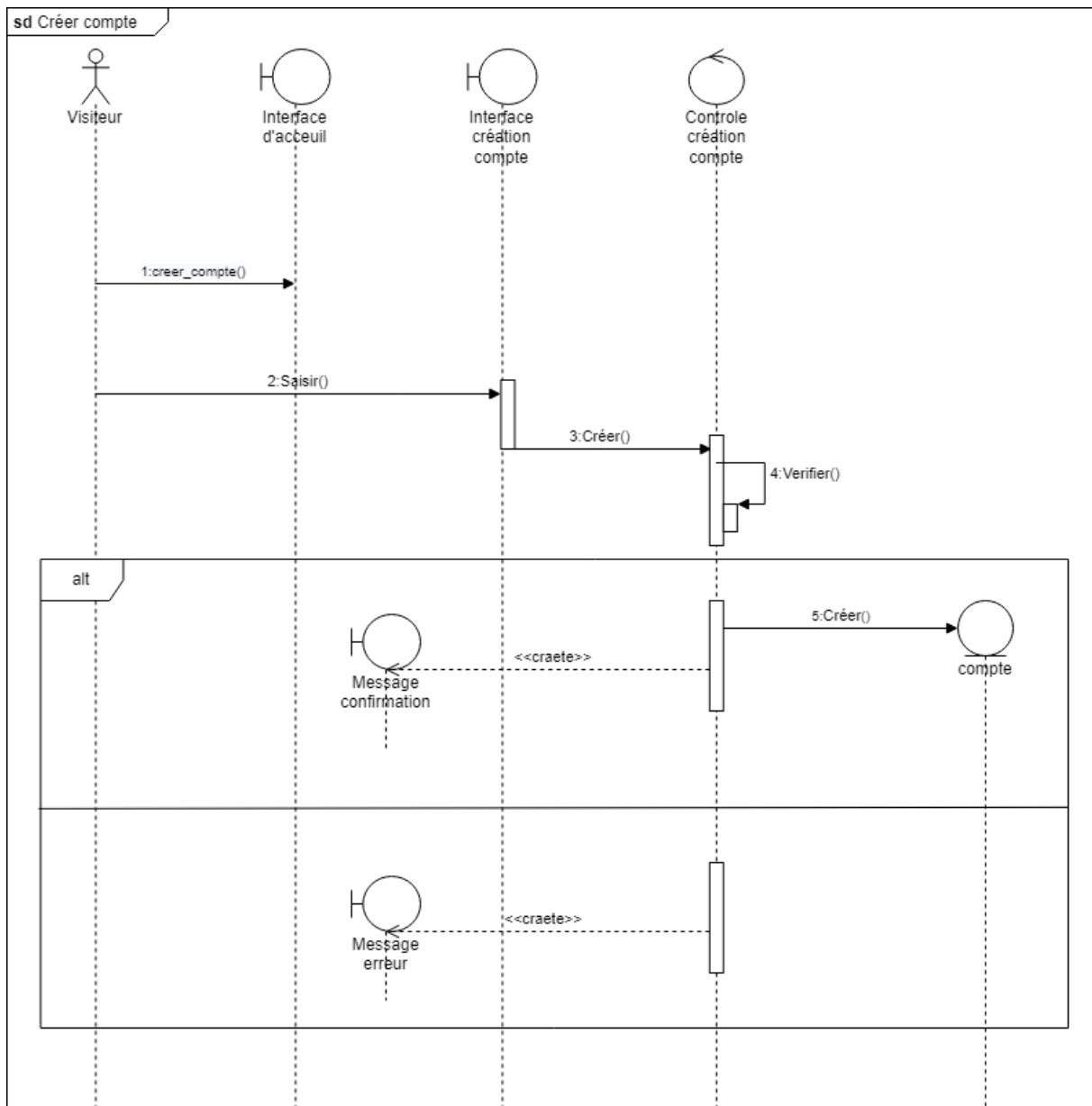


FIG. 4.40 : Diagramme de séquence détaillé Créer compte

Réservation de chambre

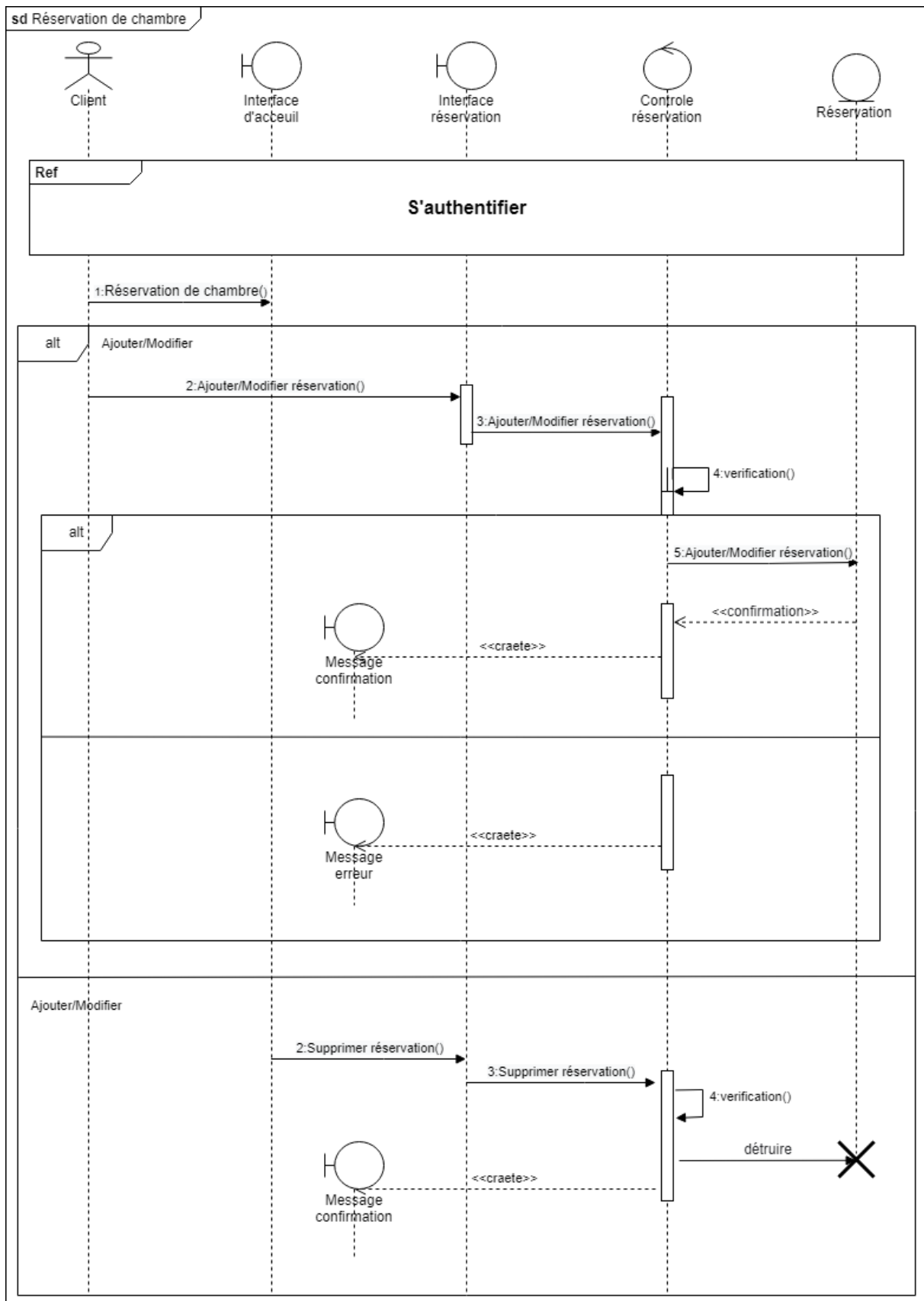


FIG. 4.41 : Diagramme de séquence détaillé Réservation de chambre

Mettre à jour informations compte

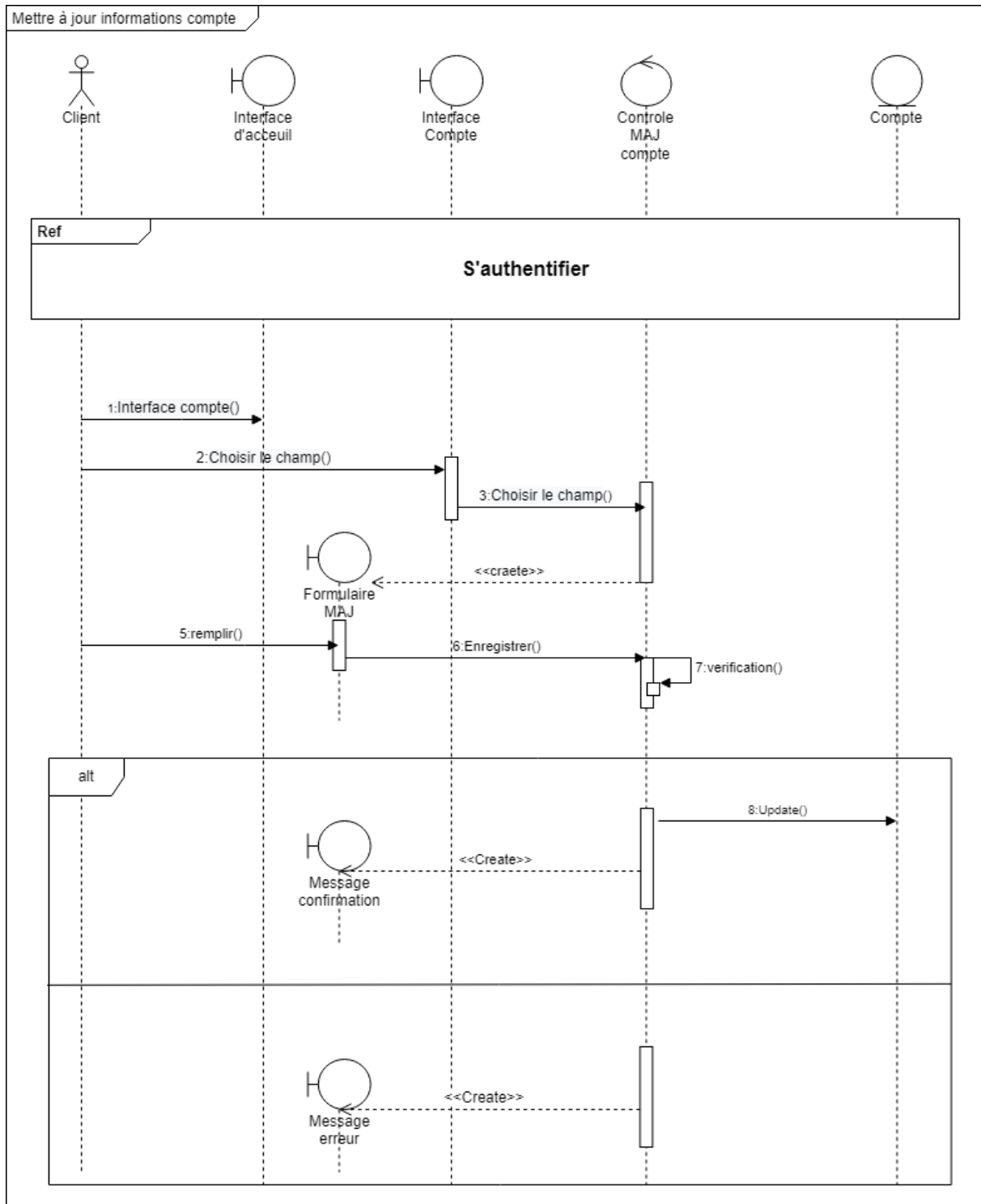


FIG. 4.42 : Diagramme de séquence détaillé Mettre à jour informations compte

Gérer préférences

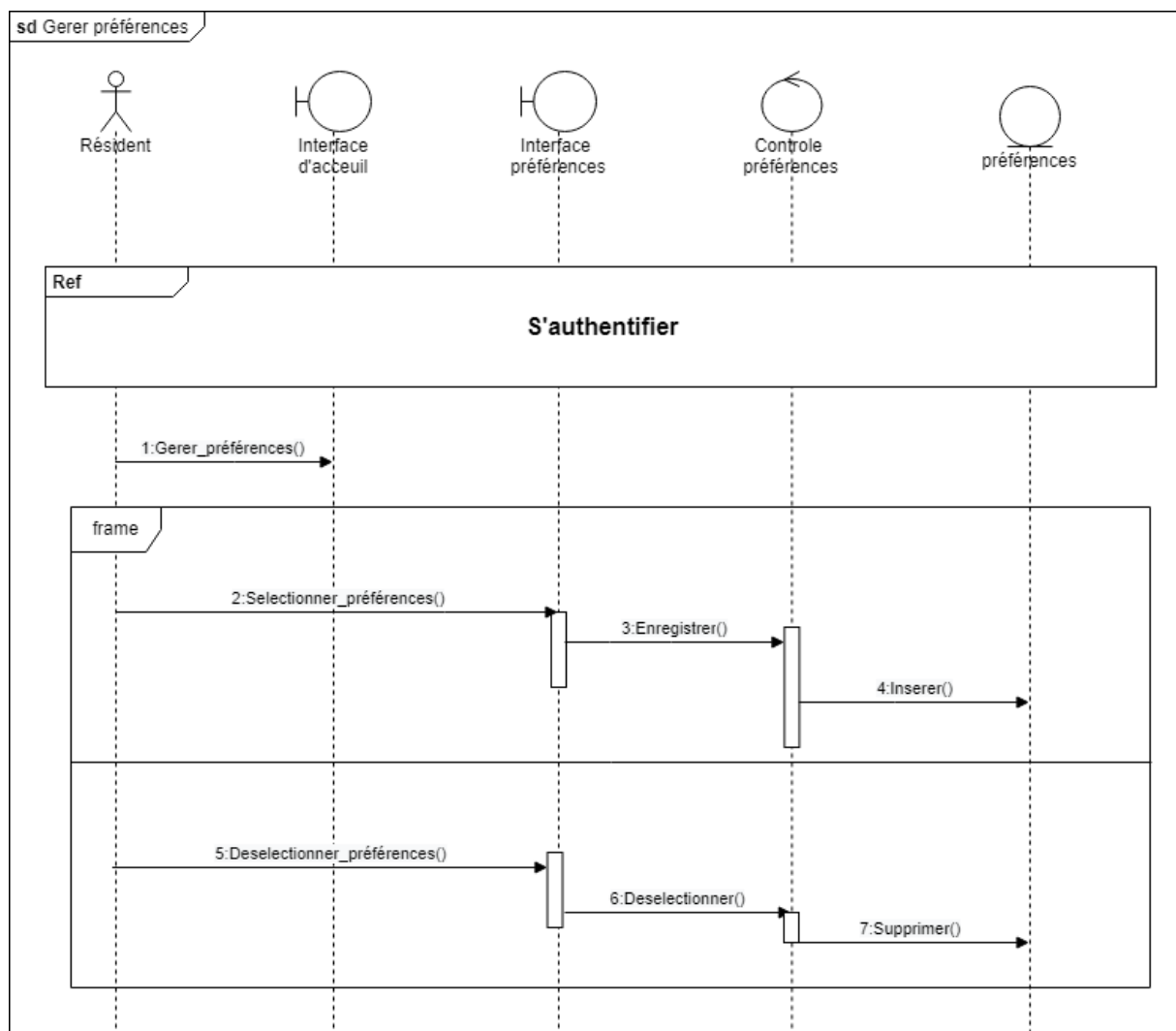


FIG. 4.43 : Diagramme de séquence détaillé Gérer préférences

Valider code séjour

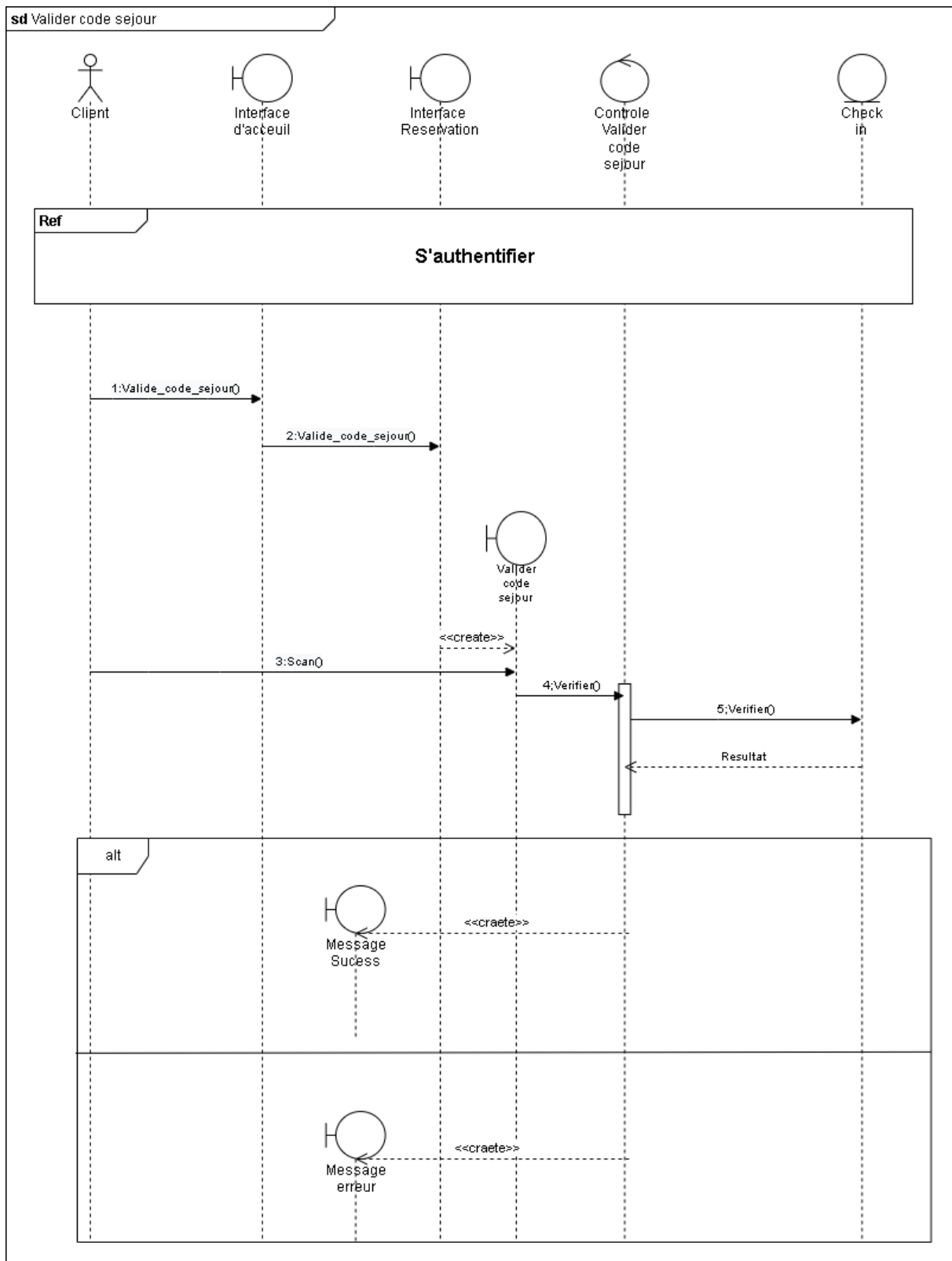


