Résumé

L'informatique, une révolution fondamentale et innovante qui a touché la vie humaine durant ce dernier siècle. En Algérie, Internet a prit une grande ampleur dans nos vies quotidiennes privées, contrairement à la vie professionnelle. Ce mémoire porte sur la réalisation d'une plateforme distribuée temps réel, on a pris comme exemple d'application un site pour le suivi des annonces en ligne par des vendeurs. On a 4 types de vendeurs(professionnels, occasionnels), chaque type à sa propre application, avec des bases de données multimédias SQL et NoSQL qui communiqueront avec la base de données Cloud du site des annonces à travers requêtes transférées à l'aide des serveurs Node.js, tout en assurant la sécurité des données.

Mots clés : Plateforme distribuée, Node.js, e-commerce, site des annonces, base de données multimédias, base de données NoSQL,base de données cloud.

Abstract

Computer science, a fundamental and innovative revolution that has touched human life during this last century. In Algeria, the Internet has taken a large scale in our private daily lives, unlike professional life. This thesis focuses on the realization of a distributed platform real time, we took as an example of application a site for tracking online ads by sellers. We have 4 types of vendors (professional, casual), each type to its own application, with SQL and NoSQL multimedia databases that will communicate with the Cloud database of the site ads through requests transferred using servers Node.js, while ensuring data security.

Key words : Distributed platform, Node.js, e-commerce, ad site, multimedia database, NoSQL database, cloud database

Table des matières

Table des figures

Introduction générale

Introduction

Cadre général

L'informatique, une révolution fondamentale et innovante qui a touché la vie humaine durant ce dernier siècle. Internet a pris une grande ampleur dans nos vies quotidiennes, que ça soit privées ou professionnelles, on est de plus en plus nombreux à l'utiliser pour faire diverses recherches, y trouver des informations ou envoyer des courriels.

Dans les pays développés, on peut aussi effectuer des achats, faire des commandes, offrir des services dans le confort de notre maison. Qu'il s'agisse de réserver un vol, commander des livres, consulter les bonnes affaires... les possibilités sont infinies ! Notamment l'achat et/ou la vente en ligne à travers les sites e-commerce.Un service qui a enregistré une révolution exponentielle durant ces dernières années ,le commerce électronique est l'échange pécuniaire de biens, de services et d'informations par l'intermédiaire des réseaux informatiques, notamment Internet. En 1994, le NetMarket était le premier site e-commerce qui a vu le jour, et depuis cette branche ne cesse de s'élargir. En effet plusieurs plateformes avec meilleures de fonctionnalités ont été créées, telles que AMAZON et ALI BABA.

En Algérie, les utilisateurs d'internet sont méfiants devant ce genre de site/application cela du à une émergence retardée dans le monde digital, ce qui a engendré un manque de confiance des gens aux peu de sites existant. Cependant plusieurs plateforme e-commerce ont été créées, telle que JUMIA malgré le manque de service e-paiement, ils ont trouvé une alternative qui est " le paiement à la livraison".

Contributions

Notre travail consiste à concevoir et réaliser une plateforme e-commerce distribuée, sécurisée et temps réel pour le suivi des annonces en ligne:

- Distribuée: Un ensemble d'entités communicantes, installées sur une architecture d'ordinateurs indépendants reliées par un réseau de communication.

- Sécurisée: Elaboration des algorithmes de sécurité personnalisés.

- Temps réel: Le système ne doit pas simplement délivrer des résultats exacts, mais dans des délais imposés.

Pour ce faire, on utilisera l'environnement de développement Node.js, basé sur le moteur V8 de Google chrome, un outil open source créé par Google qui analyse et exécute du code JavaScript très rapidement. Contrairement à la plus part des plateformes qui sont développées en utilisant des CMS(content management system) : wordpress et prestashop ou à travers des services en ligne, tel que shopify.

Organisation du mémoire

Notre mémoire est réparti sur trois chapitres, en suivant cette démarche :

- Chapitre 1 «Les généralités » : Avant de passer à l'implémentation de notre application, il est évident de définir quelques concepts de bases, pour cela nous allons présenter les technologies informatiques ainsi que la sécurité implémentée, qui ont étaient des solutions plus adaptées à la réalisation de notre travail.

- Chapitre 2 «Analyse et conception » : On présentera l'analyse et la conception de l'application, ainsi que les éléments et mécanismes du langage de modélisation utilisés UML.

- Chapitre 3 «Réalisation et implémentation » : Ce chapitre consiste à la réalisation et l'implémentation apportées aux problèmes majeurs évoqués, nous ferons part de tous les outils, langages de programmations, logiciels utilisés afin de répondre et satisfaire les besoins, et enfin nous présenterons les bases de données et les interfaces faites.

- Enfin nous finaliserons par une conclusion générale, sans oublier les bibliographies et les sources d'informations explorées.

En suivant une bonne méthodologie de travail et les étapes de cycle de vie de développement et réalisation d'un logiciel, nous avons pu atteindre une grande partie du cahier de charge.

Plateforme

Chapitre 1

Les généralités

1.1 Introduction

Dans ce premier chapitre, nous verrons les systèmes distribués en général : Dans un premier temps nous allons parler des solutions les plus adaptées dans la structure, la gestion et l'accès aux données dans ce type d'architecture .

Ensuite nous allons voir les différentes technologies ,les outils et langages de développement qui conviennent le plus à ce système. Et enfin, nous allons nous intéresser à la partie délicate dans les systèmes distribués qui est : la sécurité et ses différentes méthodes afin d'assurer la protection de notre système.

1.2 Généralités sur les systèmes distribués

Vue l'augmentation de leurs ressources, les entreprises multi-applications doivent pouvoir compter sur une infrastructure informatique hautes performances, à même d'assurer l'exécution transparente de leurs processus informatiques et une communication fiable, en interne comme en externe . Cela exige une surveillance continue de la disponibilité des ressources et de l'utilisation de la bande passante des réseaux localement distribués. Les systèmes distribués proposent des solutions pour améliorer l'agilité globale du système d'information au travers des architectures orientées services (SOA). Ces systèmes permettent aussi de mettre en place des architectures informatiques permettant d'améliorer les performances ainsi que la disponibilité des systèmes informatiques.

1.2.1 Définition d'un système distribué [11,22,23,24]

- Un système distribué est un système disposant d'un ensemble d'entités communicantes, installées sur une architecture d'ordinateurs indépendants reliés par un réseau de communication, dans le but de résoudre en coopération une fonctionnalité applicative commune.
- Autrement dit, un système distribué est défini comme étant un ensemble des ressources physiques et logiques géographiquement dispersées et reliées par un réseau de communication dans le but de réaliser une tâche commune. Cet ensemble donne aux utilisateurs une vue unique des données du point de vue logique.
- Un système distribué est un ensemble d'entités autonomes de calcul (ordinateurs, PDA, processeurs, processus, processus léger etc.) interconnectées et qui peuvent communiquer.

1.2.2 Intérêt des systèmes distribués

Les systèmes distribués ont plusieurs raisons de leur existence :

- Partage des ressources (données, programmes, services) qui permet un travail collaboratif.
- Accès distant, c'est-à-dire qu'un même service peut être utilisé par plusieurs acteurs situés à des endroits différents.
- Amélioration des performances : la mise en commun de plusieurs unités de calcul permet d'effectuer des calculs parallélisables en des temps plus courts.
- Confidentialité : les données brutes ne sont pas disponibles partout au même moment, seules certaines vues sont exportées.
- Maintien d'une vision unique de la base de données malgré la distribution.
- Réalisation des systèmes à grande capacité d'évolution.
- Augmentation de la fiabilité grâce à la duplication de machines ou de données, ce qui induit à une réalisation des systèmes à haute disponibilité.

1.3 Les bases de données dans les systèmes distribués

1.3.1 Introduction aux bases de données

Une base de données est une collection d'informations organisées afin d'être facilement consultables, gérables et mis à jour. L'histoire des bases de données remonte aux années 1960, avec l'apparition des bases de données réseau et des bases de données hiérarchiques. Dans les années 1980, ce sont les bases de données orienteé objects qui ont fait leur apparition. Aujourd'hui, les bases de données prédominantes sont les SQL, NoSQL et bases de données Cloud.