FIG. 4.44 : Diagramme de séquence détaillé Valider code séjour

Consulter des centres d'intérêt près de l'hôtel

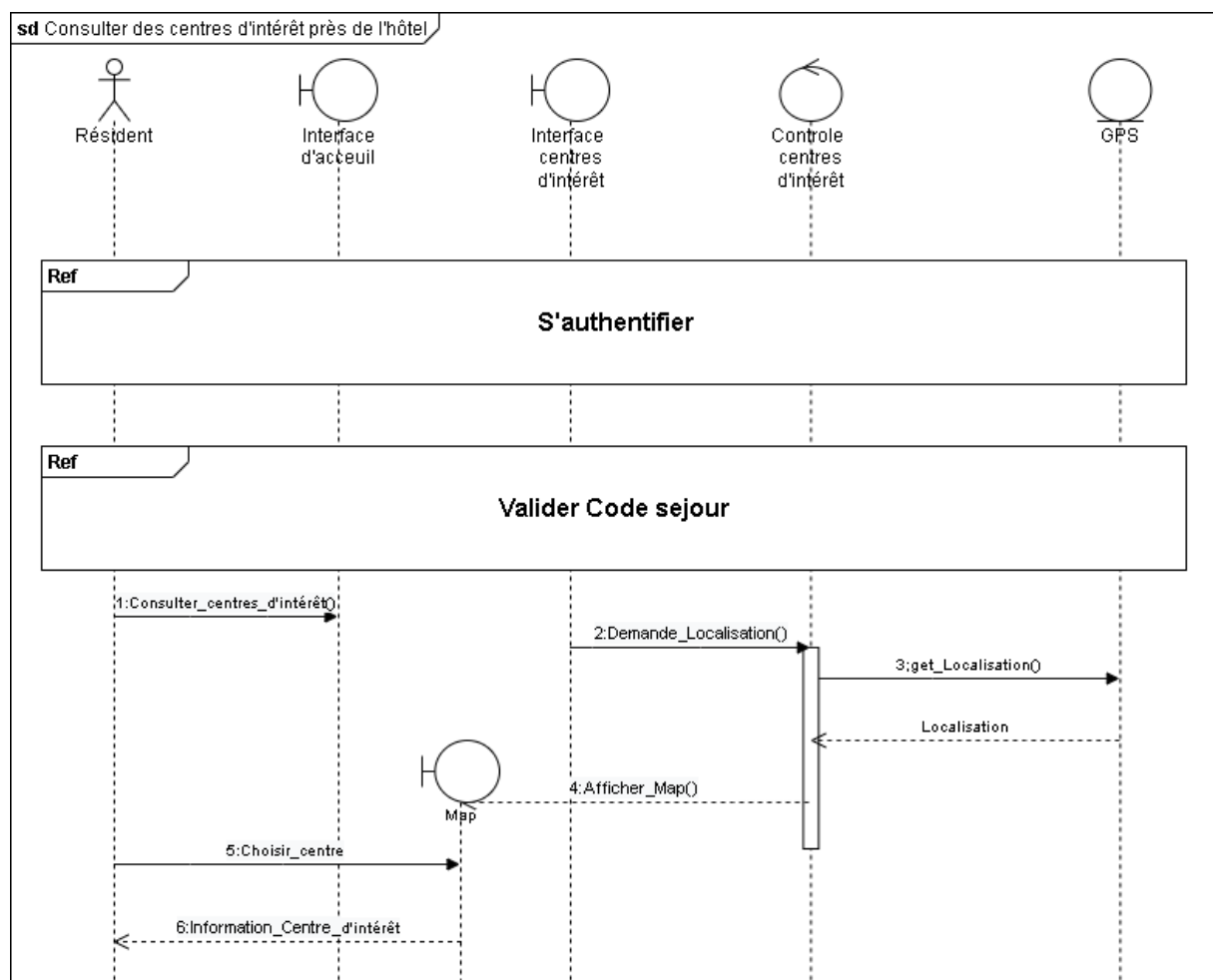


FIG. 4.45 : Diagramme de séquence détaillé Consulter des centres d'intérêt près de l'hôtel

Gérer Smart Room

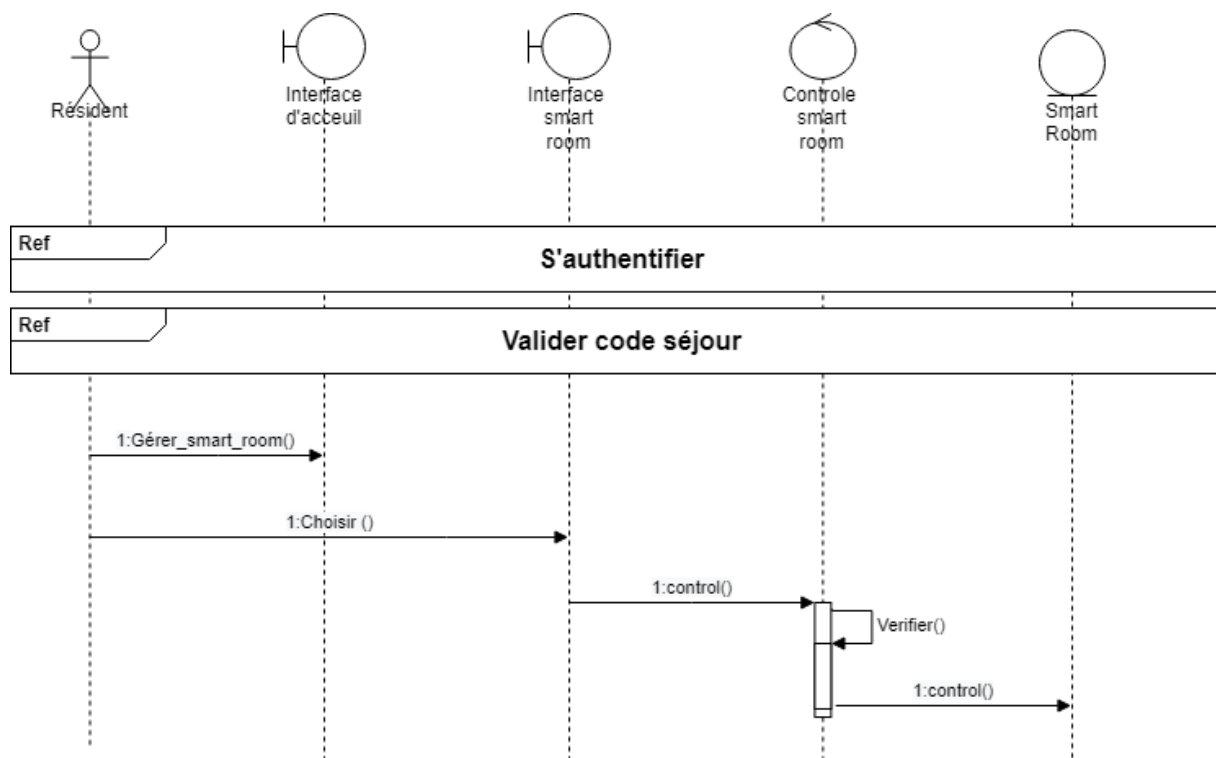


FIG. 4.46 : Diagramme de séquence détaillé Gérer Smart Room

Demander Room service

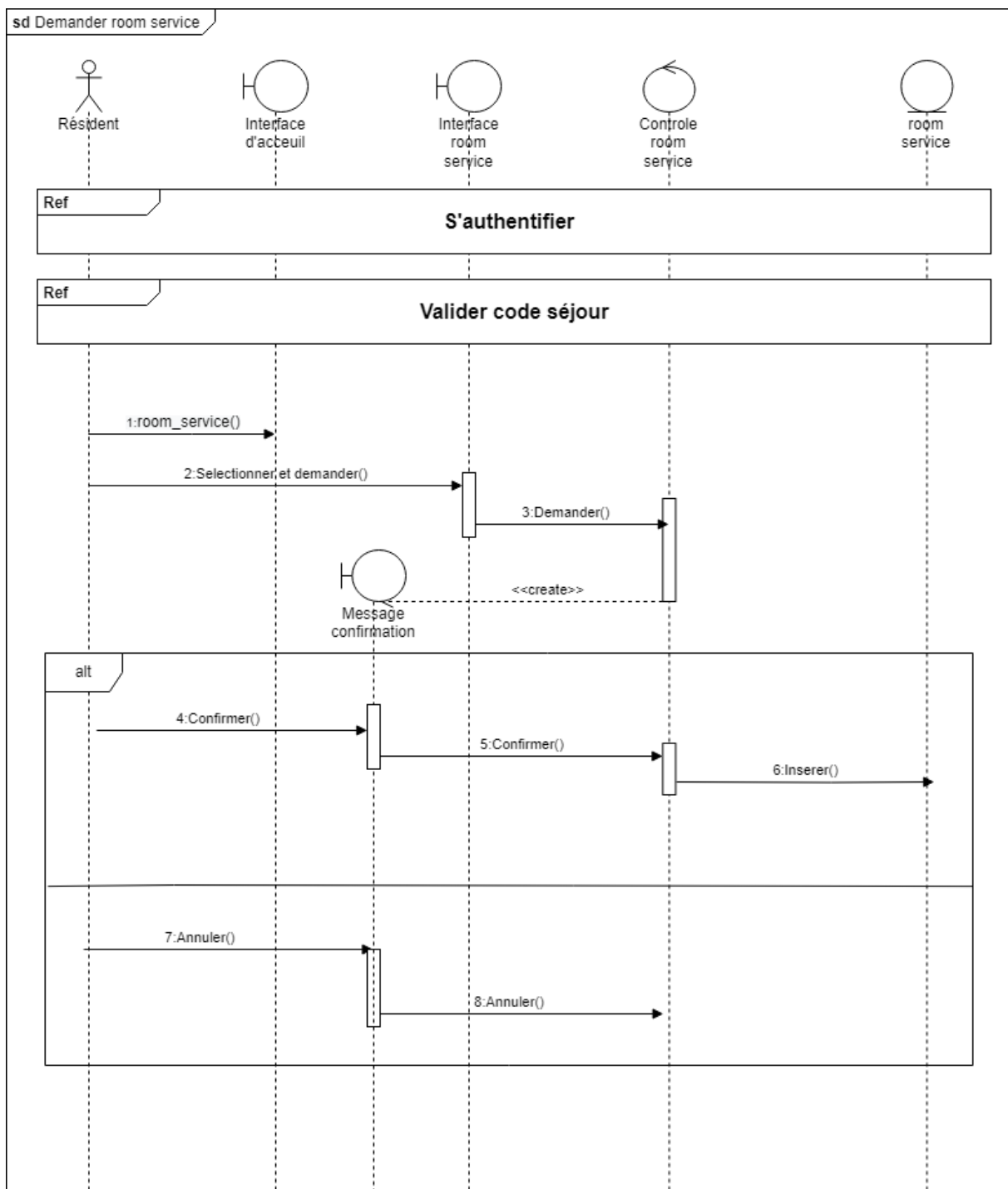


FIG. 4.47 : Diagramme de séquence détaillé Demander Room service

Consulter Menu restaurant

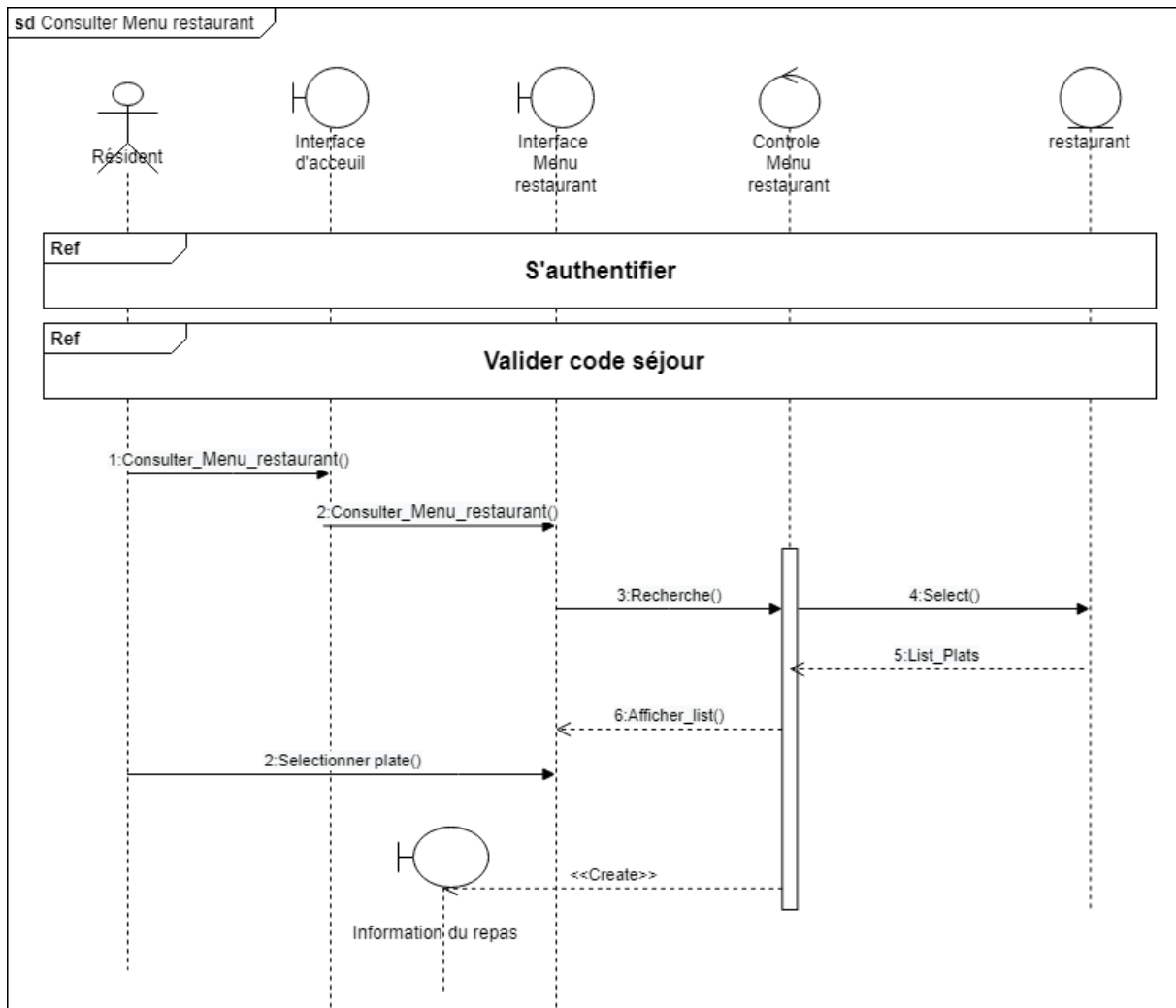
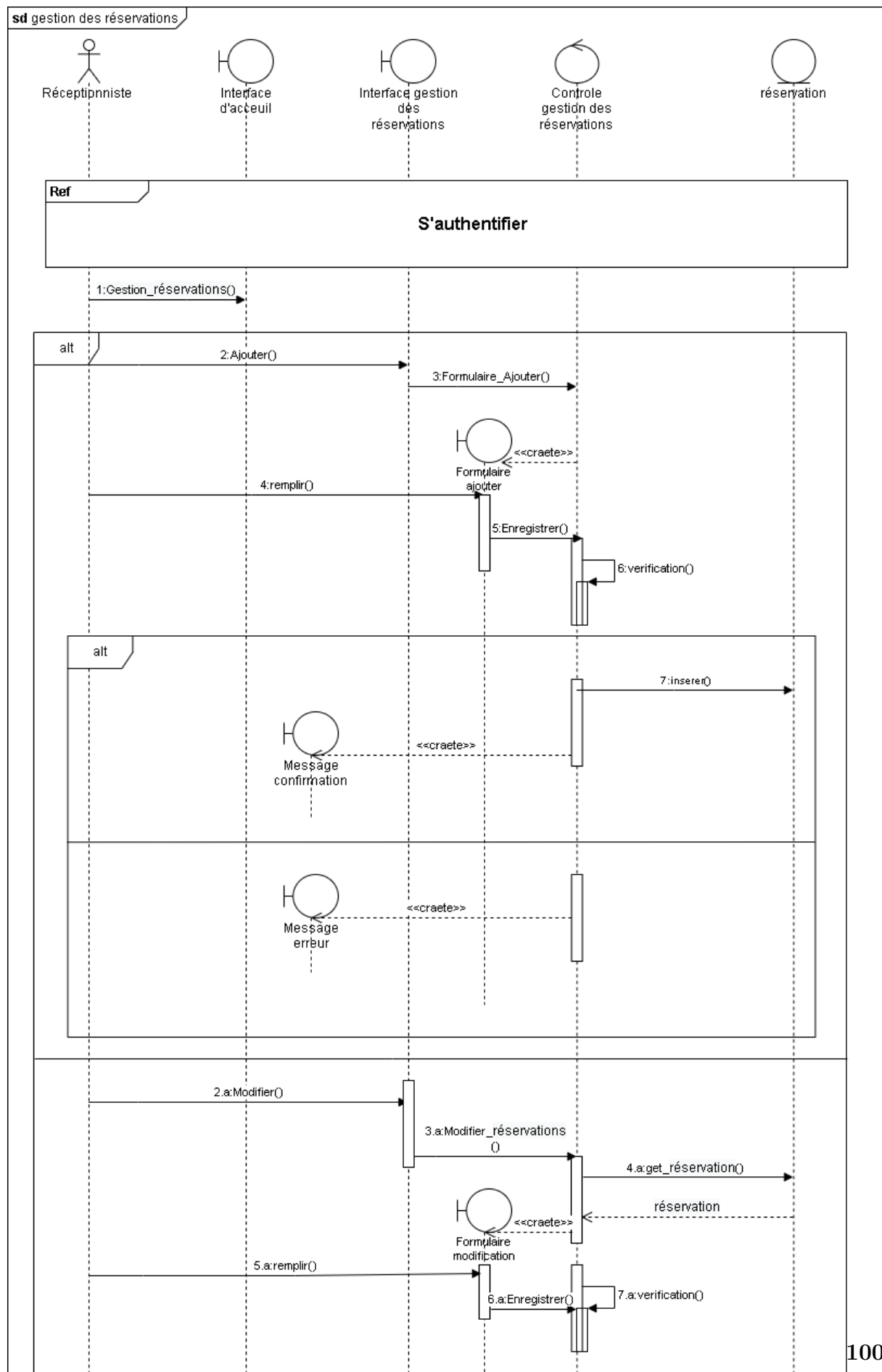


FIG. 4.48 : Diagramme de séquence détaillé Consulter Menu restaurant

Gestion des réservations



Check-in d'un client

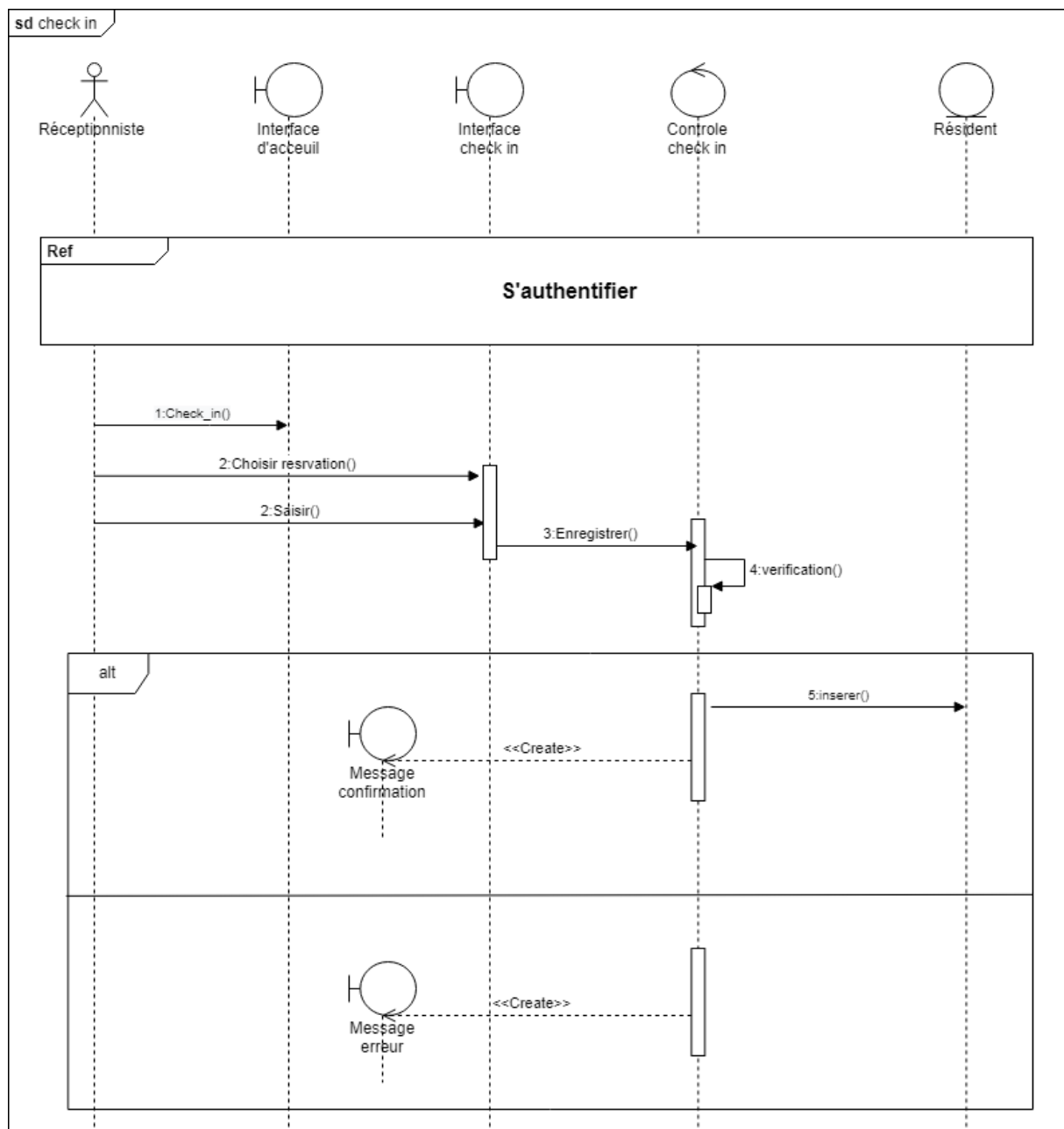


FIG. 4.50 : Diagramme de séquence détaillé Check-in

Check-out d'un client

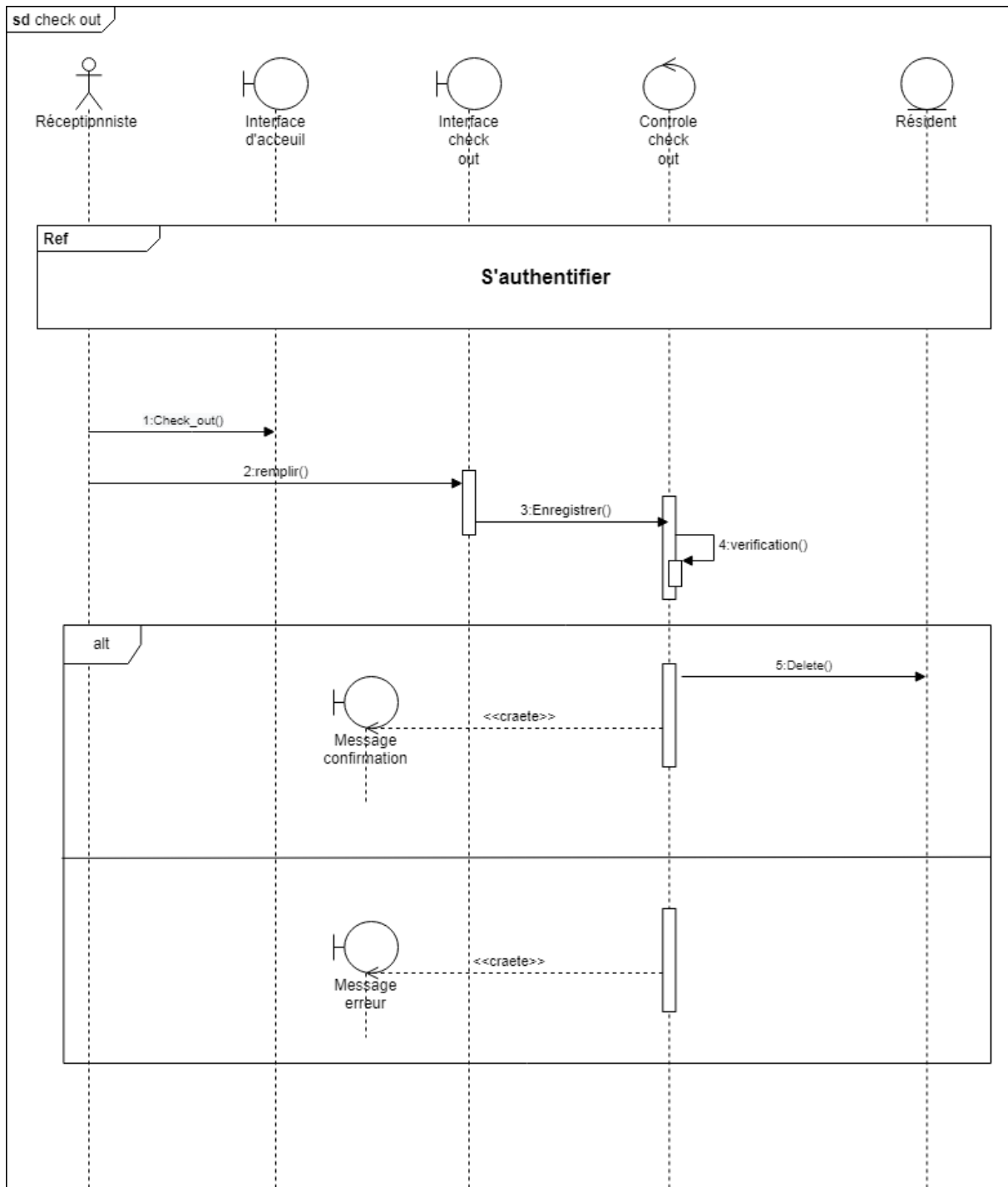


FIG. 4.51 : Diagramme de séquence détaillé Check-out

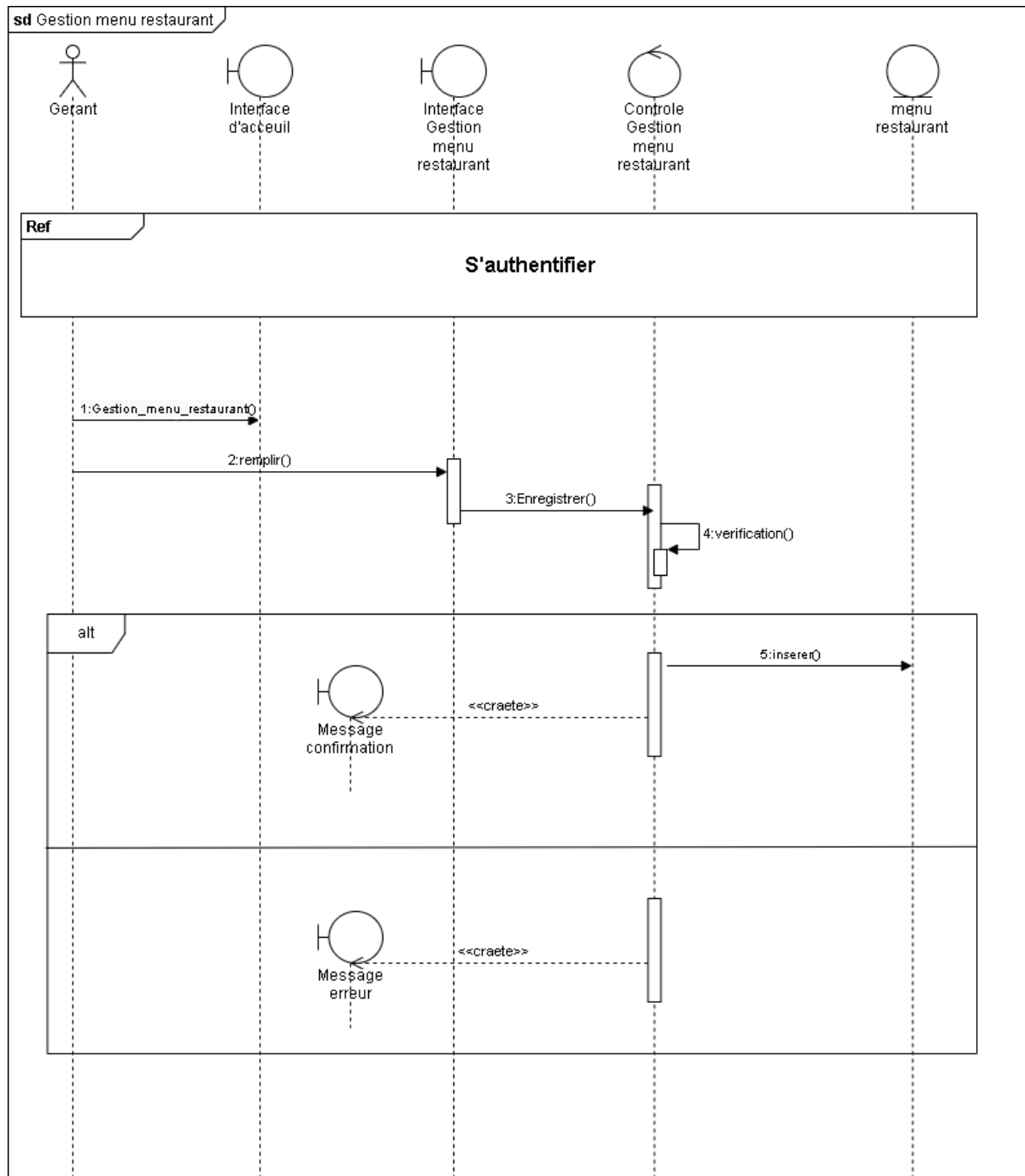


FIG. 4.52 : Diagramme de séquence détaillé Gestion du menu restaurant

4.5.3 Diagramme de classes conception

Un diagramme de classes fournit une vue globale d'un système en présentant ses classes, interfaces et collaborations, et les relations entre elles. Les diagrammes de classes sont statiques : ils affichent ce qui interagit mais pas ce qui se passe pendant l'interaction [21].

Note : Notre base de données est une base NoSql, donc qui n'est pas relationnel.

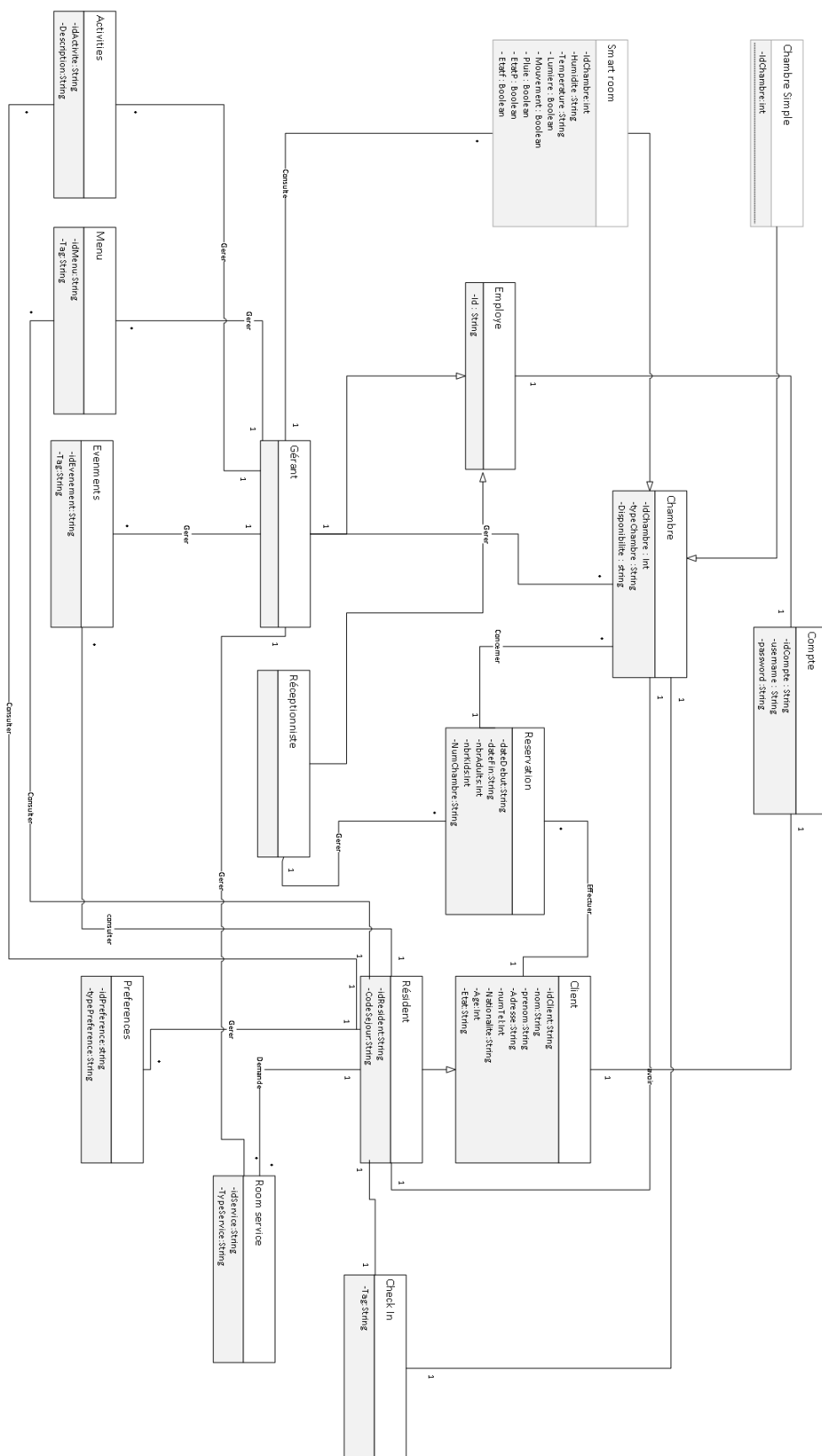


FIG. 4.53 : Diagramme de classes conception

On a trouvé des difficultés à faire ce diagramme à cause de la base de données qui n'est pas relationnel. Le diagramme de la sous-section qui suit est plus explicite.