Dans le cas d'une grande data base, les multiples utilisateurs doivent être en mesure de manipuler les informations qu'elle contient rapidement et n'importe quand.

De plus, les grandes entreprises ont tendance à cumuler de nombreux fichiers indépendants comprenant des fichiers liés ou même des données se superposant. Dans le cadre d'une analyse de données, il est nécessaire que les données en provenance de plusieurs fichiers puissent être liées. C'est pourquoi différents types de bases de données ont été développé pour répondre à ces exigences : orientée texte, hiérarchique, réseau, relationnelle, orientée objet.

Au sein d'une data base, les données sont organisées en lignes, colonnes et tableaux. Elles sont indexées afin de pouvoir facilement trouver les informations recherchées à l'aide d'un logiciel informatique. Elle se charge elle-même de créer, de mettre à jour ou de supprimer des données. Elle effectue également des recherches parmi les données qu'elle contient sur demande de l'utilisateur, et de lancer des applications à partir des données.

Les bases de données sont stockées sous forme de fichiers ou d'ensemble de fichiers sur un disque magnétique, une cassette, un disque optique ou tout autre type d'appareil de stockage. Les bases de données traditionnelles (hiérarchiques) sont organisées par champs (Fields), enregistrements et fichiers. Un champ est une seule pièce d'information. Un enregistrement est un ensemble de champs. Un fichier est une collection d'enregistrements.

1.3.2 Les bases de données relationnelles[7]

Les bases de données relationnelles ont été inventées en 1970 par E.F. Codd de IBM. Il s'agit de documents tabulaires dans laquelle les données sont définies afin d'être accessibles et de pouvoir être réorganisées de différentes manières. Oracle fut la première société à commercialiser un produit conforme à ce concept, dès 1977, avec le succès que l'on connait.

Les bases de données relationnelles sont constituées d'un ensemble de tableaux. Au sein de ces tableaux, les données sont classées par catégorie.
Chaque tableau comporte au moins une colonne correspondante à une catégorie.
Chaque colonne comporte un certain nombre de données correspondantes à cette catégorie.

L'API standard pour les bases de données relationnelles est le Structured Query Language (*SQL*). Les bases de données relationnelles sont facilement extensibles, et de nouvelles catégories de données peuvent être ajoutées après la création de la data base originale sans avoir besoin de modifier toutes les applications existantes.

Exemple

PostgreSQL [5]

C'un système de gestion de bases de données relationnelles objet (*ORDBMS*) développé à l'université de Californie au département des sciences informatiques de Berkeley. *POSTGRES* est à l'origine de nombreux concepts qui ne seront rendus isponibles au sein de systèmes de gestion de bases de données commerciaux que bien plus tard.

C'est un descendant libre du code original de Berkeley. Il supporte une grande partie du standard SQL tout en offrant de nombreuses fonctionnalités modernes : requêtes complexes, clés étrangères, triggers, vues modifiables, intégrité transactionnelle, contrôle des versions concurrentes (*MVCC*, acronyme de « *Multi Version Concurrency Control* »). De plus, PostgreSQL™ peut être étendu par l'utilisateur de multiples façons, en ajoutant,

par exemple :

- de nouveaux : types de données, opérateurs, langages de procédure.
- de nouvelles : fonctions, fonctions d'agrégat, méthodes d'indexage.

Et grâce à sa licence libérale, PostgreSQL peut être utilisé, modifié et distribué librement, quel que soit le but visé, qu'il soit privé, commercial ou académique.

1.3.3 Base de données NoSQL [1,7,20]

Les bases de données NoSQL sont utiles pour les larges ensembles de données distribuées. En effet, les bases de données relationnelles ne sont pas conçues pour le Big Data, et les ensembles de données trop larges peuvent poser des problèmes de performances, avec l'essor du Big Data, les bases de données NoSQL sont de plus en plus utilisées. Leur modèle de données résout par ailleurs beaucoup de problèmes :

- Volumes importants de données structurées, semi-structurées et non-structurées en constante mutation.
- Sprints agiles, itération de schéma rapide et intégrations fréquentes de nouveau code.
- Programmation orientée objet d'une grande flexibilité et simplicité d'utilisation.
- Architectures à scalabilité horizontale reparties géographiquement sur de nombreux sites, en lieu et place des architectures monolithiques coûteuses.

Quatre catégories se ditinguent parmi celles-ci:

- Bases de données Orientées Clé / Valeur :de type clé / valeur s'articulent sur une architecture très basique. On peut les apparenter à une sorte de HashMap, c'est-à-dire qu'une valeur, un nombre ou du texte est stocké grâce à une clé,qui sera le seul moyen d'y accéder.
- Bases de données Orientées Document:sont une évolution des bases de données de type clé-valeur. Ici les clés ne sont plus associées à des valeurs sous forme de bloc binaire mais à un document dont le format n'est pas imposé. Il peut être de plusieurs types différents comme par exemple du JSON ou du XML, pour autant que la base de données soit en mesure de manipuler le format choisit afin de permettre des traitements sur les documents
  - On retrouve principalement MongoDB et CouchBase comme solutions basées sur le concept de base documentaire.

- Bases de données Orientées Colonne:Le principe consiste dans leur stockage par colonne et non par ligne. elles stockent les données de façon à ce que toute les données d'une même colonne soient stockées ensemble. Ces bases peuvent évoluer avec le temps, que ce soit en nombre de lignes ou en nombre de colonnes
  - Les bases les plus connues se basant sur ce concept sont HBase et Cassandra.
- Base de données Orientées Graphe :permettent de résoudre des problèmes très complexes , les réseaux sociaux où des millions d'utilisateurs sont reliés de différentes manières constituent un bon exemple : amis, fans, famille etc. Le défi ici n'est pas le nombre d'élément à gérer, mais le nombre de relations qu'il peut y avoir entre tous ces éléments. En effet, il y a potentiellement n^2 relations à stocker pour n éléments.

Exemple

Mango DB [2,3]

Mongo DB est l'une des bases de données les plus populaires à l'heure actuelle, toutes catégories confondues. Elle permet de prendre en charge aussi bien des données structurées que des données non structurées, et ses performances et sa scalabilité sont excellentes, elle est conçu pour répondre aux demandes des applications modernes avec une base technologique qui vous permet:

- Le modèle de données de document, présente le meilleur moyen de travailler avec des données. Cette catégorie de bases de données repose sur divers formats (JSON, XML) et offre la capacité de changer de schéma sans avoir à arrêter la data base. Les développeurs peuvent télécharger des documents indexés et y accéder par le biais du moteur de stockage de la base de données. La flexibilité de ces bases de données les rend très polyvalentes.
- Une conception de systèmes distribués, permettant de placer intelligemment les données où on le souhaite.
- Une expérience unifiée qui vous donne la liberté d'exécuter n'importe où, vous permettant de pérenniser votre travail et d'éliminer le blocage des fournisseurs.

Avec ces fonctionnalités, vous pouvez créer une plate-forme de données opérationnelles intelligentes, reposant sur Mongo DB.

1.3.4 Base de données Cloud[8]

Une base de données Cloud est un service de base de données construit et auquel on accède via une plateforme Cloud. Sa fonction est similaire à celle d'une base de données traditionnelle, mais elle offre en plus la souplesse du Cloud computing. Une telle base de données s'implémente en installant un logiciel sur une infrastructure Cloud.

Ces bases de données offrent :

- une disponibilité plus élevée.
- Facilité d'accès, les utilisateurs peuvent accéder aux bases de données Cloud depuis virtuellement n'importe où, avec une API fournisseur ou une interface Web.
- Évolutivité, les bases de données Cloud peuvent étendre leurs capacités de stockage à l'exécution en fonction des besoins. L'entreprise ne paye que pour ce qu'elle utilise.
- Reprise après incident, les données sont sécurisées par des sauvegardes sur des serveurs distants. Il n'y a donc pas de risque de perte en cas de catastrophe naturelle, de panne d'un équipement ou de coupure de courant.