4.5.4 Modèle de base de donnée

Voici notre modèle sous forme de modèle de données en document [67] :

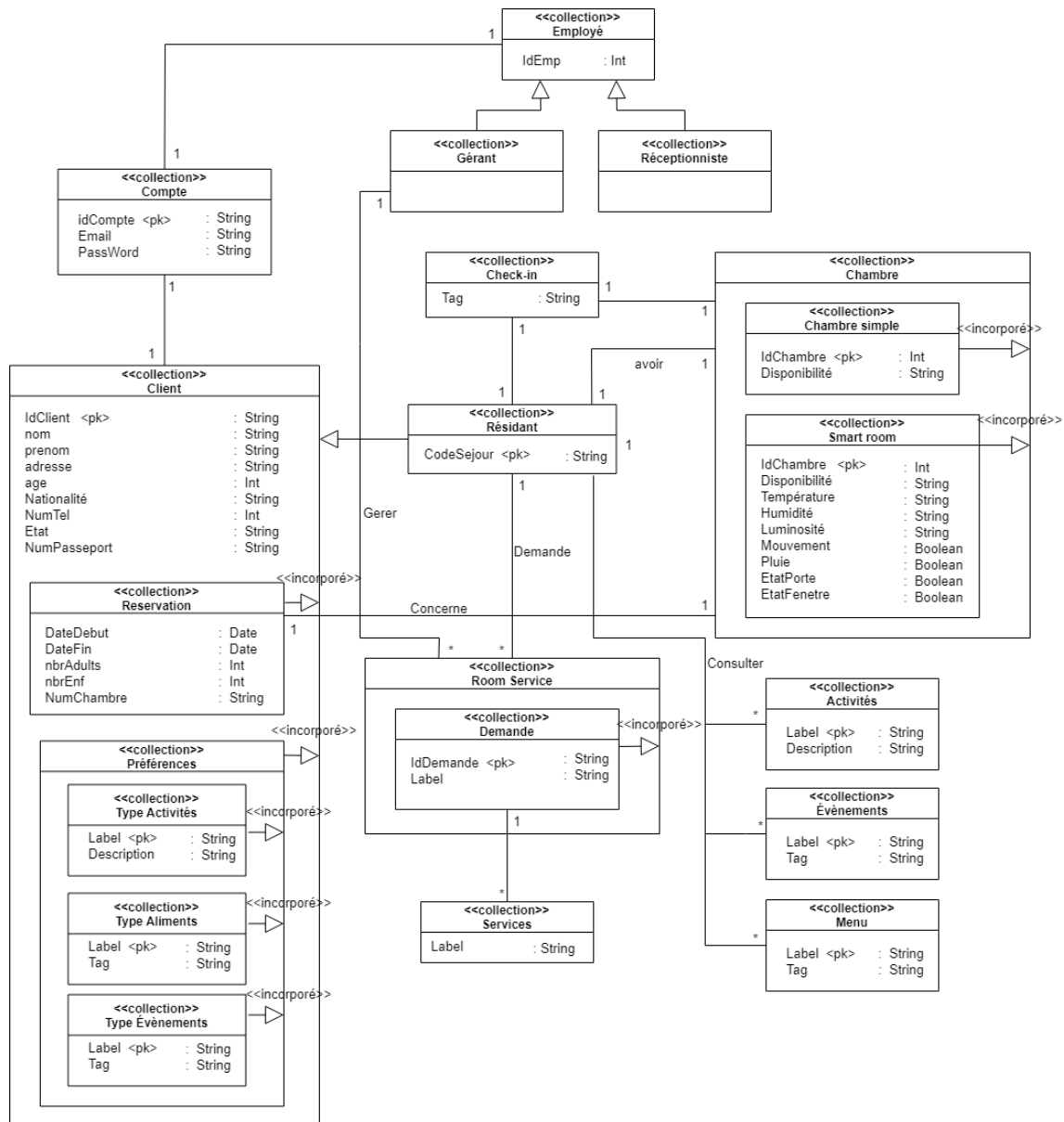


FIG. 4.54 : Modèle de base de donnée

Firestore est une base de données NoSQL, hébergée sur le cloud de Google et orientée documents. Ce type de base de données est principalement structuré en trois éléments [20] :

- Donnée : Elle représente, la donnée brute que vous souhaitez sauvegarder, comme par exemple un entier ou une chaîne de caractères etc.

- Document : Les données brutes ne peuvent pas être sauvegardées aléatoirement et n'importe où. Elles sont donc nécessairement jointes à un document, sous forme de propriétés/champs.
- Collection : Permet d'organiser sous forme de liste les documents. En effet, une collection va contenir un ou plusieurs documents.

4.6 Conception de la maquette de simulation

4.6.1 Définition des besoins

De base, notre maquette de simulation a pour but la capture des informations contextuelles, ensuite les exploiter et les utiliser pour satisfaire l'utilisateur de notre application mobile.

Les informations contextuelles qu'on vise à capturer sont la température, l'humidité, la condition météo (pluie), la luminosité, le mouvement et la position grâce à l'identification par radiofréquence (RFID).

La température, l'humidité, la condition météo et la luminosité vont nous servir pour une utilisation de notification en fonction du contexte, en plus on a pour objectif de les utiliser dans une fonction de classification pour choisir les événements que l'hôtel réalisera dans la journée en fonction du contexte du jour, qui se compose de différentes informations contextuelles. La classification sera faite à l'aide de la **classification naïve bayésienne**, une méthode d'apprentissage automatique.

Le mouvement et la position RFID vont aussi nous servir pour une utilisation de notification en fonction de la position, par exemple à la détection d'un Résident dans un emplacement de l'hôtel, le système va automatiquement lui envoyer une notification pour qu'il consulte les informations relatives à cet emplacement.

En combinant le mouvement et la position on peut avoir des notifications de sécurité. par exemple s'il y a du mouvement dans la chambre et le résident est détecté dans un autre emplacement de l'hôtel.

En plus des capteurs, on compte ajouter des actionneurs pour simuler une chambre intelligente, l'utilisation de servos pour l'automatisation de la porte et la fenêtre, un écran LCD pour l'affichage des informations utiles, et des LEDs.

4.6.2 Modèle de communication

Notre système est un système IoT qui se compose de plusieurs objets et périphériques, les appareils mobiles des utilisateurs de l'application mobile, notre application de gestion et notre système de capture de contexte (maquette). On a choisi le modèle de communication **Périphérique à cloud** de l'IoT. La communication entre les différents composants de notre système va se faire à l'aide du WiFi via un service web de base de

données (DBaaS), probablement Firebase de Google. Donc notre maquette contiendra un module de communication WiFi.

4.6.3 Schéma de création

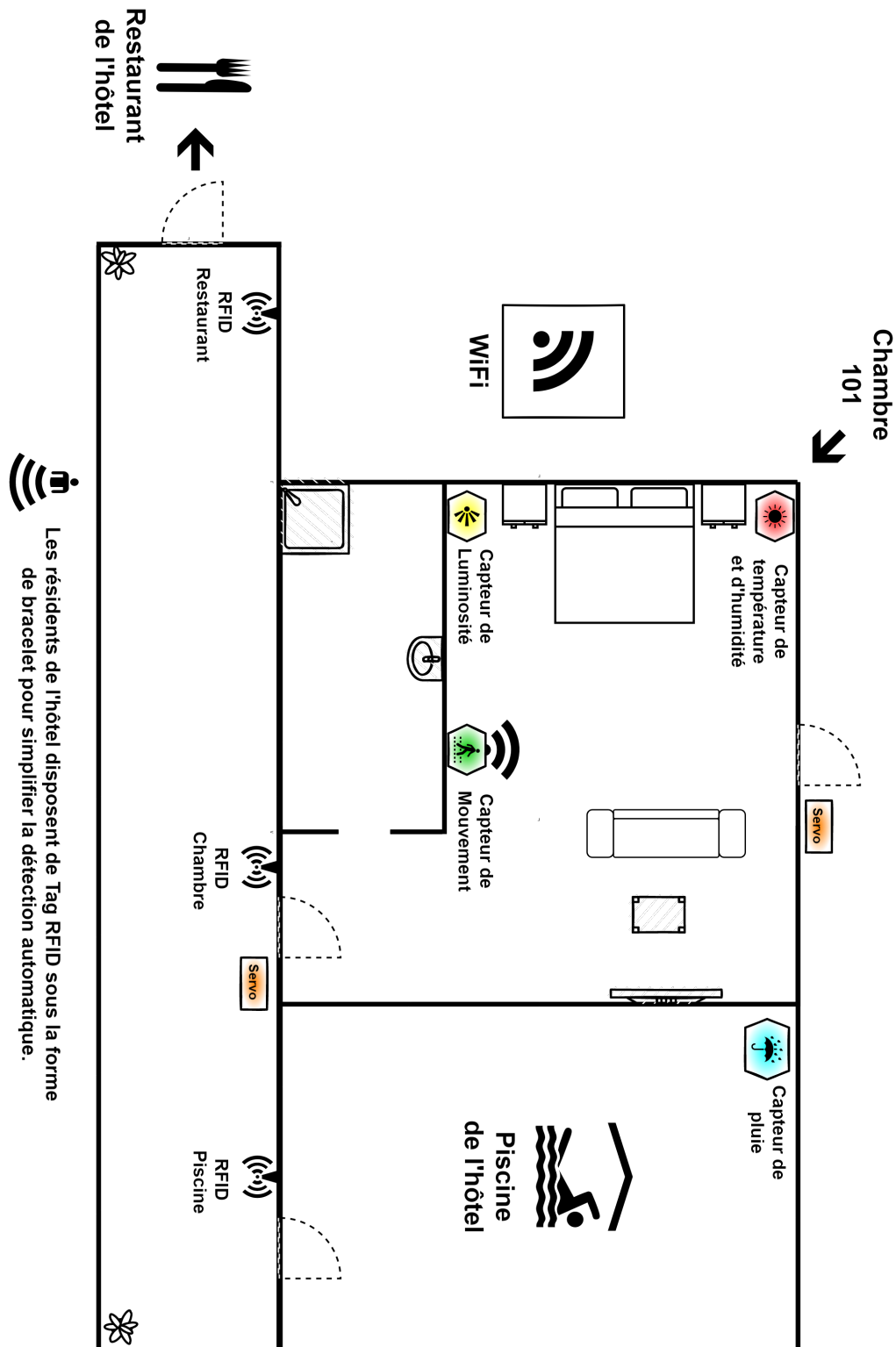


FIG. 4.55 : Schéma de création du système de capteurs

4.6.4 Conclusion

Dans ce chapitre nous avons proposé notre système. On a procédé en modélisant ce dernier tout en décrivant ses composants, son fonctionnement et sa structuration en utilisant les diagrammes de classes participants, ainsi que les diagrammes d'interaction et le diagramme de classe conception. Le chapitre suivant est consacré à sa réalisation physique.

Chapitre 5

Implémentation et Réalisation

5.1 Introduction

Après avoir présenté au chapitre précédent les diverses phases d'analyse et de conception, nous allons présenter dans ce dernier chapitre l'environnement de développement qui contient les différents langages et technologies qui ont été utilisées pour développer notre application, ensuite nous décrivons les composants matériels spécifiés dans le chapitre précédent tout en montrant leur branchement et pour finir, nous présenterons l'application et ses fonctionnalités à travers ses différentes interfaces.

5.2 Langages et outils utilisés

5.2.1 Langage Java

Java est un langage de programmation à usage général conçu par Sun Microsystems et lancé en 1995, le langage est influencée par C++. La motivation principale de Java était de créer un langage de programmation permettant à ses utilisateurs "d'écrire une seule fois, exécuter partout".

Java est le 3ème langage le plus utilisé ces 5 dernières années sur GitHub. Plusieurs plates-formes Java sont disponibles selon le type d'application pour lequel Java sera utilisé pour le développement [31].

Nous avons utilisé la plateforme **Android Studio** pour développer nos applications.



FIG. 5.1 : Logo Langage Java

5.2.2 Android Studio

Nous avons utilisé Android Studio pour développer l'application mobile Android de notre système, et pour développer l'application de gestion de l'hôtel sous forme d'application pour tablette. Les applications Android développées sont basées sur les langues Java et XML.



FIG. 5.2 : Logo Android Studio

5.2.3 Arduino IDE

Le logiciel utilisé pour développer le programme pour Arduino est communément connu sous le nom d'Arduino IDE. Le code du programme écrit pour Arduino est nommé sketch. Cet IDE contient ce qui suit [40] :

Editeur de texte

C'est ici que le code est rédigé en utilisant une version simplifiée du langage de programmation C++.

Zone de message

Il affiche les erreurs et donne aussi des commentaires sur la sauvegarde et l'exportation de code.

Console

La console affiche le texte produit par l'environnement Arduino, avec des messages d'erreur complets, des informations d'erreur et d'autres informations.

Barre d'outil

Cette barre d'outils contient une variété de boutons comme Verify, Upload, New, Open, Save et Serial Monitor.



FIG. 5.3 : Logo Arduino IDE

5.2.4 Firebase

Firebase est une base de données NoSQL et considérée comme une plateforme d'application Web afin de créer des applications de haute qualité. Il stocke les données au format JavaScript Object Notation (JSON), et il n'utilise aucune requête pour insérer, mettre à jour ou supprimer des données. Il s'agit du backend d'un système qui est utilisé comme une base de données pour stocker des données. Firebase offre les services suivants [34] :

Firebase Analytics

Il offre un aperçu de l'usage des applications, Qui permet également de mesurer l'engagement des utilisateurs. Cette fonctionnalité unique permet au développeur d'application de comprendre la façon dont les utilisateurs utilisent l'application.

Firebase Cloud Messaging

Firebase Cloud Messaging est une solution multiplate-forme pour les messages et notifications pour Android, les applications Web et IOS.

Firebase Auth

Firebase Auth est un service qui permet d'authentifier les utilisateurs en utilisant uniquement du code côté client. Il comprend aussi un système de gestion des utilisateurs qui permet aux développeurs d'activer l'authentification des utilisateurs avec un e-mail et un mot de passe stockés dans Firebase.

Real-time Database

Real-time Database est une base de données en temps réel. Un API est fournie au développeur de l'application afin de synchroniser les données de l'application entre les clients et de les stocker sur Firebase Cloud.

Les bibliothèques clients sont fournies par l'entreprise, qui permet l'intégration avec des applications Android, IOS et JavaScript.

Firestore

Firestore est un service de stockage d'objets. Le développeur peut l'utiliser pour stocker des photos, du son, des vidéos ou tout autre contenu créé par l'utilisateur.

Firestore Test Lab for Android

C'est une infrastructure basée sur le cloud pour tester les applications Android, les développeurs peuvent tester leurs applications sur une grande variété de périphériques et de configurations.

Firestore Crash Reporting

Crash Reporting est un outil en temps réel qui facilite le suivi, la priorisation et la résolution des problèmes de stabilité qui affectent la qualité des applications. Il permet de gagner du temps dans le dépannage en groupant intelligemment les pannes et en mettant en évidence les circonstances qui les conduisent.

Firestore Notifications

Il permet d'envoyer des notifications ciblées aux utilisateurs par les développeurs d'applications mobiles.



FIG. 5.4 : Logo Firebase

5.3 Système de capture de contexte

Notre système est conçu de base pour la capture des informations contextuelles à l'aide des différents capteurs intégrés dans le système. Il s'agit d'une chambre intelligente qui se compose de capteur, d'actionneur et d'un module WiFi pour la connexion à Internet.

5.3.1 Composants matériels utilisés

Carte Arduino Mega 2560

Arduino est une plate-forme informatique qui s'appuie sur une carte physique et un environnement de développement utilisé pour écrire des logiciels. Une "carte Arduino" se compose d'un **Microcontrôleur** (Atmel 8 bits AVR) et des composants complémentaires qui facilitent la programmation de la carte et son intégration avec d'autres circuits.

Une partie importante de l'Arduino est constituée par les connecteurs standards, qui permettent aux utilisateurs de connecter la carte à des modules complémentaires différents et largement interchangeables. Il y a plusieurs types de cartes Arduino, ces cartes peuvent être utilisées à de nombreux usages différents, avec des applications pratiquement illimitées, La plupart de ces cartes ont quelques pièces qui sont partagées entre les modèles et qui effectuent plus ou moins les mêmes fonctions [47].

L'Arduino Mega que nous avons utilisé a un total de 54 pins d'entrée et de sortie, dont 14 peuvent être utilisés comme des sorties PWM (Pulse Width Modulation). Il a également 16 entrées analogiques. Ce type de carte est très utile aux projets qui nécessitent beaucoup d'entrées et de sorties numériques.

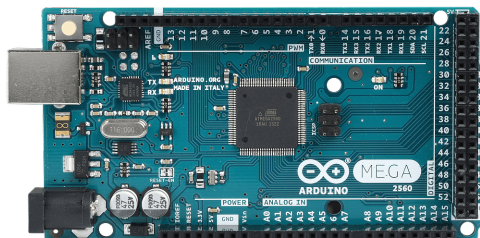


FIG. 5.5 : Carte Arduino Mega 2560

Lecteur RFID RC522

RFID signifie radio-frequency identification qui utilise des champs électromagnétiques pour transférer des données à courte distance. il permet d'identifier les personnes, de réaliser des transactions, etc [64].



FIG. 5.6 : Lecteur RFID RC522

Capteur de température et d'humidité DHT22

Les capteurs DHT sont destinés à la mesure de la température et de l'humidité. Ces capteurs comportent une puce qui fait une conversion analogique numérique et émet un signal numérique indiquant la température et l'humidité. Ces signaux sont faciles à lire avec n'importe quel Microcontrôleur (MCU) [64].

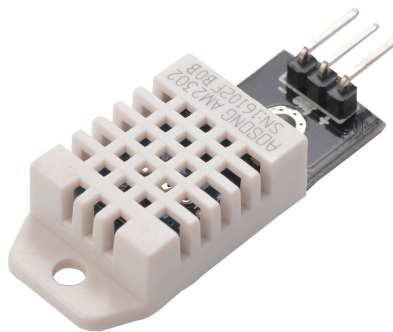


FIG. 5.7 : Capteur de température et d'humidité DHT22

Capteur de Pluie

Le capteur de pluie est utilisé pour détecter l'eau, il se compose de deux parties : la carte électronique et la carte de collecte qui recueille les gouttes d'eau [64]. Il détecte les gouttes d'eau lorsque celles-ci court-circuitent les pistes du circuits imprimés de la carte de collecte.

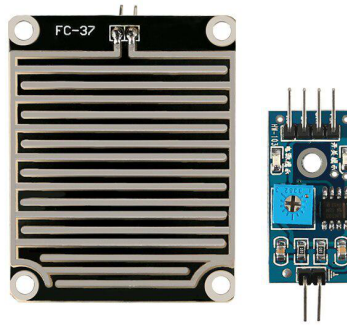


FIG. 5.8 : Capteur de Pluie

Servomoteur

Les Servo moteurs sont des dispositifs simples qui tournent selon leur axe lorsqu'ils sont alimentés en énergie. Les servomoteurs se distinguent par le fait qu'ils connaissent exactement l'angle de leur localisation et ajustent leur position selon les besoins, la plupart des servomoteurs ne peuvent pas tourner à 360 degrés, mais seulement à 180 degrés [38].



FIG. 5.9 : Servomoteur

Capteur de luminosité TSL2561

TSL2561 est un capteur de lumière numérique avancé et précis, idéal pour une utilisation dans une large gamme de situations lumineuses, permettant des calculs exacts de lux et peut-être configuré pour différentes plages de gain/temps. Il contient des diodes infrarouges et à spectre complet servant à mesurer la lumière infrarouge, la lumière à spectre complet ou la lumière visible humaine séparément [4].

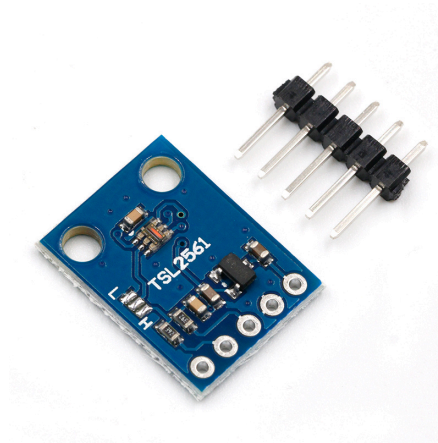


FIG. 5.10 : Capteur de luminosité TSL2561

Capteur de Mouvement

Le capteur de mouvement permet de détecter les mouvements, il mesure la lumière infrarouge des objets dans son champ de vision. Il peut donc détecter le mouvement quand la lumière infrarouge change dans l'environnement. Il est idéal pour détecter si un humain s'est déplacé dans ou hors de la portée du capteur [64].



FIG. 5.11 : Capteur de Mouvement

Module WiFi ESP8266-01

L'ESP8266 est une micro puce WiFi low-cost, dotée d'une pile TCP/IP complète et d'un Microcontrôleur, produite par le fabricant Espressif Systems à Shanghai, en Chine. L'ESP8266 offre une solution réseau WiFi complète et autonome, permettant d'héberger l'application ou de transférer toutes les fonctions réseau WiFi d'un autre processeur d'application [53].

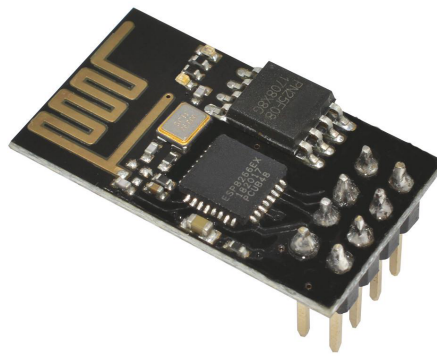


FIG. 5.12 : Module WiFi ESP8266-01

Liquid Crystal Displays

L'écran LCD est programmé pour afficher ce que vous voulez dans une matrice de 16x2. L'écran est composé de deux lignes pouvant contenir 16 caractères par ligne [64].



FIG. 5.13 : LCD

Breadboard

Breadboard est une carte rectangulaire en plastique avec de nombreux petits trous. Il est facile d'insérer un composant électrique ou électronique dans ces trous lors de l'essai d'un circuit. Les connexions ne sont pas permanentes, il est donc facile d'enlever un composant ou de recommencer à zéro. Aucune soudure n'est nécessaire pour établir une connexion. Il suffit d'insérer et de pousser le composant [49].

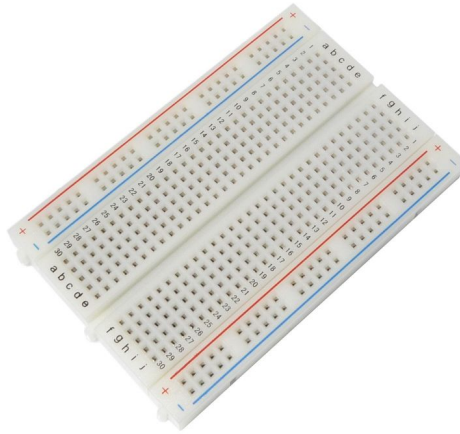


FIG. 5.14 : Breadboard

5.3.2 Raccordement des composants du système

Raccordement de chaque composant a la carte Arduino Mega, la source d'énergie pour alimenter la carte est le câble USB, les autres composants sont alimentés soit par l'Arduino, soit par le module d'alimentation rattacher à la breadboard (MB-102).

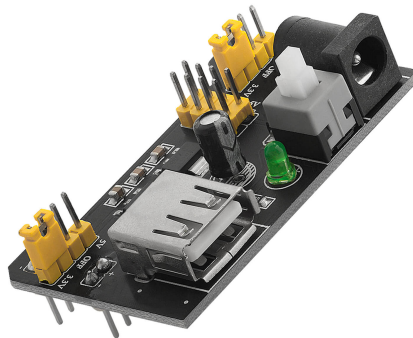


FIG. 5.15 : Alimentation rattacher à la breadboard MB-102

Le raccordement peut changer en fonction du type de pin (les entrées), ou de la disponibilité de certaine pièce.

Raccordement du lecteur RFID RC522

Notre système contient trois lecteurs RFID, raccordés à la même carte Arduino. Le RC522 à 7 pins, deux pour l'alimentation (3.3V) et le ground (la terre), quatre autres qui sont le RST, MISO, MOSI et SCK qui peuvent être commun entre les 3 lecteurs RFID, le dernier est une sortie digitale SDA qui doit être unique.

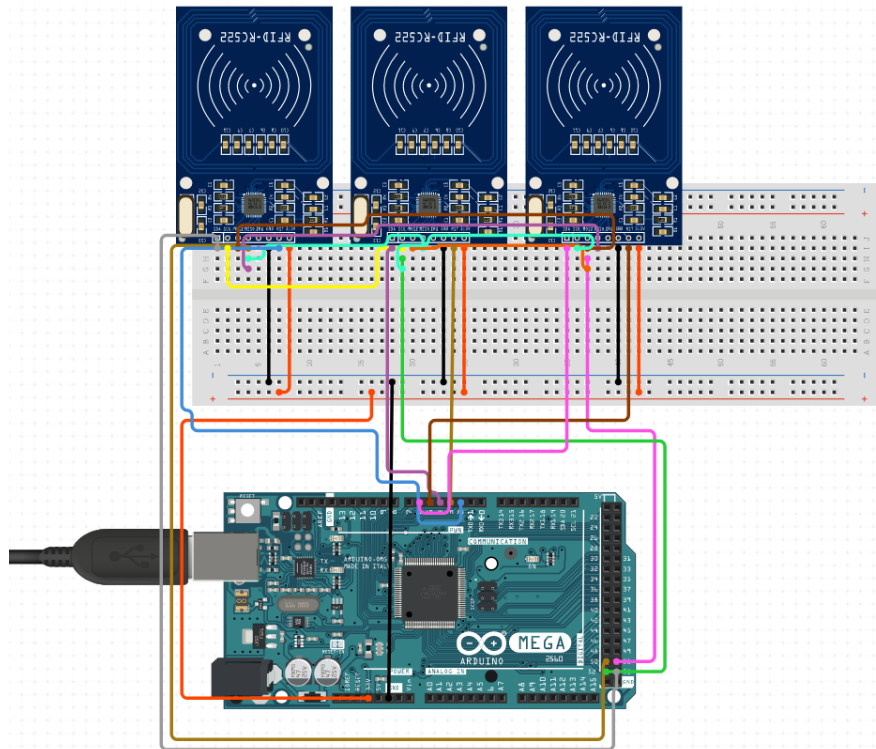


FIG. 5.16 : Raccordement des lecteurs RFID RC522

Raccordement du capteur de température et d'humidité DHT22

Ce capteur à 3 pins, deux pour l'alimentation (5V) et le ground, et une sortie digitale.

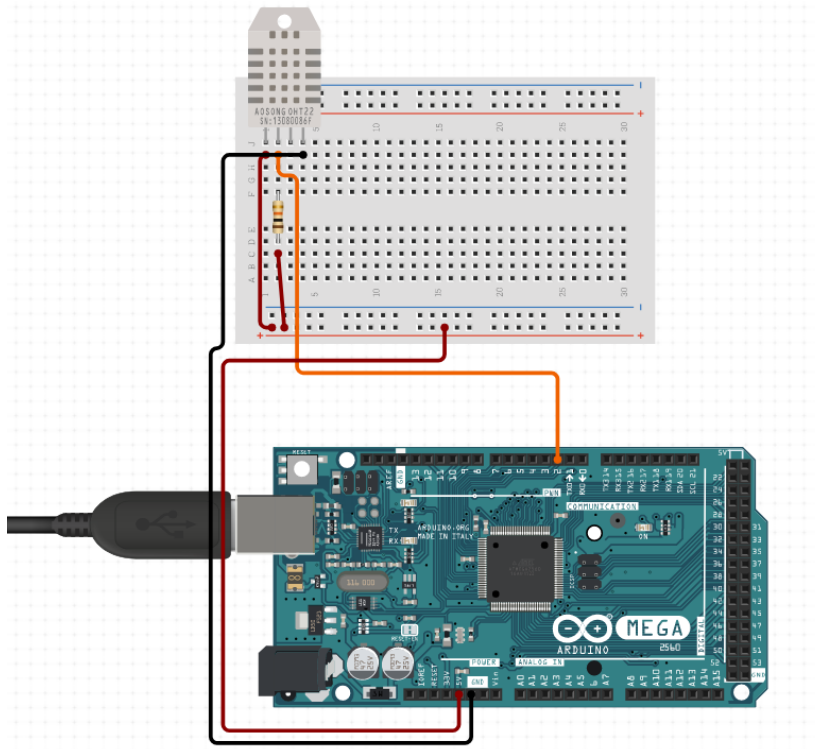


FIG. 5.17 : Raccordement du capteur de température et d'humidité DHT22

Le capteur de pluie a presque le même raccordement, la différence est que la sortie est analogique.

Raccordement du capteur de luminosité TSL2561

Ce capteur à 4 pins, deux pour l'alimentation (5V) et le ground, et deux pins SDA et SCL qui doivent être connecté avec les pins du même nom sur l'Arduino (I2C).

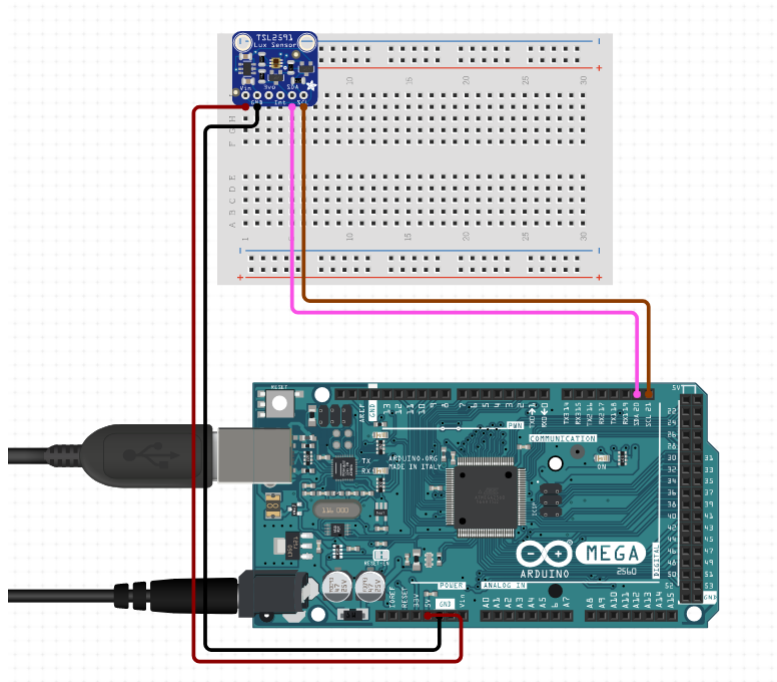


FIG. 5.18 : Raccordement du capteur de luminosité TSL2561

Raccordement du capteur de Mouvement

Ce capteur à 3 pins, deux pour l'alimentation (5V) et le ground, et une sortie digital.

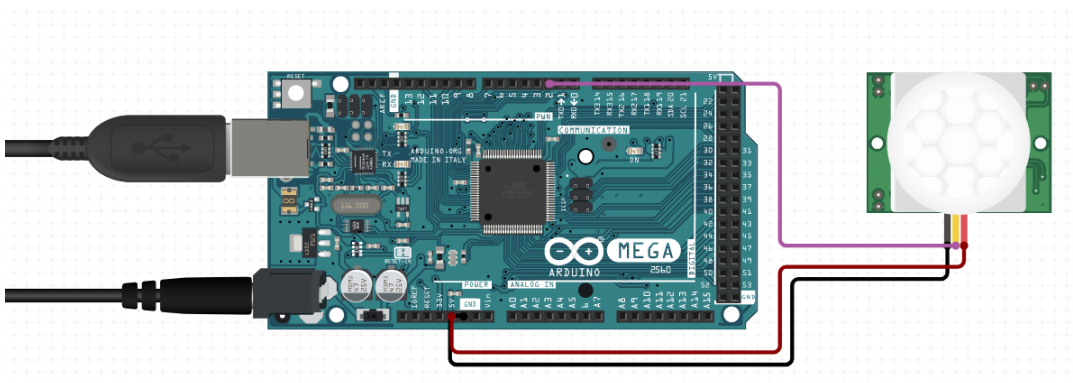


FIG. 5.19 : Raccordement du capteur de Mouvement

Raccordement du Servomoteur

Ce capteur à 3 pins, deux pour l'alimentation (5V) et le ground, et une sortie digital.

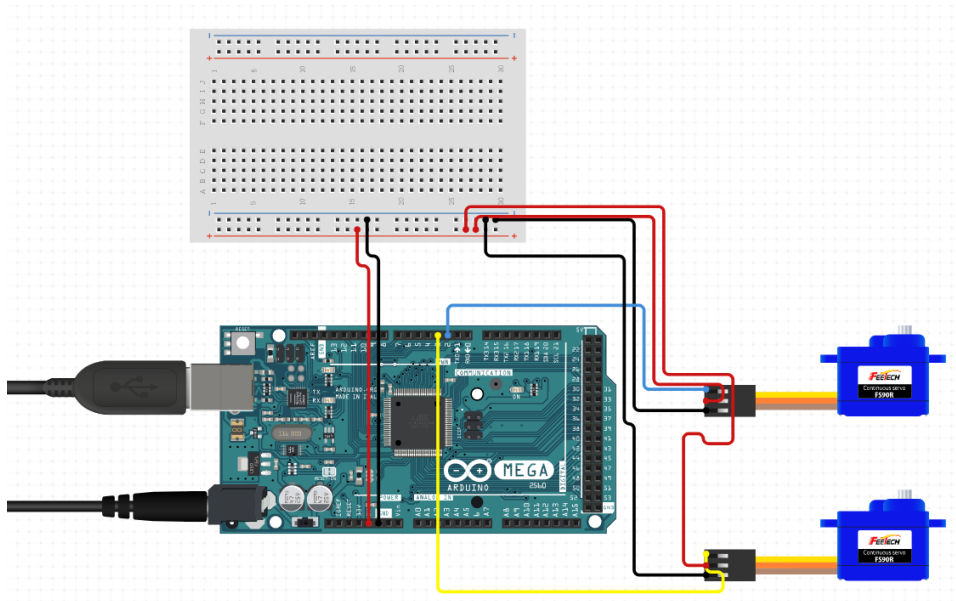


FIG. 5.20 : Raccordement du Servomoteur

Raccordement du LCD

Pour le raccordement du LCD on a utilise un module I2C, qui réduit le nombre de pin a 4 au lieu de 16, deux pour l'alimentation (5V) et le ground, et deux pins SDA et SCL qui doivent être connecté avec les pins du même nom sur l'Arduino.

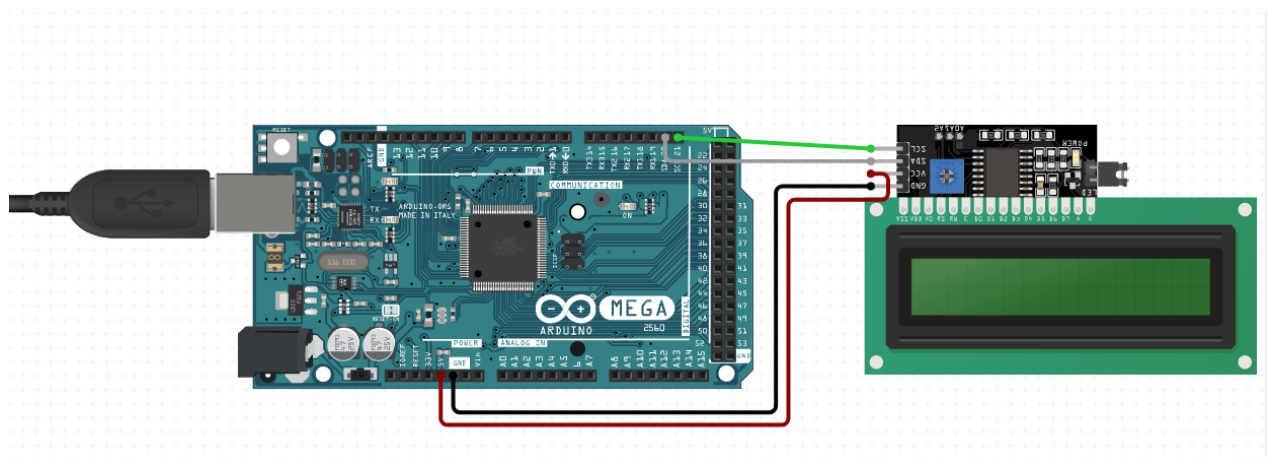


FIG. 5.21 : Raccordement du LCD

Ce composant utilise les mêmes pins que le capteur de luminosité TSL2561, c'est-à-dire SDA et SCL, ils peuvent être commun entre les deux composants.

Raccordement du module WiFi

Le module WiFi ESP8266-01 est une carte dotée d'un microcontrôleur. Il a donc son propre code, et communique avec la carte Arduino avec une communication en série (serial communication).

Les deux pins RX et TX (R pour receive et T pour transmit) son raccordés avec un des pins RX et TX de la carte Arduino (il doivent être inversés). Les deux pin VCC et EN son raccordés a une l'alimentation (3.3V) et le GND au ground.

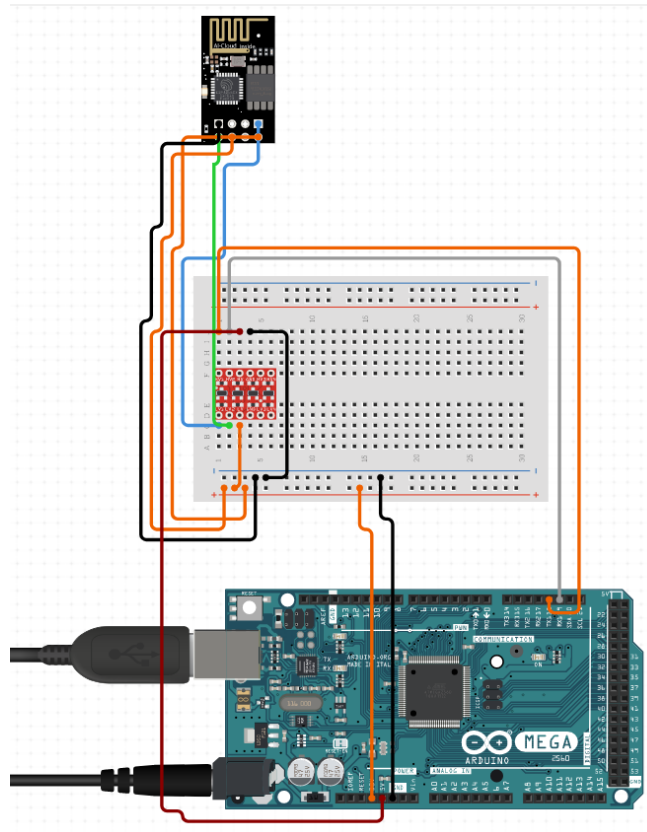


FIG. 5.22 : Raccordement du Module WiFi ESP8266-01

5.3.3 Bibliothèque Arduino IDE

Arduino Mega

- SPI.h : Cette bibliothèque permet de communiquer avec des périphériques SPI (Serial Peripheral Interface), avec l'Arduino comme périphérique contrôleur. Une communication en bus de données série.
- SoftwareSerial.h : La bibliothèque permet la communication en série sur d'autres broches numériques d'une carte Arduino.
- RFID.h : Permet de Lire et écrire différents types de cartes d'identification par radio-fréquence (RFID) sur l'Arduino en utilisant un lecteur basé sur le RC522 connecté via l'interface SPI.