Exemple

Cloud Firestore[14]

Firestore est une base de données flexible et évolutive pour le développement mobile, Web et serveur de Firebase et de Google Cloud Platform. Comme la base de données en temps réel Firebase, elle synchronise les données entre les applications clientes et les écouteurs en temps réel. Elle offre également une prise en charge hors ligne pour mobile et Web, de sorte qu'on puisse créer des applications réactives fonctionnant indépendamment de la latence du réseau ou de la connectivité Internet. Cloud Firestore offre également une intégration transparente avec d'autres produits Firebase et Google Cloud Platform, y compris les fonctions cloud

1.3.5 Bases de données multimédia[21]

C'est un type de base de données consacré au stockage,traitement et à l'organisation de données multimédia : documents sonores, images, vidéos.

Une base multimédia a la particularité de représenter les documents intégralement, un SGBD multimédia doit permettre la modélisation d'objets réels mais également de leurs interactions.

Les documents peuvent être classés suivant trois types :

- Statiques : Média dont l'état n'évolue pas durant sa lecture, à l'instar des textes et images.
- Dynamiques : Média qui possède un contenu dont l'état lu évolue au fil du temps. Le son et la vidéo sont les exemples les plus communs.
- Dimensionnels : Objets 3D.

1.4 Les technologies de développement Back-end

1.4.1 Node.js (Node) [6,9,10]

Node.js est une plateforme de développement open source permettant d'exécuter du code JavaScript côté serveur. Node est utile pour développer des applications nécessitant une connexion persistante du navigateur au serveur et est souvent utilisé pour des applications en temps réel telles que la discussion en ligne, les flux de nouvelles et les notifications Web push, il ne bloque pas les entrées / sorties.

Node.js est destiné à être exécuté sur un serveur HTTP dédié et à employer un seul thread avec un processus à la fois. Les applications Node.js sont basées sur des évènements et s'exécutent de manière asynchrone ,le code créé sur la plateforme Node.js ne suit pas le modèle traditionnel de recevoir, traiter, envoyer, attendre. Au lieu de cela, Node traite les requêtes entrantes dans une pile d'évènements constante et envoie les petites requêtes les unes après les autres sans attendre les réponses. Cela nous éloigne des modèles classiques qui exécutent des processus plus volumineux et complexes et exécutent plusieurs threads simultanément , chaque thread attend la réponse appropriée avant de poursuivre.

Parmi les principaux avantages de Node.js, selon son créateur Ryan Dahl :

- Node.js est un Framework open-source sous licence MIT. (La licence MIT est une licence de logiciel libre issue du Massachusetts Institute of Technology (MIT).)
- Utilise JavaScript pour créer une application côté serveur complète.
- Structure légère comprenant des modules à la base minimale. D'autres modules peuvent être inclus selon le besoin d'une application.
- Asynchrone par défaut. Donc, il exécute plus rapidement que d'autres cadres.
- Framework multiplateforme fonctionnant sous Windows, MAC ou Linux

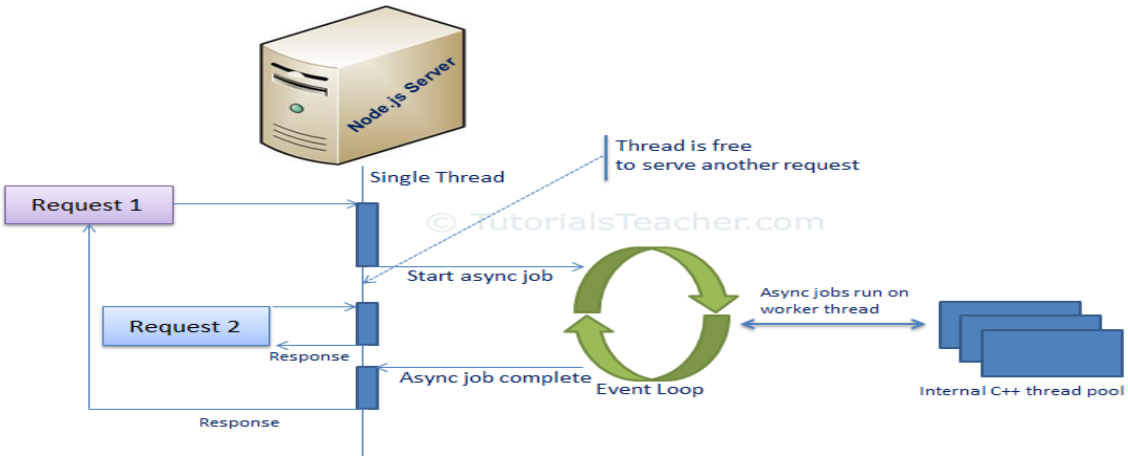


Figure 1.4.1: Modèle de processus Node.js

1.4.2 Modules Node.js

C'est une fonctionnalité simple ou complexe organisée en un ou plusieurs fichiers JavaScript pouvant être réutilisés dans l'application Node.js.

Chaque module dans Node.js a son propre contexte, il ne peut donc pas interférer avec d'autres modules ni polluer la portée globale. De plus, chaque module peut être placé dans un fichier .js séparé dans un dossier séparé.

Les modules implémentés par Node sont ceux de CommonJS standard . (CommonJS est un groupe de volontaires qui définit les normes JavaScript pour les applications de serveur Web, de bureau et de console).

Node.js comprend deux types de modules:

1. Modules de base : incluent les fonctionnalités minimales de Node.js. Ces modules de base sont compilés dans sa distribution binaire et chargés automatiquement au démarrage du processus Node.js. Cependant, on doit d'abord importer le module principal pour pouvoir l'utiliser dans votre application. Exemple : http (pour créer le serveur http Node.js), url (inclut des méthodes de résolution et d'analyse d'URL), chaine de requête, chemin(ou path, gérer les chemins de fichiers),fs(inclut des classes, des méthodes et des évènements permettant de travailler avec les entrées / sorties de fichiers),util(inclut des fonctions utilitaires utiles aux programmeurs)

2. Modules locaux : Ces modules incluent différentes fonctionnalités de l'application dans des fichiers et des dossiers distincts.Exemple, si on doit se connecter à Mongo DB et récupérer des données, on peut créer un module pour celle-ci, qui peut être réutilisé dans l'application.

    - On peut également le conditionner et le distribuer via NPM (Node Package Manager) afin que la communauté Node.js puisse l'utiliser , mais il sera plus un module local.

3. Modules tiers :Tout module Node.js créé par un développeur et mis à la disposition des autres utilisateurs via GitHub et à téléchargement via le NPM de Node.js en l'ajoutant au registre de la page NPM Registry.

1.4.3 Node Package Manager (NPM)

C'est un outil de ligne de commande qui installe, met à jour ou désinstalle les packages Node.js de l'application. C'est également un référentiel en ligne pour les paquets Node.js à code source ouvert. La communauté de nœuds du monde entier crée des modules utiles et les publie sous forme de packages dans ce référentiel, NPM est inclus dans l'installation de Node.js.

1.5 Les technologies de développement Front-end

1.5.1 Html 5 (HyperText Markup Language 5)

Développé par le W3C (World Wide Web Consortium) et le WHATWG (Web Hypertext Application Technology Working Group), une version du célèbre format HTML utilisé pour concevoir les sites Internet. Celui-ci se résume à un langage de balisage qui sert à l'écriture de l'hypertexte indispensable à la mise en forme d'une page Web. Lancée en octobre 2014, cette version HTML5 apporte de nouveaux éléments et de nouveaux attributs par rapport à ses versions précédentes. Elle offre par exemple la possibilité de définir le contenu principal d'une page Web, d'ajouter une introduction en header, d'insérer un sous-titre à un contenu multimédia de type vidéo,... etc.

1.5.2 Les API JavaScript de HTML5

Le HTML5 propose également une série de nouvelles API, pour la plupart sous JavaScript, qui peut être implémentées sur tous les navigateurs. Voici quelques exemples d'API ajoutées pour HTML5 :

- Géo-localisation .
- Activation des contenus audio et vidéo via les balises <audio> et <video>, également nouvelles .
- Applications hors connexion : l'application cache .
- Exécution de tâches parallèles au code de la page. Cette API peut interagir avec le script principal de la page.
- L'édition de contenus, qui fonctionne avec le nouvel attribut <contenteditable> .
- glisser-déposer activée sous l'attribut <draggable> .
- Un web storage étendu, qui ne remplace pas les cookies mais offre une amplitude largement supérieure aux versions précédentes.