- DHT.h : Une bibliothèque Arduino pour la série DHT de capteurs de température/humidité.
- Adafruit-TSL2561-U.h : Une bibliothèque pour faire fonctionner le capteur de luminosité TSL2561.
- Servo.h : Permet aux cartes Arduino de contrôler une variété de servomoteurs.
- LiquidCrystal-I2C.h : Cette bibliothèque permet de contrôler les écrans LCD I2.

Module WiFi ESP8266-01

- ESP8266WiFi.h : Une bibliothèque qui gère le bon fonctionnement du module ESP8266-01.
- SPI.h : La même que celle de l'Arduino.
- SoftwareSerial.h : La même que celle de l'Arduino.
- ArduinoJson.h : Une bibliothèque C++ JSON pour Arduino et IoT, nécessaire pour le fonctionnement d'autres bibliothèques (FirebaseArduino.h).
- FirebaseArduino.h : Une bibliothèque Arduino qui montre comment appeler l'API Firebase à partir du noyau Arduino ESP8266.
- WiFiUdp.h : Une bibliothèque qui gère la communication UDP entre l'ESP8266 et un client externe, nécessaire pour le fonctionnement d'autres bibliothèques (NTPClient.h).
- NTPClient.h : Une bibliothèque Arduino pour avoir l'heure à partir d'un serveur NTP (Network Time Protocol).

5.3.4 Réalisation du système

Les informations contextuelles sont captées par l'Arduino, ensuite transmises au module WiFi ESP grâce à la communication en série, le module les transferte à la base de données cloud (Firebase), à l'aide de l'API de Firebase qui utilise l'architecture du service web REST pour transmettre les informations ou les recevoir. Ensuite les informations sont exploitées par les autres périphériques du système.

Note : Il existe plusieurs types de modules d'identification par radiofréquence (RFID), en fonction de la puissance de la radiofréquence, pour la capture de la position, il faut un module de haute radiofréquence (HF), mais pour la simulation on va utiliser des modules à basse radiofréquence (LF) qui est plus accessible.

Voici des images de la maquette réalisée :



FIG. 5.23 : Images 1 de la maquette

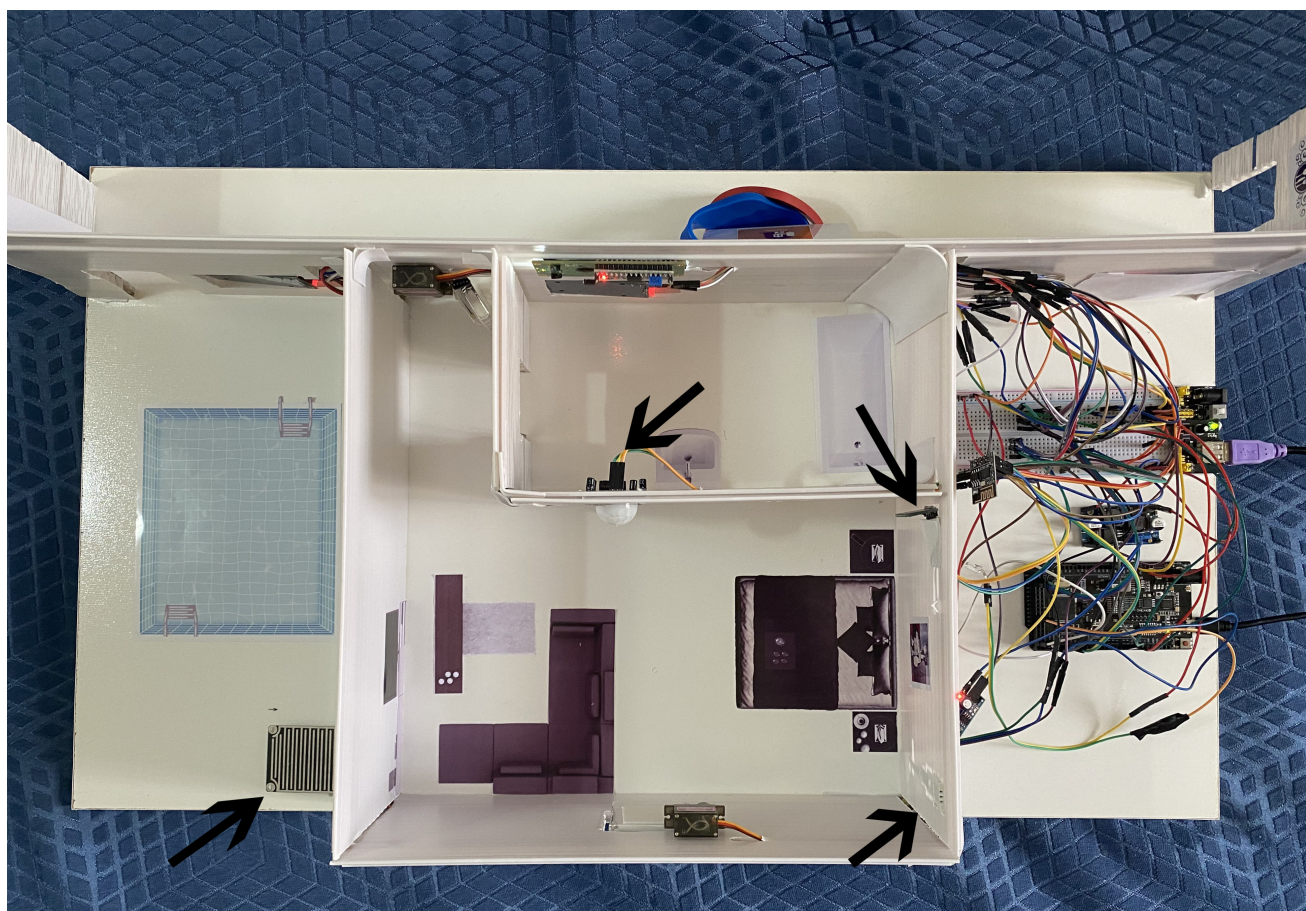


FIG. 5.24 : Images 2 de la maquette



FIG. 5.25 : Images 3 de la maquette

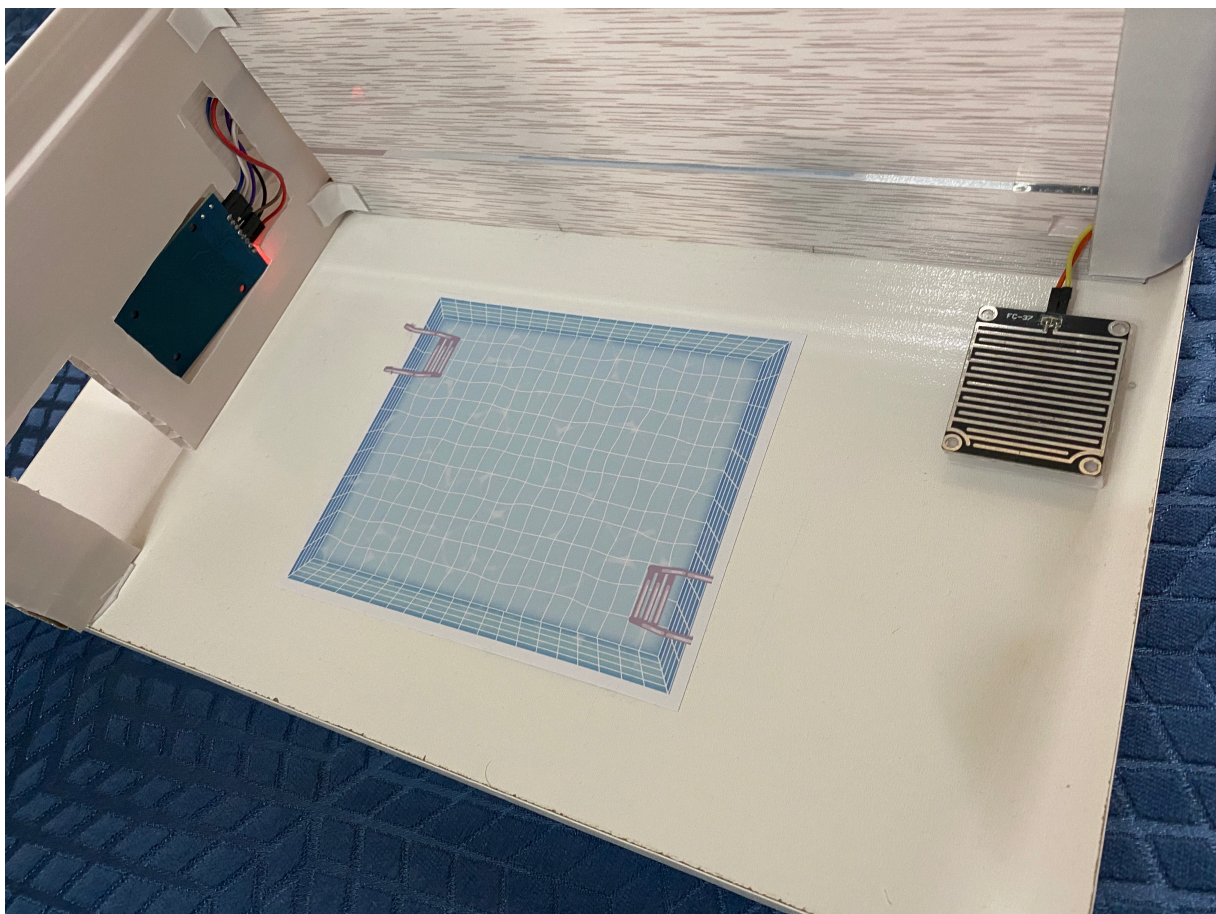


FIG. 5.26 : Images 4 de la maquette



FIG. 5.27 : Images 5 de la maquette

Les données contextuelles sont transférés à la basse de données cloud Firebase :

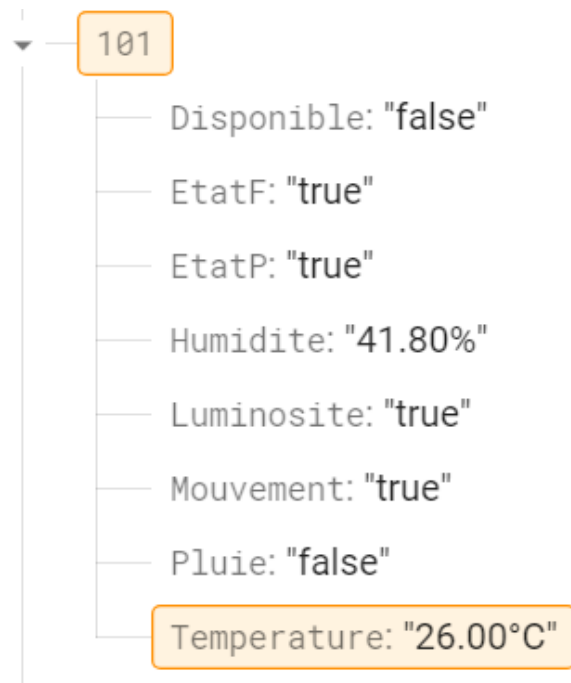


FIG. 5.28 : Données contextuelles dans BDD

5.4 Applications Android de l'hôtel

5.4.1 Librairies et API utilisés

Application mobile Android de l'hôtel

- Google maps : Permet aux développeurs Android d'intégrer Google Maps dans l'application.
- Zxing 4.1 : Une librairie de traitement d'image de codes-barres 1D/2D multi-format, open-source, implémentée en Java, avec des ports pour d'autres langues (pour les code QR).
- Firebase API : Permet aux développeurs de synchroniser et de stocker facilement des données en temps réel dans le cloud. Les développeurs peuvent utiliser ce service pour créer leurs applications sans avoir à gérer des serveurs.

Application gestion de l'hôtel

Elle utilise juste l'API Firebase pour se connecter à la base de données.

5.4.2 Réalisation de l'application de gestion de l'hôtel

C'est une application basique qui est utilisée juste pour une légère gestion de l'hôtel, son objectif est de veiller au bon déroulement des cas d'utilisation de l'application mobile des clients et de la maquette. Voici quelques interfaces des cas d'utilisation :

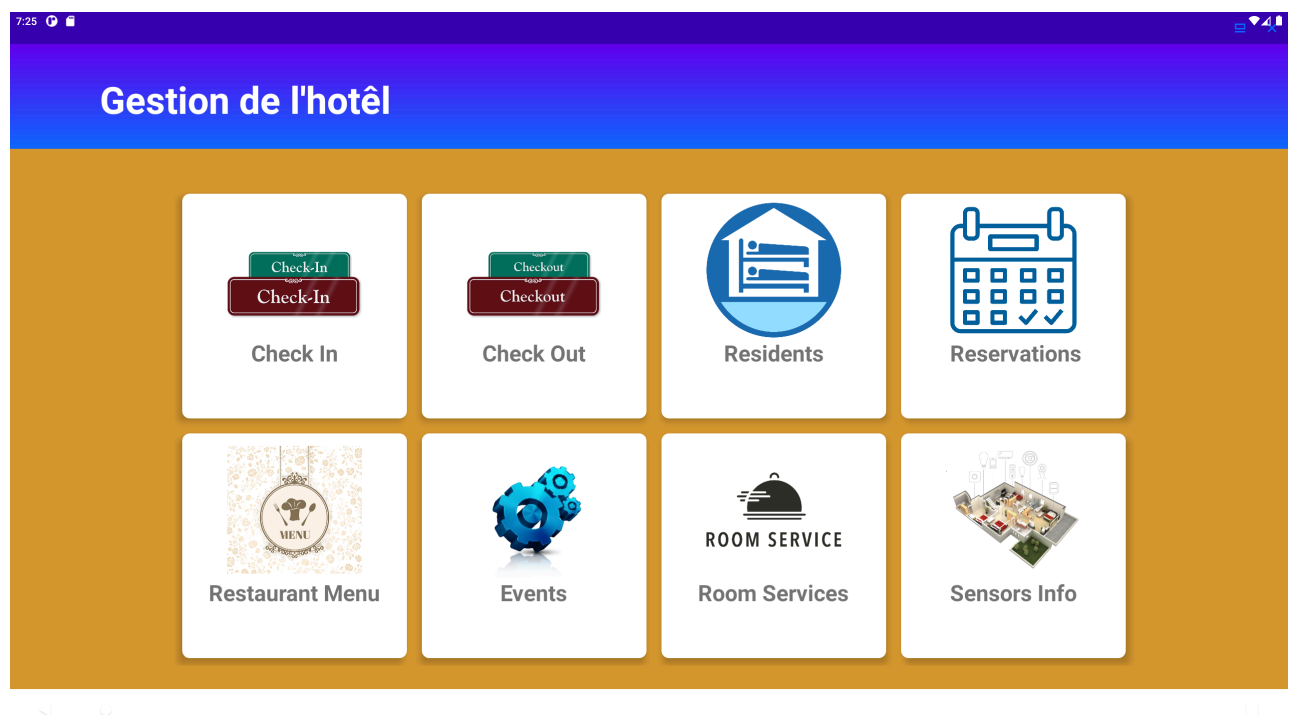
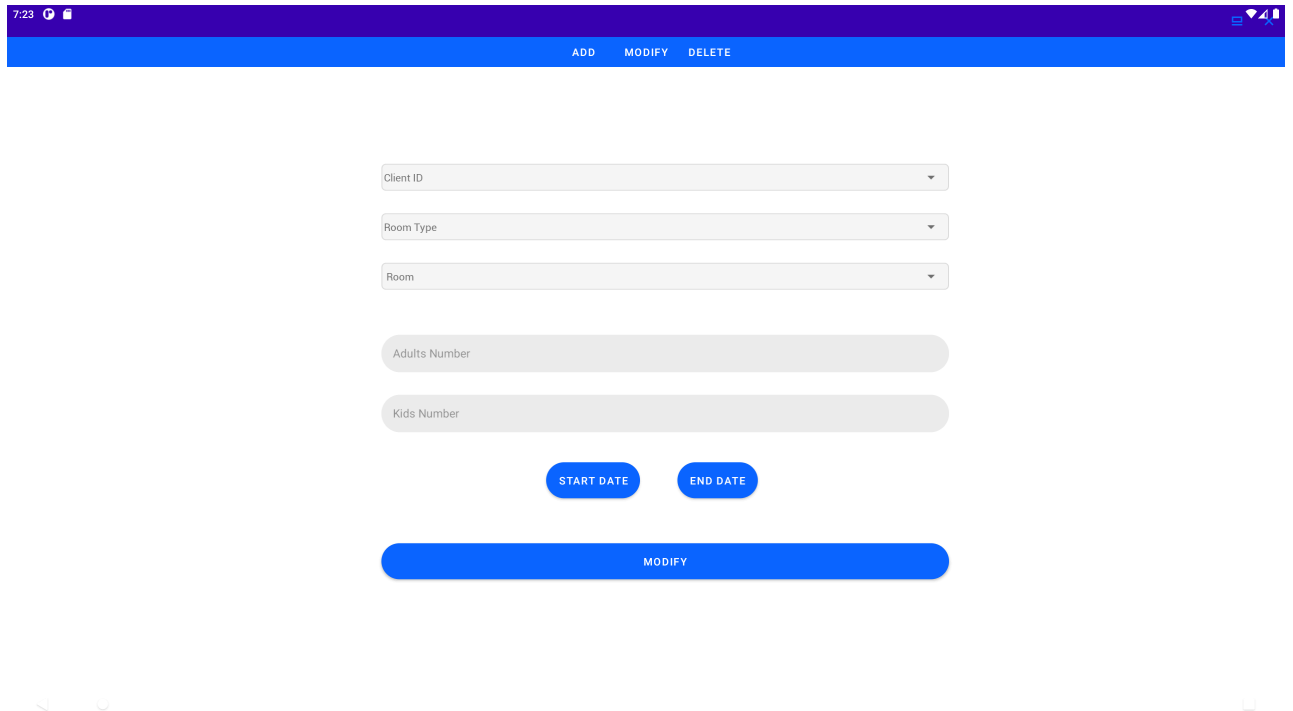
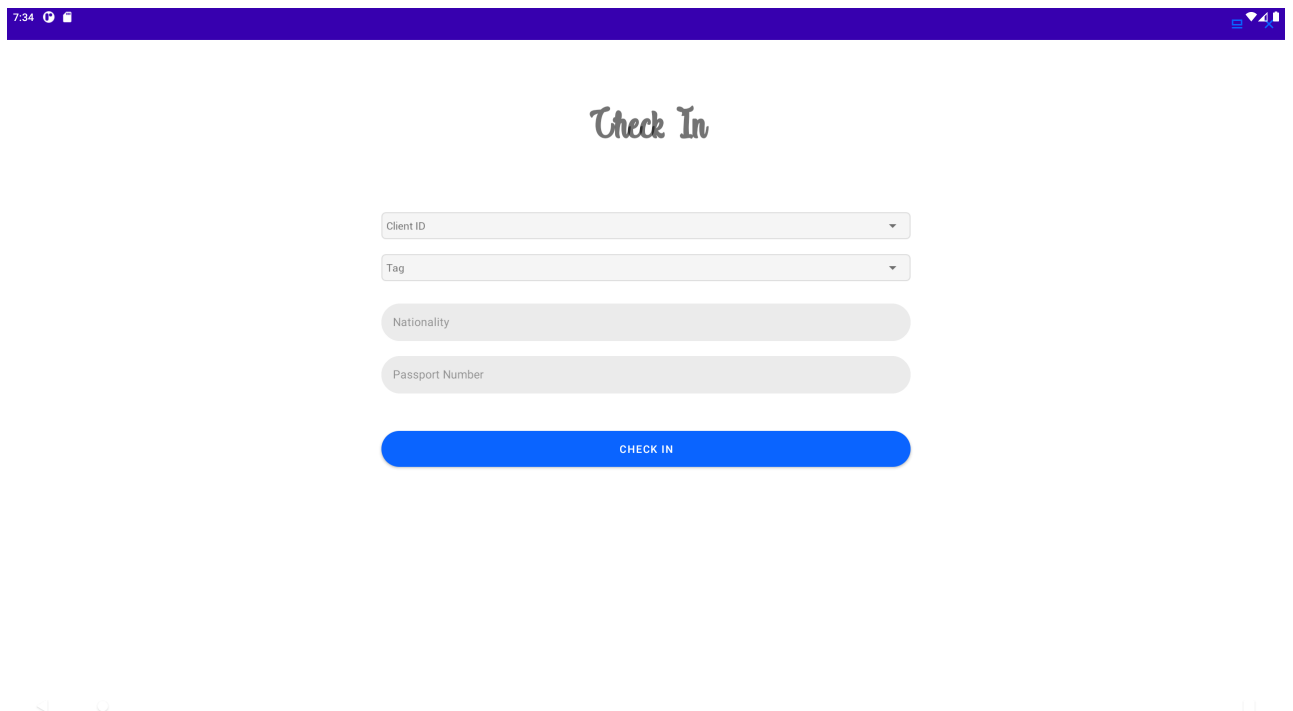


FIG. 5.29 : Interface application de gestion menu



The screenshot shows a mobile application interface for reservation management. At the top, there is a dark blue header bar with the time 7:23 and status icons on the left, and a menu icon and signal icons on the right. Below the header is a blue navigation bar with three options: ADD, MODIFY, and DELETE. The main content area is white and contains a form with the following fields: Client ID (dropdown), Room Type (dropdown), Room (dropdown), Adults Number (text input), Kids Number (text input), START DATE (blue button), END DATE (blue button), and a large blue button labeled MODIFY at the bottom.

FIG. 5.30 : Interface application de gestion réservation



The screenshot shows a mobile application interface for check-in management. At the top, there is a dark blue header bar with the time 7:34 and status icons on the left, and a menu icon and signal icons on the right. Below the header is a white area with the title "Check In" in a stylized font. The main content area is white and contains a form with the following fields: Client ID (dropdown), Tag (dropdown), Nationality (text input), Passport Number (text input), and a large blue button labeled CHECK IN at the bottom.

FIG. 5.31 : Interface application de gestion check-in

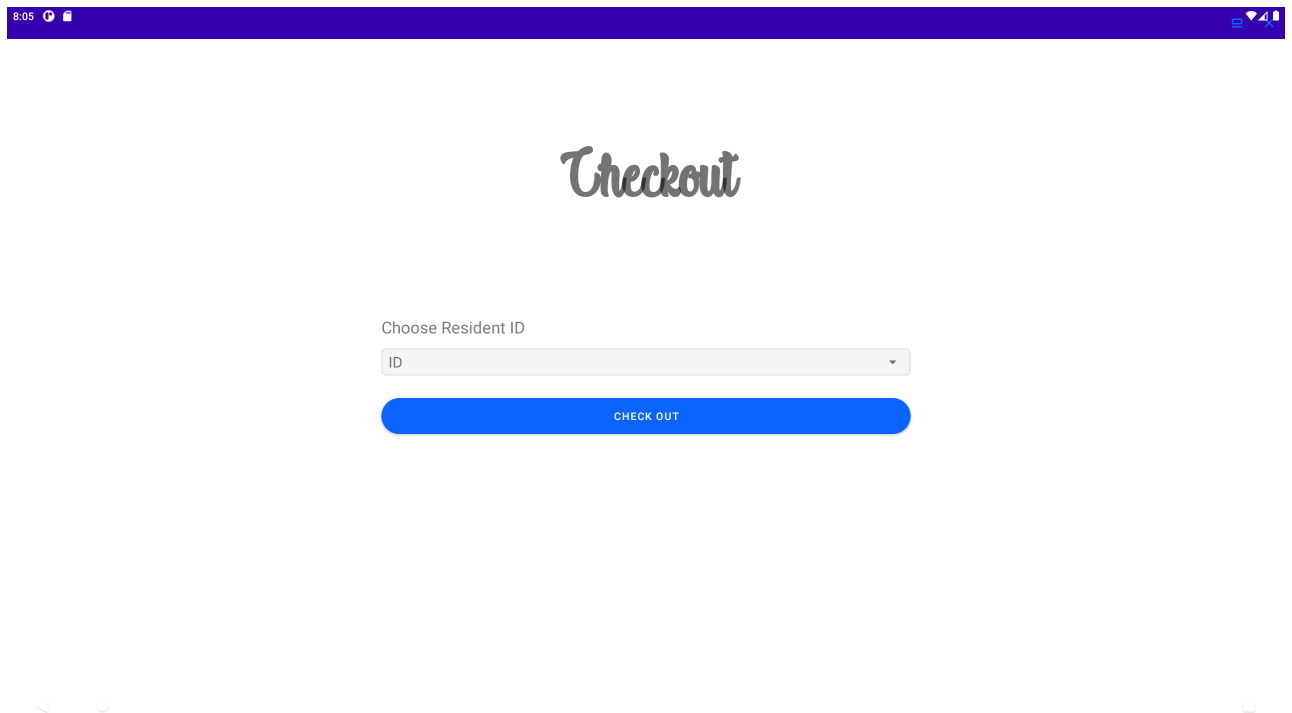


FIG. 5.32 : Interface application de gestion check-out



FIG. 5.33 : Interface application de gestion évènements

5.4.3 Réalisation de l'application mobile

Voici les interfaces des cas d'utilisation de l'application mobile :

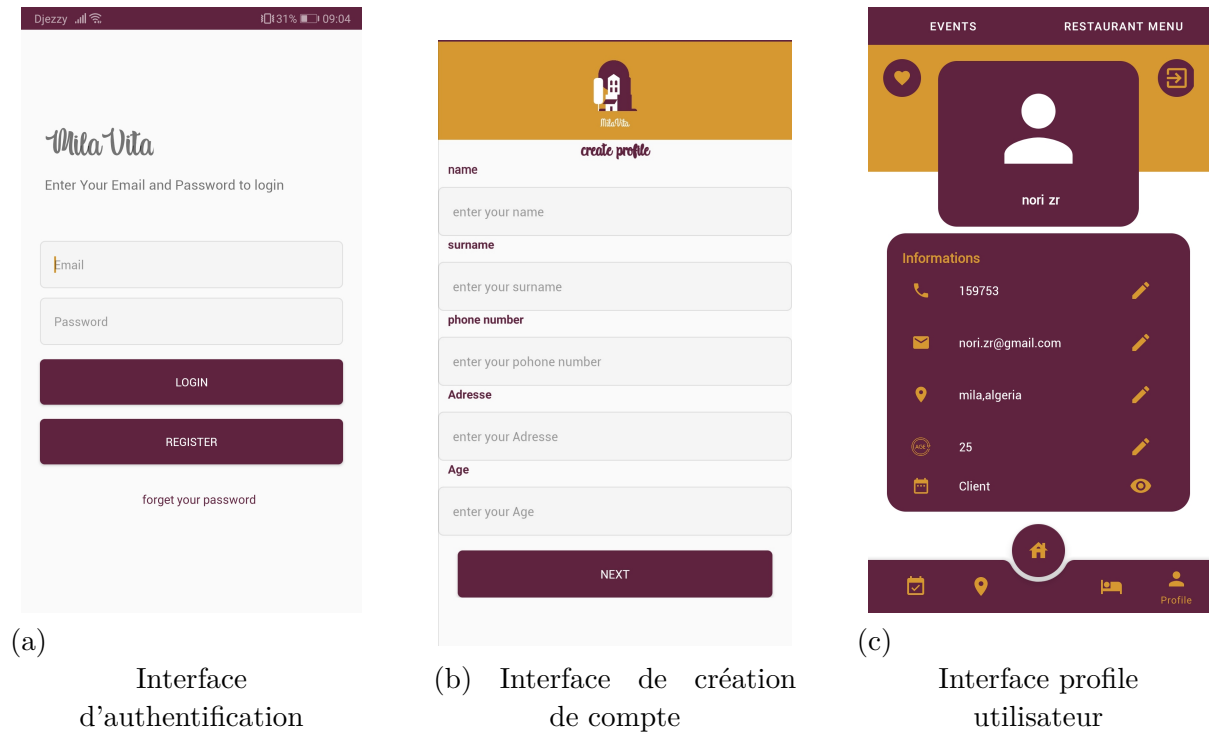


FIG. 5.34 : Interface 1 application mobile

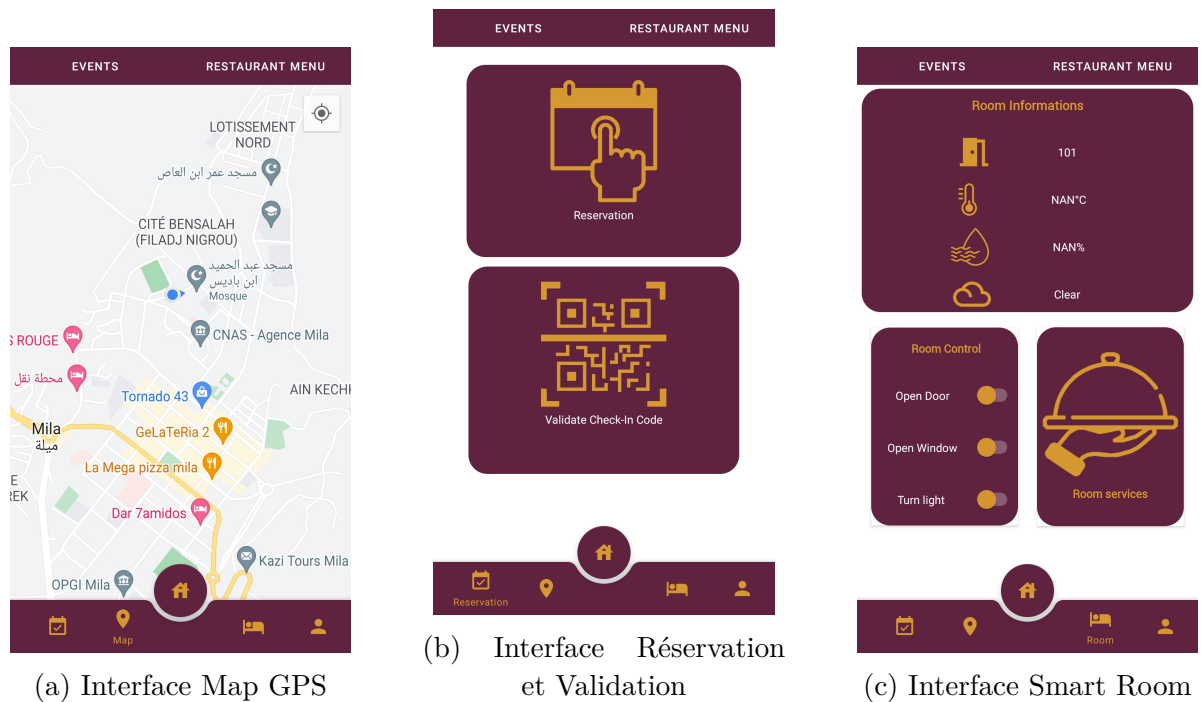


FIG. 5.35 : Interface 2 application mobile

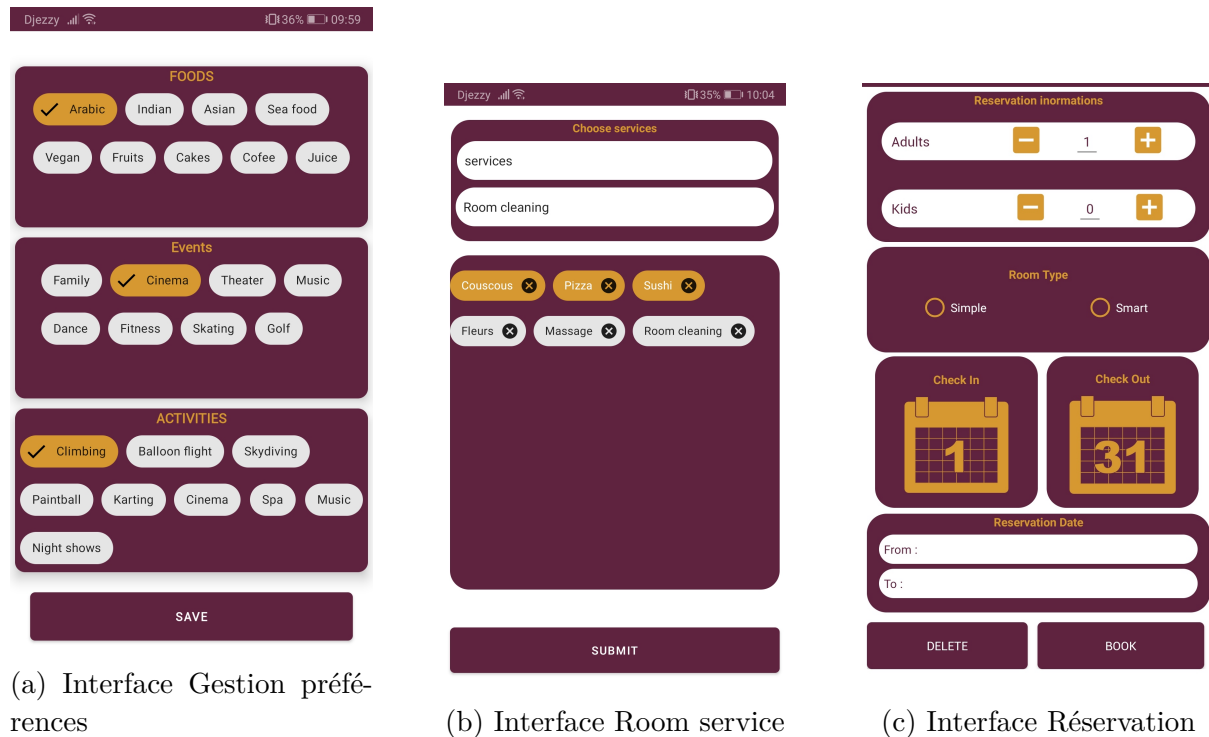


FIG. 5.36 : Interface 3 application mobile

5.5 Sensibilité au contexte du système

On va citer les différents cas d'utilisations du contexte dans notre système :

5.5.1 Notification et prévention

Localisation

Majoritairement basée sur RFID, envoi de notifications relatives à un lieu a la détection des résidents dans ce dernier. En plus la position captée par les RFIDs donne des informations sur la position exacte des résidents à une heure précise. Le GPS aussi fait partie de l'utilisation du contexte pour guider les résidents en dehors de l'hôtel.

Changement du contexte

Envoi de notifications relatives au changement dans l'environnement et du contexte, par exemple une notification pour prévenir de prendre un parapluie quand il y a de la pluie, prévenir d'activer le mode nuit quand la luminosité est basse, prévenir lors du changement brusque de température etc.

Sécurité

Prévention dans le cas d'un résident captée dans un endroit de l'hôtel, et qu'il y a du mouvement dans sa chambre. En plus la localisation intérieure par RFID peut se trouver

utile en cas de problème dans l'hôtel.

Préférence

Envoi de notification relative aux préférences de l'utilisateur, par exemple avertir un résident quand un plat ou aliment dans ces préférences est mis dans le menu du restaurant, même avertissement concernant les activités.

5.5.2 Classification naïve bayésienne

Cette sensibilité consiste à utiliser les informations contextuelles captées pour offrir une meilleure expérience pour les résidents de l'hôtel, et cela en classifiant des activités en fonction de la température, l'humidité, la condition météo (pluie) et la moyenne d'âge des résidents.

La classification d'une activité selon le contexte du jour, consiste dans le calcul de la probabilité de la possibilité d'effectuer cette dernière ou non, et ce grâce à un Data Set de l'historique de l'activité concernée.

Comme exemple, est ce qu'on peut faire une activité de Football dans l'hôtel, alors que la température est à 25 °C, l'humidité est à 60%, avec de la pluie et la moyenne d'âge des résidents est de 46 ans. Avec l'aide de cette classification, et de la véracité du Data Set, on aura une réponse qui prend en compte le contexte, et la condition des résidents pour une automatisation et pour une satisfaction de ces derniers.