1.5.3 JavaScript (js)

C'est un langage de script léger, orienté objet, principalement connu comme le langage de script des pages web, permet à une page web qu'elle soit, entre autres, interactive, dynamique. Mais il est aussi utilisé dans de nombreux environnements extérieurs aux navigateurs web tels que Node.js.

1.5.4 Les feuilles de style en cascade ou CSS 3(Cascading Style Sheets)

C'est un langage de feuille de style utilisé afin de décrire la présentation d'un document écrit en HTML ou en XML, CSS décrit la façon dont les éléments doivent être affichés à l'écran.

1.5.5 Bootstrap 4

C'est une infrastructure frontale gratuite pour un développement Web plus rapide et plus facile, comprend des modèles de conception basés sur HTML et CSS pour la typographie, les formulaires, les boutons, les tableaux, la navigation, les carrousels d'images et de nombreux autres, ainsi que des plugins JavaScript optionnels.

1.5.6 jQuery

C'est une bibliothèque JavaScript rapide, petite et riche en fonctionnalités. Elle simplifie considérablement la navigation et la manipulation de documents HTML, la gestion d'évènements, l'animation et Ajax grâce à une API simple à utiliser qui fonctionne sur une multitude de navigateurs. Avec une combinaison de polyvalence et d'extensibilité, jQuery a changé la façon dont des millions de personnes écrivent JavaScript.

1.6 La sécurité dans les systèmes distribués

Les applications ne peuvent pas compter uniquement sur leurs propres données, elle doit être en mesure d'avoir des applications tierces, entremêler avec d'autres, et avoir ses données facilement accessibles par les développeurs

. L'authentification est une problématique récurrente du développement d'applications et il est important de pouvoir offrir une solution commune performante et sécurisée afin de s'assurer que les informations du système informatique ne puissent pas être volées et lues par quelqu'un qui souhaite les utiliser à des fins malveillantes .

1.6.1 Le chiffrement des données

Dans le monde de l'informatique, le chiffrement est la conversion des données d'un format lisible à un format codé qui peut être lu ou traité qu'après déchiffrement. C'est un bloc de construction de base de la sécurité des données et le moyen le plus simple pour protéger les données sensibles sur leurs serveurs et bases de données. Utilisé à la fois par les utilisateurs individuels et les grandes sociétés, le chiffrement est largement employé sur Internet afin de garantir l'inviolabilité des informations de l'utilisateur transmises entre un navigateur et un serveur. Ces informations peuvent inclure tout type de renseignements, des données de paiement aux informations personnelles. Au-delà de l'utilité évidente de protéger les informations privées contre le vol ou la menace, le chiffrement permet également de prouver que les informations sont authentiques et issues de la source dont elles prétendent venir. Il peut être utilisé pour vérifier l'origine d'un message et pour confirmer qu'il n'a pas été modifié pendant la transmission.

Le principe du chiffrement

Le chiffrement se base sur la notion d'algorithmes de chiffrement et de « *clés* ». Lorsque l'information est envoyée, elle est chiffrée à l'aide d'un algorithme et peut être décodée uniquement à l'aide de la clé appropriée. Une clé peut être stockée sur le système de réception, ou peut être transmise avec les données chiffrées. Plusieurs méthodes sont utilisées pour coder et décoder les informations. Ces méthodes se développent parallèlement à l'évolution perpétuelle des logiciels informatiques et des méthodes permettant d'intercepter et de voler les informations :

- Clé de chiffrement symétrique : également connue sous le nom d'algorithme à clé secrète, c'est une méthode de décodage unique qui doit être fournie au destinataire avant que le message ne puisse être déchiffré. La clé utilisée pour encoder dite « symétrique » est la même que celle utilisée pour décoder, ce qui convient

parfaitement pour les utilisateurs individuels et les systèmes fermés. Autrement, la clé doit être envoyée au destinataire, ce qui augmente le risque de compromission si elle est interceptée par un tiers (un pirate par exemple). L'avantage de cette méthode est qu'elle est beaucoup plus rapide que la méthode asymétrique.

- Algorithmes de chiffrement symétriques : Il existe deux types :

1. *Chiffrement par bloc :* division du texte clair en blocs fixe, puis chiffrement bloc par bloc, exemples : DES (IBM, Standard NIST 1976) ; IDEA (Xuejia Lai et James Massey en 1992) ; AES (Joan Daemen et Vincent Rijmen 2000).
2. *Chiffrement par flux :* le bloc a une dimension unitaire (1 bit, 1 octet, ...), ou une taille relativement petite, exemples : RC4 (Ron Rivest 1987) ; SEAL (Don Coppersmith et Phillip Rogaway pour IBM 1993).

- Cryptographie asymétrique : cette méthode utilise deux clés différentes (publique et privée) mathématiquement reliées. Concrètement, les clés se composent uniquement de grands nombres qui ont été couplés entre eux mais ne sont pas identiques, d'où le terme « asymétrique ». La clé publique peut être partagée avec tout le monde, mais la clé privée doit rester secrète. Les deux peuvent être utilisées pour chiffrer un message, et la clé opposée à celle utilisée à l'origine pour chiffrer ce message est ensuite utilisée pour le décoder.

  - Algorithmes de chiffrement asymétrique :

1. *RSA:* Rivest, Shamir et Adleman en 1978.
2. *Diffie et Hellman* en 1976.
3. ECC: Elliptic Curve Cryptography.

1.6.2 Proxy [12]

Un serveur proxy désigne un intermédiaire, un mandataire chargé d'établir la liaison entre un réseau local privé et le réseau internet. On parle également dans cette situation de proxy web. Sur Internet, il existe différents types de proxy, les plus courants sont les proxy http, ils supportent les protocoles HTTP et FTP.

Un proxy peut avoir plusieurs utilisations :

- Le proxy peut protéger : il peut autoriser une connexion à l'extérieur et interdire les ordinateurs d'Internet de venir se connecter sur l'appareil . Cette fonction de protection du proxy est souvent incluse dans les firewalls (murs de feu), des ordinateurs programmés pour filtrer les communications entre les réseaux (par exemple entre le réseau d'une entreprise et Internet).
- Le proxy peut masquer les informations concernant un ordinateur: en effet, quand on surfe, tous les sites Web peuvent savoir de quel site qu'on vient, quel navigateur qu'on utilise, quel est le système d'exploitation, l'adresse IP... Certains proxy masquent ces informations. Ces proxy sont dits proxy anonymes.
- Le proxy peut mémoriser les pages les plus demandées : Ainsi si on demande plusieurs fois la page http://www.yahoo.com/index.html, le proxy la donnera immédiatement sans aller la chercher sur
  "www.yahoo.com". Si on est proche du proxy, cela peut accélérer les choses. Il s'appelle alors proxy-cache.

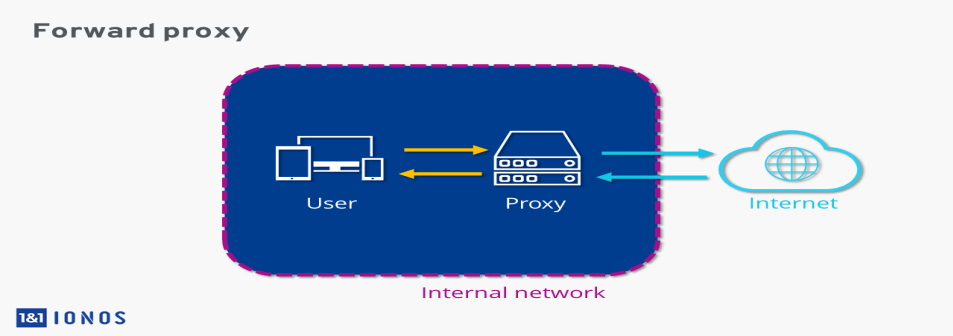


Figure 1.6.1: Forward-Proxy

1.6.2.1 Proxy Reverse [21]

Tandis qu'un Forward-proxy protège les appareils clients des mauvaises influences du réseau public, un reverse-proxy lui opère exactement dans le sens inverse, d'où son nom. Ce système sert ainsi de sécurité supplémentaire et protège les serveurs Web. Les requêtes sont reçues par le proxy qui vérifie si elles sont conformes aux règles de sécurité de configuration. En cas de doute, ces dernières sont transmises au serveur en arrière-plan.

Les serveurs reverse-proxy sont en général protégés par un pare-feu dans un réseau interne ou une zone démilitarisée (DMZ). Le proxy inverse, similaire au proxy-forward, est la seule connexion entre Internet et le réseau privé. Toutes les requêtes du serveur d'arrière-plan au réseau local passent par la même interface de communication avant d'être transférées aux systèmes ciblés. Cette mise en commun permet ainsi de contrôler les données du trafic entrant et autorise divers serveurs sous le même URL pour distribuer les requêtes de manière uniforme sur différents serveurs et ainsi d'accélérer la récupération des données par l'utilisation de la mémoire cache ou caching.

Ci-dessous les différentes applications d'un serveur reverse-proxy :

- Anonymisation : Etant le seul accès au réseau interne, le reverse-proxy intercepte ainsi toutes les requêtes vers le serveur d'arrière-plan et agit de telle sorte que les programmes clients pensent qu'ils sont directement en lien avec le système ciblé actuel. Pour cela, le proxy transfère les requêtes aux systèmes ciblés correspondants dans le réseau local, acceptant leurs réponses puis les transférant aux clients. Le serveur d'arrière-plan reste donc anonyme.

- Encodage et protection : En amont, un reverse-proxy offre la possibilité d'installer des systèmes de contrôle comme les scanners de virus ou les filtres de paquets, ce qui renforce la protection des serveurs qui sont en arrière-plan. Un serveur reverse-proxy peut aussi être utilisé pour l'encodage.

- Répartition de charge : Grâce à l'utilisation en amont d'un proxy inverse, il est possible de lier une URL de différents serveurs sur le réseau privé. Ceci permet de distribuer les requêtes entrantes à de multiples serveurs. La répartition de charge, empêche la surcharge d'un système et fonctionne en cas de défaillance d'un serveur.

- Caching : Afin d'accélérer la vitesse du service, le reverse-proxy apporte une fonction de cache, qui autorise à cacher des réponses du serveur. Cette mise en cache permet ainsi au serveur proxy de répondre aux requêtes répétitives, de manière partielle ou complète. Du contenu statique comme des images ou des pages Web fréquemment consultées par les utilisateurs du réseau local sont gardés en mémoire (en « cache ») par le proxy. Cela permet ainsi d'une part de réduire l'utilisation de la bande passante vers le Web et de réduire le temps d'accès aux

documents pour les utilisateurs. Toutefois comme les contenus changent rapidement il est nécessaire que le proxy compare régulièrement les données qu'il stocke en mémoire cache avec les données distantes pour s'assurer que les données ne sont pas périmées.

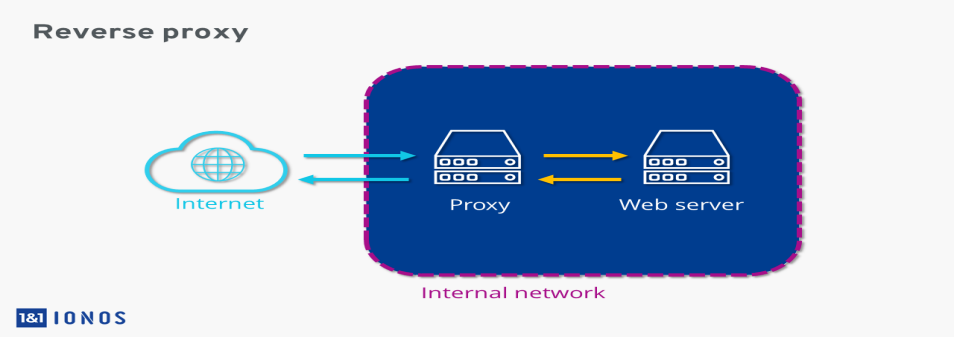


Figure 1.6.2: Reverse-Proxy

1.6.3 Le principe du JSON WEB TOKEN « JWT » [25,26]

Après l'authentification d'un utilisateur, le serveur va générer un token qui est un hash de plusieurs caractères encodé en base 64. Ce token servira de signature et transitera dans chaque requête entre le client et le serveur afin de vérifier l'intégrité de l'utilisateur.

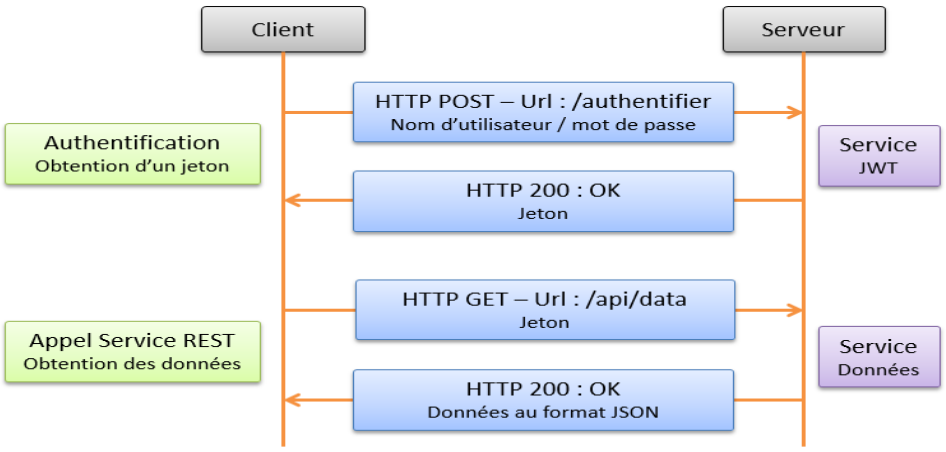


Figure 1.6.3: *Principe du JWT*

Structure d'un token JWT

Il est composé de trois parties:

- Header (ou en-tête) : est un document au format JSON, qui est encodé en base 64 et qui contient deux parties :
  - Le type du token
  - L'algorithme de chiffrement à utiliser pour hasher le payload
- Payload (ou contenu) : est un document au format JSON, qui est encodé en base 64 et qui contient les informations à échanger entre le client et le serveur. Ces informations sont appelées claims ou revendications. En règle générale, on fait transiter des informations sur l'identité de l'utilisateur, mais il ne doit absolument pas contenir de données sensibles. On distingue trois types de claims :
  - *Les claims réservés* : Il s'agit de nom réservé et ne pouvant pas être utilisés par le développeur. Ils contiennent des informations concernant le token lui-même. Exemples : *iss* (L'origine du token) ; *sub* (Le sujet du token) ; *exp* (Définie l'expiration du token) ; *iat* (Date création du token).
  - *Les claims publics* : Il s'agit de noms personnalisés que l'on créés et qui sont propres à nos besoins.
  - *Les claims privés* : Il s'agit de noms à usage privé pour répondre à des besoins spécifiques à vos applications. Ils ne doivent pas entrer en conflit avec les autres types de claims.
- Signature : La signature est composée d'un hash qui composé à son tour des éléments suivant :
  - Header
  - Payload
  - Secret

  Le secret est une signature détenue par le serveur. C'est de cette façon que notre serveur sera capable de vérifier les tokens existant et d'en signer des nouveaux. JWT permet donc d'échanger du contenu pour un utilisateur authentifié grâce à

la clé secrète utilisée dans la signature. La signature permet également d'assurer l'intégrité du contenu.