Un classificateur naïve bayésien est un modèle d'apprentissage automatique probabiliste utilisé pour les tâches de classification. Le cœur du classificateur est basé sur le théorème de Bayes.

Tout nos attribues sont numériques, le calcul de la probabilité se fait comme suit pour les deux cas (pratiquer ou pas). Pour le calcul de la probabilité de chaque contexte, on a :

$$P(\text{Contexte}_{k,j}|\text{Pratiquer}) = f_{k,j}(x) = \frac{1}{\sqrt{2\pi\sigma_{k,j}^2}} \exp\left(\frac{-1}{2\sigma_{k,j}^2}(x - \mu_{k,j})^2\right)$$

Calcule de la moyenne :

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i$$

Calcule de la variance :

$$\sigma^2 = \frac{1}{(N-1)} \sum_{i=1}^N (x_i - \mu)^2$$

Après le calcul de la probabilité de chaque contexte, on calcule la probabilité générale de pratiquer l'activité en question :

$$P_p(\text{Pratiquer}) = P(\text{Pratiquer})P(\text{Température}|\text{Pratiquer})P(\text{Humidité}|\text{Pratiquer}) \\ P(\text{Météo}|\text{Pratiquer})P(\text{Moyenne d'âge}|\text{Pratiquer})$$

On a donc la probabilité de ne pas pratiquer :

$$P_p(\text{Ne pas pratiquer}) = 1 - P_p(\text{Pratiquer})$$

le data set a été alimenté manuellement vu la non-disponibilité, mais il peut être enrichi par l'historique d'activité de l'hôtel si on est satisfait du résultat de la classification, notre data set est sous forme de fichier excel. Voici un exemple du data set de l'activité Tennis :

Tennis				
Resultat	Température	Humidité	Pluie	Moyenne d'age
success	30	60	1	25
success	15	35	1	30
success	16	80	1	24
success	10	55	1	29
success	25	50	2	35
success	30	60	1	34
success	15	35	1	40
success	10	55	1	24
success	25	50	2	26
success	25	50	1	30
fail	20	60	2	50
fail	14	50	2	60
fail	20	90	2	55
success	30	60	1	25
fail	40	65	2	35
fail	14	50	2	44
fail	20	90	2	60
fail	2	50	2	45
fail	16	80	2	26
fail	10	55	1	56
success	9	60	2	21
success	25	50	2	35
fail	2	50	1	45
success	16	80	1	24

FIG. 5.37 : Exemple de Data set

le résultat dépend de la véracité du data set et de l'historique des activités, mais peut être renforcé après chaque classification.

5.6 Conclusion

Dans ce dernier chapitre nous avons présenté tout ce qui est lié au développement de notre système, avec la description de chaque composant. Nous avons parlé des différents outils et choix technologiques utilisés dans l'implémentâmes. On va conclure notre travail dans la section suivante.

Conclusion générale

L'Objectif de notre travail était de réaliser un système sensible au contexte afin de garantir une prestation de qualité dans l'hôtellerie. En abordant une variété de notions telles que l'informatique diffuse, la sensibilité au contexte et les applications mobiles, les capteurs, l'IoT nous avons essayé de mettre en œuvre ces dernières et de construire un système ubiquitaire.

L'utilisation des informations contextuelles pour la découverte de services fait de l'application implémentée un système "intelligent", capable de réagir de manière dynamique et automatique sans intervention de l'utilisateur (résident / personnel de l'hôtel) ce qui répond à l'un des objectifs principaux de l'informatique ubiquitaire.

Ce projet nous a été très bénéfique, car il nous a permis d'enrichir nos connaissances sur les deux plans : théorique et pratique. Il nous a aussi permis de découvrir et d'acquérir de nouvelles connaissances en matière de programmation et de développement dans le domaine des applications mobiles (Android), le domaine électronique (raccordement électriques des capteur) ,configuration et programmation des capteurs (Arduino), utilisation du cloud Google, etc.

En réalisant ce travail, nous pouvons dire que l'objectif de notre projet de Master a été bien achevé. Certes, le développement a nécessité énormément de temps, de ressources, de recherches et aussi de grands efforts de programmation. Donc, faute de temps et de moyen certains points n'ont pas été pris en compte dans la partie réalisation de l'application de gestion et mobile, d'autre n'ont pas été achevés.

Ces contraintes de temps et de ressources nous ont permis de tracer une ligne de perspectives, nous évoquons :

- Utiliser d'autres capteurs matériaux qui vont permettre de capter et exploiter plus d'informations contextuelles telles que la caméra.
- Le déploiement sur le terrain pour mesurer ses performances.
- Enrichir le profil des utilisateurs pour une meilleure gestion de contexte.
- Étendre l'utilité de l'application à l'environnement de l'utilisateur.

En fin, nous espérons que ce travail sera une source d'inspiration bénéfique pour les futurs étudiants qui aborderont ce genre de thème dans leur conduite de projet informatique.

Bibliographie

- [1] Mohamed ABDEL-BASSET et al. "Internet of things in smart education environment : Supportive framework in the decision-making process". In : *Concurrency and Computation : Practice and Experience* 31 (mai 2018), DOI : 10.1002/cpe.4515.
- [2] Gregory D. ABOWD et al. "Towards a Better Understanding of Context and Context-Awareness". In : *HUC*. 1999, ISBN : 978-3-540-48157-7.
- [3] Sonia ACHIRI. "Une Architecture Agents pour l'Adaptation au Contexte des Systèmes d'Information Ubiquitaires, mémoire en vue de l'obtention du diplôme de Magister, Université Badji Mokhtar Annaba". In : 2012.
- [5] Xia Feng et AL. "Internet of things, International Journal of Communication Systems, vol 25, no 9, pp 1101". In : 2012.
- [6] BENSEGHIR Lynda AREZKI YASMINE. "Sélection et composition de services sensible au contexte avec QoS globale dans les environnements ubiquitaires, mémoire en vue de l'obtention du diplôme de Master en Informatique, Université Badji Mokhtar Annaba". In : 2014.
- [7] Jim ARLOW et Ila NEUSTADT. *UML and the Unified Process : Practical object-oriented analysis and design*. en. Boston, MA : Addison Wesley, 2001.
- [8] Kevin ASHTON. "That 'Internet of Things' Thing : In the Real World Things Matter More than Ideas. RFID Journal." In : 2009.
- [9] Matthias BALDAUF, Schahram DUSTDAR et Florian ROSENBERG. "A survey on context-aware systems". In : *Int. J. Ad Hoc Ubiquitous Comput.* 2 (2007), DOI : 10.1504/IJAHUC.2007.014070.
- [10] Lionel BALME. "Interfaces homme-machine Plastiques : une approche par composants dynamiques". In : (juin 2008).
- [11] Jakob Eyvind BARDRAM et Adrian FRIDAY. "Ubiquitous Computing Systems". In : *Ubicomp 2009*. 2009.
- [12] NGUYEN Thanh BINH. "Gestion de profil apprenant distribue dans un environnement ubiquitaire, mémoire de stage de fin d'étude master en informatique, Institut de la Francophonie pour l'informatique". In : 2009.
- [13] Peter J. BROWN. "The Stick-e Document : a Framework for Creating Context-aware Applications". In : 1996.
- [14] Angelo Paolo CASTELLANI et al. "Web Services for the Internet of Things through CoAP and EXI". In : *2011 IEEE International Conference on Communications Workshops (ICC)* (2011), p. 1-6.

- [15] Tarak CHAARI. “Adaptation d’applications pervasives dans des environnements multi-contextes”. In : 2007.
- [16] M. CHEBBI. “Utilisation des patrons de conception pour l’adaptation des interfaces graphiques des applications mobiles dépendantes du contexte”. In : 2014.
- [22] Anind K. DEY et Gregory D. ABOWD. “Providing architectural support for building context-aware applications”. In : 2000.
- [23] Anind K. DEY, Albrecht SCHMIDT et Joseph F. MCCARTHY. “UbiComp 2003: Ubiquitous Computing”. In : *Lecture Notes in Computer Science*. 2003.
- [24] Anind K. DEY, Albrecht SCHMIDT et Joseph F. MCCARTHY. “UbiComp 2003: Ubiquitous Computing”. In : *Lecture Notes in Computer Science*. 2003.
- [27] Ling FENG. *Context-Aware Computing*. De Gruyter, déc. 2017.
- [28] Lizeth GHANDI et al. “Mobile application development process : A practical experience”. In : *2017 12th Iberian Conference on Information Systems and Technologies (CISTI)* (2017), p. 1-6.
- [29] Twinkle Rameshbhai GONDALIYA. “A Survey on An Efficient IOT Based Smart Home”. In : 2016.
- [30] MD Rashedul ISLAM et Tridib MAZUMDER. “Mobile application and its global impact, January 2010 International Journal of Engineering and Technology”. In : 2010.
- [31] Mahmud JABRI. *Java Programming Language Report*. en. Rapp. tech. Jan. 2021.
- [32] Vasileios KARAGIANNIS et al. “A survey on application layer protocols for the Internet of Things”. In : *Trans. IoT Cloud Comput.* 3 (jan. 2015), p. 11-17.
- [33] Mohammed Fethi KHALFI et S M BENSLIMANE. “Systèmes D’information Pervasifs : Architecture et Challenges”. In : (2014).
- [34] Chunnu KHAWAS et Pritam Gajkumar SHAH. “Application of Firebase in Android App Development-A Study”. In : *International Journal of Computer Applications* (2018).
- [35] Panu KORPIPÄÄ et al. “Managing Context Information in Mobile Devices”. In : *IEEE Pervasive Comput.* 2 (2003), p. 42-51.
- [36] Sylvain KUBLER et al. “IoT-based Smart Parking System for Sporting Event Management”. In : nov. 2016, p. 104-114.
- [37] J.kiran KUMAR et D.YUGANDHAR. “A Study on Current Mobile Operating Systems, International Journal of Scientific & Engineering Research Volume 8”. In : 2017.
- [38] James A LANGBRIDGE. *Arduino sketches : Tools and techniques for programming wizardry*. en. 1^{re} éd. Nashville, TN : John Wiley & Sons, 2015.
- [39] Jinshan LIU. “Supporting QoS-aware Service Discovery in Ubiquitous Computing Environments. (Découverte de services sensible à la qualité de service dans les environnements de l’informatique diffuse)”. In : 2006.

- [40] Leo LOUIS. “Working principle of arduino and using it as a tool for study and research”. In : *International Journal of Control, Automation, Communication and Systems* 1.2 (2016), p. 21-29.
- [41] M.NARMATHA et S.Venkata KRISHNAKUMAR. “Study on Android Operating System And Its Versions, International Journal of Scientific Engineering and Applied Science (IJSEAS) - Volume 2”. In : 2016.
- [42] Moeiz MIRAOU. “Architecture logicielle pour l’informatique diffuse : modelisation du contexte et adaptation dynamique des services”. In : 2009.
- [44] Ghita Kouadri MOSTÉFAOUI, Jacques PASQUIER-ROCHA et Patrick BRÉZILLON. “Context-Aware Computing : A Guide for the Pervasive Computing Community”. In : *The IEEE/ACS International Conference on Pervasive Services* (2004), p. 39-48.
- [45] Sougata MUKHERJEA. “Mobile Application Development, Usability, and Security”. In : 2016.
- [46] Salma NAJAR et al. “Analyse des mécanismes de découverte de services avec prise en charge du contexte et de l’intention”. In : 2012.
- [47] Matthew OATES. *Arduino for beginners : How to get the most of out of your arduino, including arduino basics, arduino tips and tricks, arduino projects and more !* Ingram Publishing, 2020.
- [49] Ezekiel OCHAMI. *A Novice Guide to Arduino Programming*. en. DRAFT2DIGITAL.COM, 2020.
- [50] Jae-Hyeon PARK et al. “Low-power sensing model considering context transition for location-based services”. In : *Soft Computing* 21 (2017), p. 5223-5233.
- [51] Roques PASCAL. *UML 2 en action : de l’analyse des besoins à la conception / Pascal Roques, Franck Vallée*. fre. 4e édition. Architecte logiciel. Paris : Eyrolles, DL 2007.
- [52] Charith PERERA et al. “Context Aware Computing for The Internet of Things : A Survey”. In : *IEEE Communications Surveys & Tutorials* 16 (2014), p. 414-454.
- [53] Tatap PERETS. “INVESTIGATION OF WI-FI (ESP8266) MODULE AND APPLICATION TO AN AUDIO SIGNAL TRANSMISSION”. Thèse de doct. Août 2019.
- [54] Dr. Yusuf PERWEJ et al. “The Internet of Things (IoT) and its Application Domains”. In : *International Journal of Computer Applications* 182 (avr. 2019), p. 36-49.
- [55] Yusuf PERWEJ et al. “An Extended Review on Internet of Things (IoT) and its Promising Applications”. In : *Communications on Applied Electronics* 7.26 (fév. 2019), p. 8-22.
- [56] Nicolas PLOUZNIKOFF et J. M. ROBERT. “Caractéristiques, enjeux et défis de l’informatique portée”. In : *IHM ’04*. 2004.
- [57] “Présentation générale de l’Internet des objets, Union internationale des télécommunications UIT”. In : *Série y : infrastructure mondiale de l’information, protocole internet et réseaux de prochaine génération* (2012).

- [59] Deepak S.Dandwate PRITEE S.UTTARWAR Rani P.Tidke et Umesh J.TUPE. "A Literature Review on Android - A Mobile Operating system, International Research Journal of Engineering and Technology (IRJET), Volume 08". In : 2021.
- [60] Meikang QIU, Wenyun DAI et Keke GAI. "Mobile Applications Development with Android : Technologies and Algorithms". In : 2016.
- [61] Aparna RAJ et Sujala D. SHETTY. "IoT Eco-system, Layered Architectures, Security and Advancing Technologies : A Comprehensive Survey". In : *Wireless Personal Communications* 122.2 (août 2021), p. 1481-1517.
- [62] Pascal ROQUES. *UML 2 par la pratique*. Eyrolles, 2008.
- [63] Debashis SAHA et Amitava MUKHERJEE. "Pervasive Computing : A Paradigm for the 21st Century". In : *Computer* 36 (2003), p. 25-31.
- [64] Rui SANTOS et Sara SANTOS. *Ultimate Guide Arduino Sensors Modules*. en.
- [65] B.N. SCHILIT et M.M. THEIMER. "Disseminating active map information to mobile hosts". In : *IEEE Network* 8.5 (sept. 1994), p. 22-32.
- [66] Pallavi SETHI et Smruti R. SARANGI. "Internet of Things : Architectures, Protocols, and Applications". In : *Journal of Electrical and Computer Engineering* 2017 (2017), p. 1-25.
- [67] Kwang-chul SHIN, Chulhyun HWANG et Hoekyung JUNG. "DESIGN NoSQL Data Model : The proposal is how to use conceptual data model in NoSQL database design based on". In : 2017.
- [68] Akshay SINGH, Sakshi SHARMA et Shashwat SINGH. "Android Application Development using Android Studio and PHP Framework". In : 2016.
- [71] Thomas STRANG et Claudia LINNHOFF-POPIEN. "A Context Modeling Survey". In : 2004.
- [72] Suriya SUNDARAMOORTHY. "Introduction to UML diagrams and its components". In : *UML Diagramming*. Boca Raton : Auerbach Publications, 2022, p. 1-22.
- [73] Andrew S. TANENBAUM et Maarten van STEEN. "Distributed systems : Principles and Paradigms". In : 2001.
- [74] Mark D. WEISER. "The computer for the 21st century". In : *Human-Computer Interaction* (1991), p. 933-940.
- [75] Miao WU et al. "Research on the architecture of Internet of Things". In : *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICAETE)* 5 (2010), p. V5-484-V5-487.

Webographie

- [4] Lady ADA. *TSL2561 Luminosity Sensor*. <https://cdn-learn.adafruit.com/downloads/pdf/tsl2561.pdf>. Nov. 2021.
- [17] *Concevoir une maison intelligente : les erreurs à éviter*. fr. URL : <http://blog.latrivenetacavi.com/fr/concevoir-une-maison-intelligente-les-erreurs-eviter-2/> (visité le 15/06/2022).
- [18] *ConnectWave : Comment se compose un système IoT ?* fr. URL : <https://www.connectwave.fr/techno-appli-iot/iot/reseaux-et-infrastructures-iot/> (visité le 31/05/2022).
- [19] *Cours Android, Université d'Orléans par Jean-Francois Lalande*. fr. URL : <http://www.univ-orleans.fr/lifo/Members/Jean-Francois.Lalande/enseignement/android/cours-android.pdf>.
- [20] *Créez un backend scalable et performant sur Firebase*. fr. URL : <https://openclassrooms.com/fr/courses/4872916-creez-un-backend-scalable-et-performant-sur-firebase/4982769-creez-votre-premiere-structure-de-donnees>.
- [21] *Définition des diagrammes de classes UML 1.5*. fr. URL : https://docwiki.embarcadero.com/RADStudio/Sydney/fr/D%5C%C3%5C%A9finition_des_diagrammes_de_classes_UML_1.5 (visité le 17/06/2022).
- [25] *Diagrammes de séquence : l'échange de messages dans un système UML*. fr. URL : <https://www.ionos.fr/digitalguide/sites-internet/developpement-web/diagrammes-de-sequence-uml/> (visité le 15/06/2022).
- [26] *Dictionnaires Larousse*. fr. URL : <https://www.larousse.fr/dictionnaires/francais-monolingue> (visité le 31/05/2022).
- [43] *Mobile Applications : Types, Advantages and Disadvantages*. en. URL : <https://gradesfixer.com/free-essay-examples/mobile-applications-types-advantages-and-disadvantages/> (visité le 08/06/2022).
- [48] *Objet Connecté, Orange IoT Journey*. fr. URL : <https://iotjourney.orange.com/fr-FR/explore/les-solutions-iot/objet-connecte> (visité le 07/06/2022).
- [58] *Présentation technique de l'Internet des objets, Centre d'architecture Cloud, Google Cloud*. fr. URL : <https://cloud.google.com/architecture/iot-overview> (visité le 05/06/2022).
- [69] Alexander SOKHANYCH. *Best hotel apps for iOS and Android*. en. URL : <https://thinkmobiles.com/blog/best-hotel-apps/> (visité le 06/06/2022).

- [70] *STI2D : Les Capteurs, Cour en linge.* fr. URL : http://lgt.garnier.free.fr/espace_ee_fichiers/capteurs/capteurs.pdf (visit  le 25/05/2022).