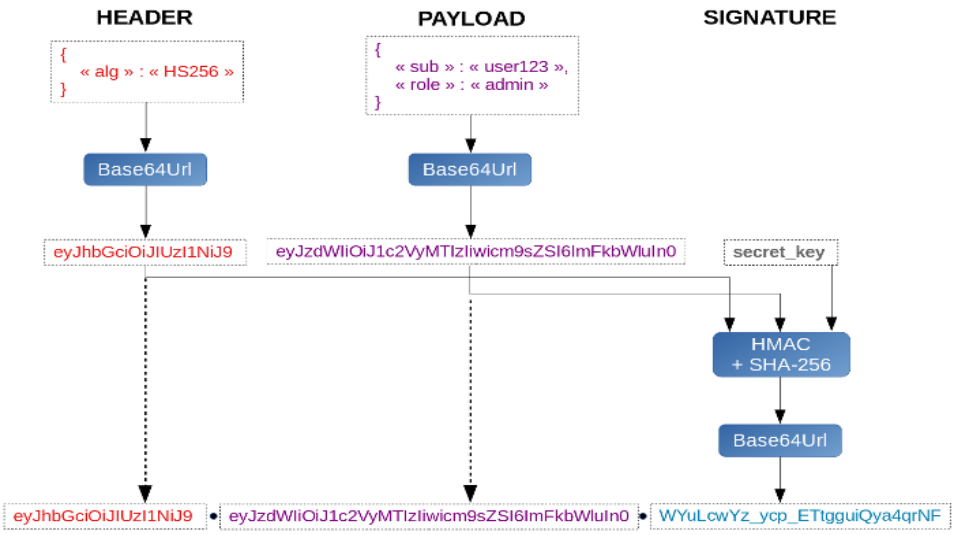


Figure 1.6.4: *Exemple d'un token JWT*

La norme JSON Web Token peut être utilisée dans plusieurs langues et est rapidement et facilement interchangeables. Vous pouvez utiliser le jeton dans une URL, le paramètre POST, ou un en-tête HTTP. La polyvalence du Web Token JSON nous permet de nous authentifier à une API rapidement et facilement en passant par le jeton.

1.6.4 les cookies[27]

Les cookies sont de simples fichiers texte créés dans l'objectif principal d'aider le navigateur à gérer les fonctionnalités spéciales des sites web qui utilisent des cookies. Les cookies aident les serveurs de sites web à se rappeler de de l'utilisateur lorsque qu'il passe d'une page à une autre. Cette simple fonctionnalité rend le e-commerce possible car on a pas à recharger le panier d'achats à chaque fois qu'on quitte une page.

Les cookies sont utilisés aux fins suivantes :

1. Protection – s'assurer qu'on est qui on affirme être et non une autre personne qui a réussi à obtenir une copie de mot de passe.

2. Déterminer rapidement l'identité d'une page à une autre et se souvenir des articles qu'on a mit dans le panier d'achat. Cette fonctionnalité est essentielle pour n'importe

quel type de e-commerce.

3. Paramètres - les cookies aident le site web que'on visite à se souvenir des paramètres qu'on a configurés lors d'une visite précédente. Cela inclut le fait de se rappeler des préférences en termes de thèmes et de langues, ainsi que des identifiants et des mots de passe pour faciliter l'entrée sur le site lors de visites futures.

4. Limiter les publicités - les cookies empêchent les scripts publicitaires d'afficher des pop-up publicitaires ennuyeuses encore et encore. Ils se rappellent également des pages qu'on a consultées précédemment afin qu'on ne voit pas les publicités destinées aux visiteurs qui viennent sur ces pages pour la première fois au cours d'une session.

1.6.5 Cron jobs[28]

C'est un programme qui permet aux utilisateurs d'exécuter automatiquement des scripts, des commandes ou des logiciels à une date et une heure spécifiées à l'avance, ou selon un cycle défini à l'avance.

Il s'agit d'une fonctionnalité très utile pour des tâches routinières d'admini-stration système, mais elle peut très bien être exploitée pour tout autre chose. Par exemple, cron peut jouer un fichier ogg tous les jours à sept heures sauf le samedi et le dimanche afin de se réveiller en musique.

cron est un daemon, ce qui, dans le jargon informatique, désigne un programme qui s'éxécute en arrière-plan. Le service cron (crond) attend ainsi jusqu'au moment spécifié dans le fichier de configuration puis effectue l'action correspondante et se rendort jusqu'à l'événement suivant.

1.6.6 Les sockets[29,30]

Les sockets servent à communiquer entre deux hôtes appelés Client / Serveur à l'aide d'une adresse IP et d'un port ; ces sockets permettront de gérer des flux entrant et sortant afin d'assurer une communication entre le client et le serveur, soit de manière fiable à l'aide du protocole TCP/IP (dans ce mode de communication, une connexion durable est établie entre les deux processus, de telle façon que l'adresse de destination n'est pas nécessaire à chaque envoi de données), soit non fiable mais plus rapide avec le protocole UDP(e mode nécessite l'adresse de destination à chaque envoi, et aucun accusé de réception n'est donné).

Socket.io:

est un module de Node.js qui permet de créer des Web Sockets, c'est-à-dire des connections bi-directionnelles entre clients et serveur qui permettent une communication en temps réel sur un autre protocole que le protocole http normalement utilisé dans les pages web. Ce type de technique est utilisée pour créer des applications telles que des systèmes de communication en temps réel (i.e. des Chats), des jeux multi-utilisateur, des applications de collaboration,.. etc.

1.6.7 Un certificat Secure Socket Layer (SSL) [31]

Permet d'assurer la confidentialité des données échangées entre un utilisateur (Internaute ou application / logiciel) et une application (site Internet, service de messagerie / e-mail).

Cette confidentialité est assurée par un chiffrement des données échangées entre l'utilisateur et l'application. Elle évite toute interception en "clair" des informations communiquées.

Un certificat SSL est un fichier qui contient diverses données :

- Une clé cryptographique dite "publique" qui est liée à une seconde clé cryptographique dite "privée" (installée sur le serveur qui héberge le site ou l'application).
- La ou les adresse(s) sécurisée(s) (www.certificat-ssl.info, par exemple) .
- Et pour un certificat OV ou EV, la raison sociale de l'entreprise / organisation propriétaire de l'adresse sécurisée ; C'est ce certificat qui permet l'initialisation de la connexion sécurisée visible sur le navigateur par la présence du HTTPS et par l'affichage d'un cadenas de sécurité.

Les 4 types de certificats SSL :

- Le certificat SSL EV (Extended Validation) : identité de l'entreprise propriétaire du site Internet est inscrite à l'intérieur de celui-ci, elle est préalablement vérifiée par l'Autorité de Certification qui se doit de respecter des règles simples, mais très strictes. Ces vérifications visent à s'assurer que le demandeur du SSL EV est bien

propriétaire du site Internet qu'il souhaite sécuriser : Un pirate informatique ne pourra donc pas obtenir un certificat SSL pour "www.nom-de-votre-banque.fr".

Le SSL EV procure un niveau d'identification le plus élevé possible : C'est pour cette raison qu'il s'appelle EV (à Validation Etendue/ Extended Validation). En plus du chiffrement SSL, la raison sociale est affichée dans la barre d'adresse URL qui devient verte. Le cadenas de sécurité s'active, tout comme le protocole HTTPS.

- Le certificat SSL OV (Organization Validation) : Le certificat SSL OV (validation de l'organisation) est similaire au EV. Il se distingue par le fait que la raison sociale n'est pas affichée dans la barre d'adresse URL mais uniquement à l'intérieur du certificat SSL. Il n'active donc pas la barre d'adresse verte.

- Le certificat SSL DV (Domain Validation) : Les AC (Autorités de Certification) ne vérifient pas l'identité du propriétaire du site Internet. Ils s'assurent simplement que le titulaire du nom de domaine à sécuriser a bien donné son accord pour l'émission d'un SSL DV pour cette adresse. Cette vérification s'effectue par l'envoi d'un e-mail d'approbation (simple lien sur lequel l'utilisateur doit cliquer), ou par l'upload d'un fichier à la racine du nom de domaine.L'identité n'étant pas vérifiée, celle-ci n'est ni affichée dans le certificat, ni affichée dans la barre d'adresse. Ce certificat active néanmoins le cadenas de sécurité et permet bien entendu l'utilisation du protocole HTTPS.

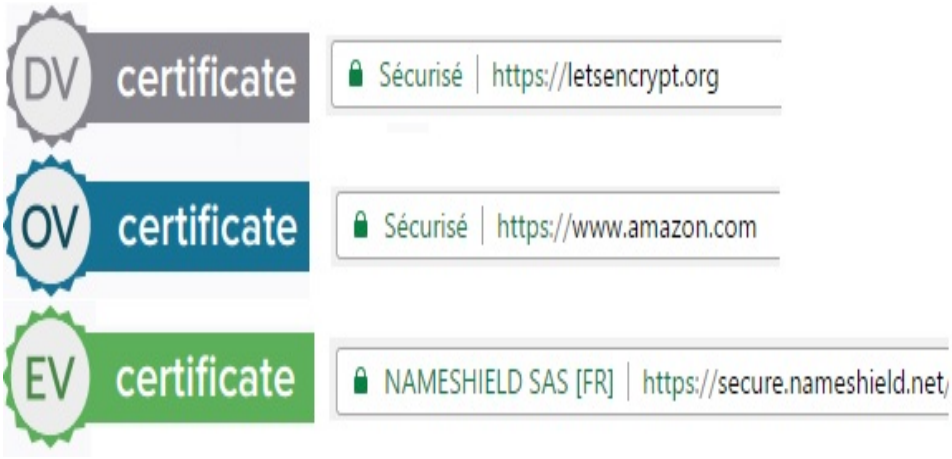


Figure 1.6.5: *Différence entre les certificats SSL :DV/OV/EV*

- Le certificat SSL auto signé : Un certificat auto signé est un certificat SSL gratuit qui est signé par le serveur qui l'a édité. Il n'est donc pas émis par une Autorité de Certification. Par conséquence, il affiche une erreur de sécurité sur tous les navigateurs web, ce qui pousse les visiteurs à quitter la page.

Figure 1.6.6: *Certificat SSL auto signé*

1.6.8 Pare-feu d'applications Web (WAF)[31]

C'est un pare-feu qui surveille, filtre ou bloque le trafic HTTP depuis et vers une application Web.Exécuté sous la forme d'une appliance, d'un module d'extension serveur (ou plug-in serveur) ou d'un service en Cloud, un WAF inspecte chaque paquet de données HTML, HTTPS, SOAP et XML-RPC.

Un procédé d'inspection personnalisable lui permet d'empêcher des attaques , les injections SQL, le piratage de session et le débordement de capacité de tampon ; des attaques que les pare-feu réseau et les systèmes de détection des intrusions sont souvent incapables de contrer.

Un pare-feu d'applications Web peut également détecter et empêcher de nouvelles attaques inconnues en surveillant des schémas inhabituels au sein du trafic de données,il peut reposer sur le réseau ou sur un hôte. Il est généralement déployé via un proxy et installé devant une ou plusieurs applications Web. En temps réel ou en quasi-temps réel, il surveille alors le trafic avant que celui-ci n'atteigne l'application Web, et analyse

toutes les requêtes grâce à une base de règles, afin d'éliminer par filtrage tout trafic ou schéma de trafic potentiellement préjudiciable.

1.6.9 Réseau Privé Virtuel (VPN) [13]

En temps normal, quand on surfe sur internet, on est sur ce qu'on appelle un « réseau commun ». Autrement dit, on y laisse des informations, qui peuvent être interceptées d'une manière ou d'une autre, notamment par notre fournisseur d'accès internet, par les gouvernements ou par des hackers.

VPN est l'abréviation anglaise de « Virtual Private Network » , un groupe d'ordinateurs ou de réseaux discrets mis en place sur un réseau publique , qui sécurisent la connexion internet de votre ordinateur. On choisit ainsi un VPN lorsque l'on cherche à gagner en anonymat sur internet, en ne laissant ainsi aucune trace de la navigation grâce à l'absence totale de conservation des logs des meilleurs fournisseurs du marché. L'utilisateur qui va se connecter sur internet avec un Réseau Privé Virtuel pourra donc facilement crypter ses données (il existe plusieurs protocoles, comme le PPTP, OpenVPN, L2TP/IPSec, IKEVv2,...).

Figure 1.6.7: *Réseau Privé Virtuel*

1.7 Conclusion

Dans la première partie de ce chapitre nous nous sommes intéressés à la définition des différents outils de gestion et stockage d'informations, . Dans la deuxième partie nous avons présenté les différentes technologies: coté front-end et back-end, et enfin dans la dernière partie nous l'avons consacrée à la sécurité informatique et les différente méthodes qui assurent la protection des systèmes distribués .

Chapitre 2

Conception et modélisation

2.1 Introduction

Afin de faciliter la réalisation d'une application ou un système informatique, il convient de suivre une démarche méthodologique et rigoureuse qui est l'analyse et la conception, c'est une étape qui nous permet d'étudier un problème, lister les résultats attendus et enfin apporter une solution qui peut être informatisée. Dans ce chapitre, nous allons analyser, concevoir et définir le formalisme de modélisation utilisé : l'UML (Unified Modeling Language) pour pouvoir réaliser notre application web .

2.2 Eléments et mécanismes généraux d'UML [18]

2.2.1 Définition d'UML :

UML, c'est l'acronyme anglais pour « Unified Modeling Language ». On le traduit par « Langage de modélisation unifié ». La notation UML est un langage visuel constitué d'un ensemble de schémas, appelés des diagrammes, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par le logiciel.UML est surtout utilisé lorsqu'on prévoit de développer des applications avec une démarche objet (développement en Java, en C++,Node js ...

etc.).

Pour réaliser ces diagrammes revient donc à modéliser les besoins du logiciel à développer.

2.2.2 Composants et utilisation :

Utilisation :

UML est utilisé pour spécifier, visualiser, modifier et construire les documents nécessaires au bon développement d'un logiciel orienté objet. UML offre un standard de modélisation, pour représenter l'architecture logicielle. Les différents éléments représentables sont :

- Activité d'un objet/logiciel : une activité définit un comportement décrit par un séquencement organisé d'unités dont les éléments simples sont les actions.

- Acteurs : représente l'abstraction d'un rôle joué par des entités externes qui interagissent directement avec le système étudié.

- Processus : décrit une approche du développement logiciel. Il définit une séquence d'étapes, en partie ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. Deux points de vue sont considérés dans un processus : la production de logiciels de qualité qui répondent aux besoins des utilisateurs (c'est le point de vue technique). Le point de vue de la planification et de l'estimation de coûts consiste à imposer des délais et des coûts raisonnables.

- Schéma de base de données : il s'agit de passer du monde réel, complexe et confus, au monde informatique ou les structures et les propriétés des objets doivent être identifiées. Ce schéma permet de décrire la structure d'une base de données, en décrivant l'ensemble des types de données de la base.

- Composants logiciels: un élément logiciel remplaçable et réutilisable qui fourni ou reçoit un service bien précis. Il peut être vu comme une pièce détachée du logiciel. Exemple:Les plugins, les drivers, les bibliothèques sont des composants.

Composants:

Diagrammes de structure ou diagrammes statiques :

- Diagramme de classes : représentation des classes intervenantes dans le système. La classe est un concept abstrait qui permet de représenter toutes les entités d'un système.
- Diagramme de composants : représentation des composants du système d'un point de vue physique, tels qu'ils sont mis en œuvre (fichiers, bibliothèques, bases de données...) .
- Diagramme de déploiement : représentation des éléments matériels (ordinateurs, périphériques, réseaux, systèmes de stockage...) et la manière dont les composants du système sont répartis sur ces éléments matériels et interagissent entre eux.
- Diagramme de contexte: délimite le domaine d'étude en précisant ce qui est à la charge du système et en identifiant l'environnement extérieur au système étudié avec lequel ce dernier communique. Ses composants sont : les acteurs externes,le processus unique symbolisant le système d'information étudié , échange entre le système étudié et son environnement.

Diagrammes de comportement :

- Diagramme des cas d'utilisation : représentation des possibilités d'interaction entre le système et les acteurs (intervenants extérieurs au système), c'est-à-dire de toutes les fonctionnalités que doit fournir le système.
- Diagramme d'activité : représentation sous forme de flux ou d'enchaînement d'activités le comportement du système ou de ses composants.

Diagrammes d'interaction ou diagrammes dynamiques :

- Diagramme de séquence : représentation de façon séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou de ses ac-

teurs.

- Diagramme de communication : représentation de façon simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets.
- Diagramme de temps : représentation des variations d'une donnée au cours du temps.

2.3 L'analyse

2.3.1 Logiciel de modélisation utilisé :

StarUML :

Un logiciel de modélisation UML. Étant simple d'utilisation, nécessitant peu de ressources système, supportant UML 2, ce logiciel constitue une excellente option pour une familiarisation à la modélisation. Prise en charge de nombreux raccourcis dans Quick Edit pour créer des éléments et des relations à la fois, tels que des sous-classes, des interfaces de support,... etc. Les diagrammes peuvent être exportés au format PDF,PNG pour une impression nette avec des options d'impression telles que la mise en page et les tailles.

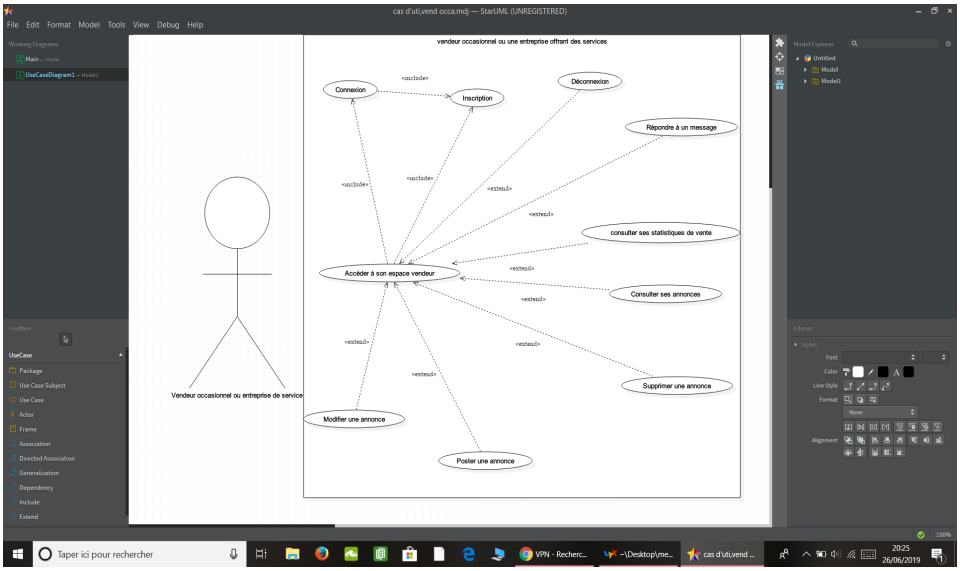


Figure 2.3.1: Le logiciel de modélisation UML

2.3.1.1 Le diagramme de contexte:

Les acteurs :

Notre système nécessite cinq acteurs :

- Vendeur Permanent : c'est le vendeur qui possède un magasin /une entreprise, il peut poster des annonces et faire la gestion des ses produits à l'aide du logiciel de gestion de stock fourni avec l'application, comme il a la possibilité de poster des annonces libres en dehors des produits stockés dans le logiciel.

- Vendeur Occasionnel : c'est une personne quelconque qui veut vendre des objets neufs ou d'occasion sur le site.

- Vendeur de Service : c'est un vendeur de biens immatériels qui poste son travail et ses réalisations sur le site.

- Entreprise de Service: c'est une entreprise qui vend des biens immatériels sur le site.

- Client : c'est une personne qui va avoir un compte, consulte des annonces, contacte le vendeur, achète des produits sur l'application.

Les différents acteurs sont représentés dans le diagramme de contexte sur la figure 10 ci-dessus:

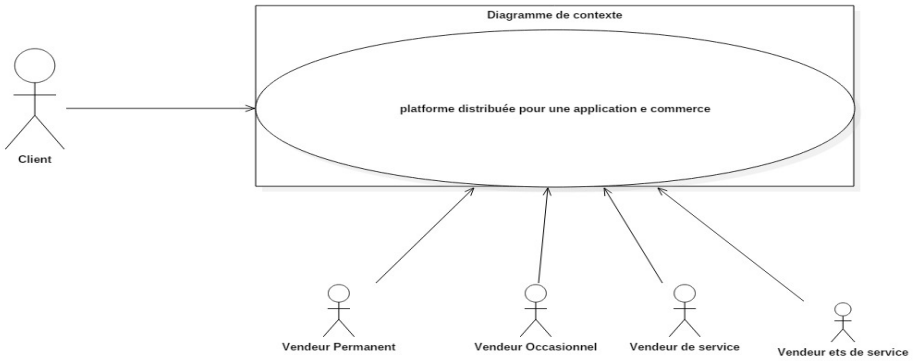


Figure 2.3.2: *Diagramme de contexte*

2.3.1.2 Le diagramme de cas d'utilisation général:

- Général:

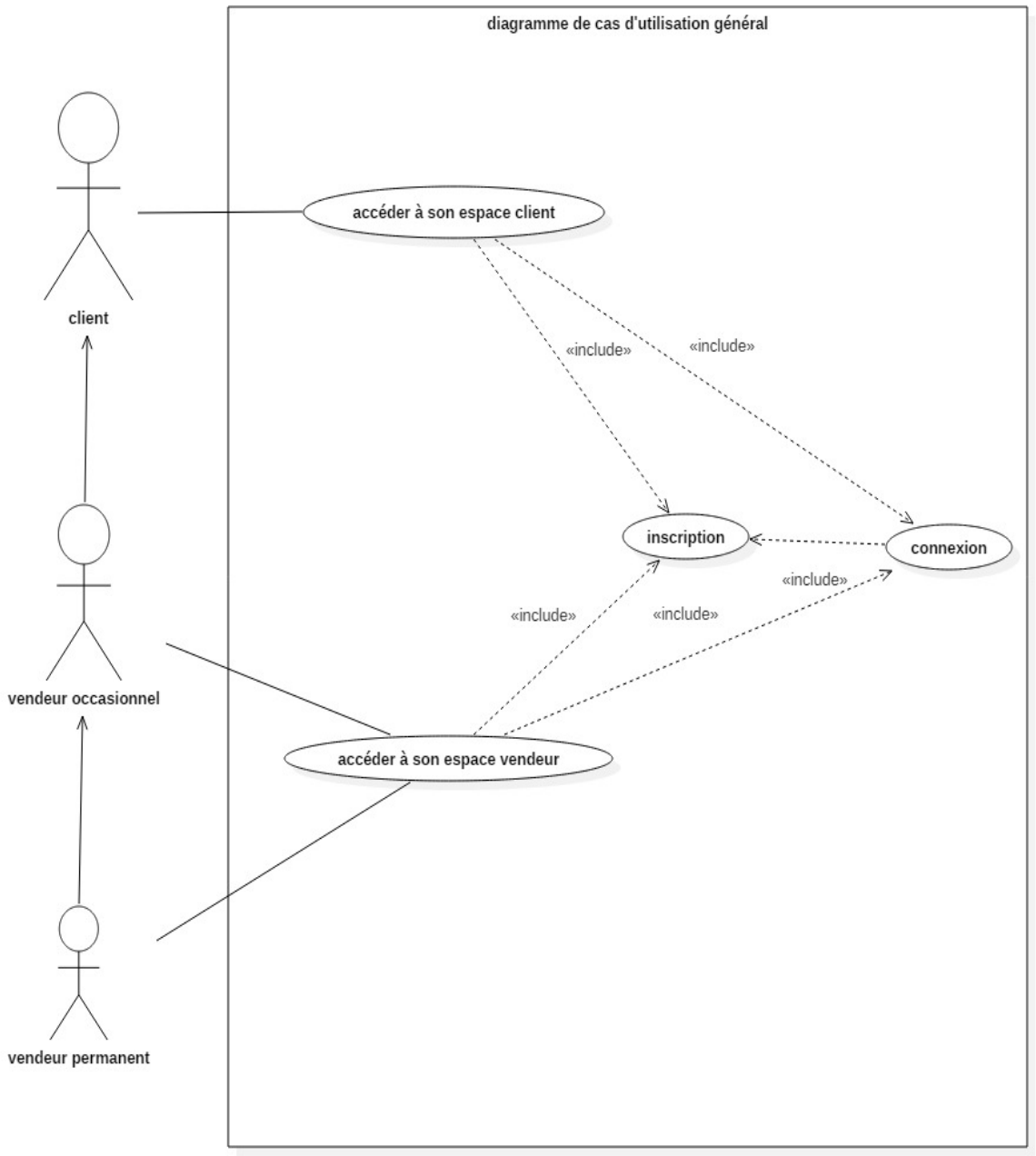


Figure 2.3.3: *Diagramme de cas d'utilisation « général »*

2.3.1.3 Les diagrammes de cas d'utilisation détaillés :

- Vendeur Permanent :

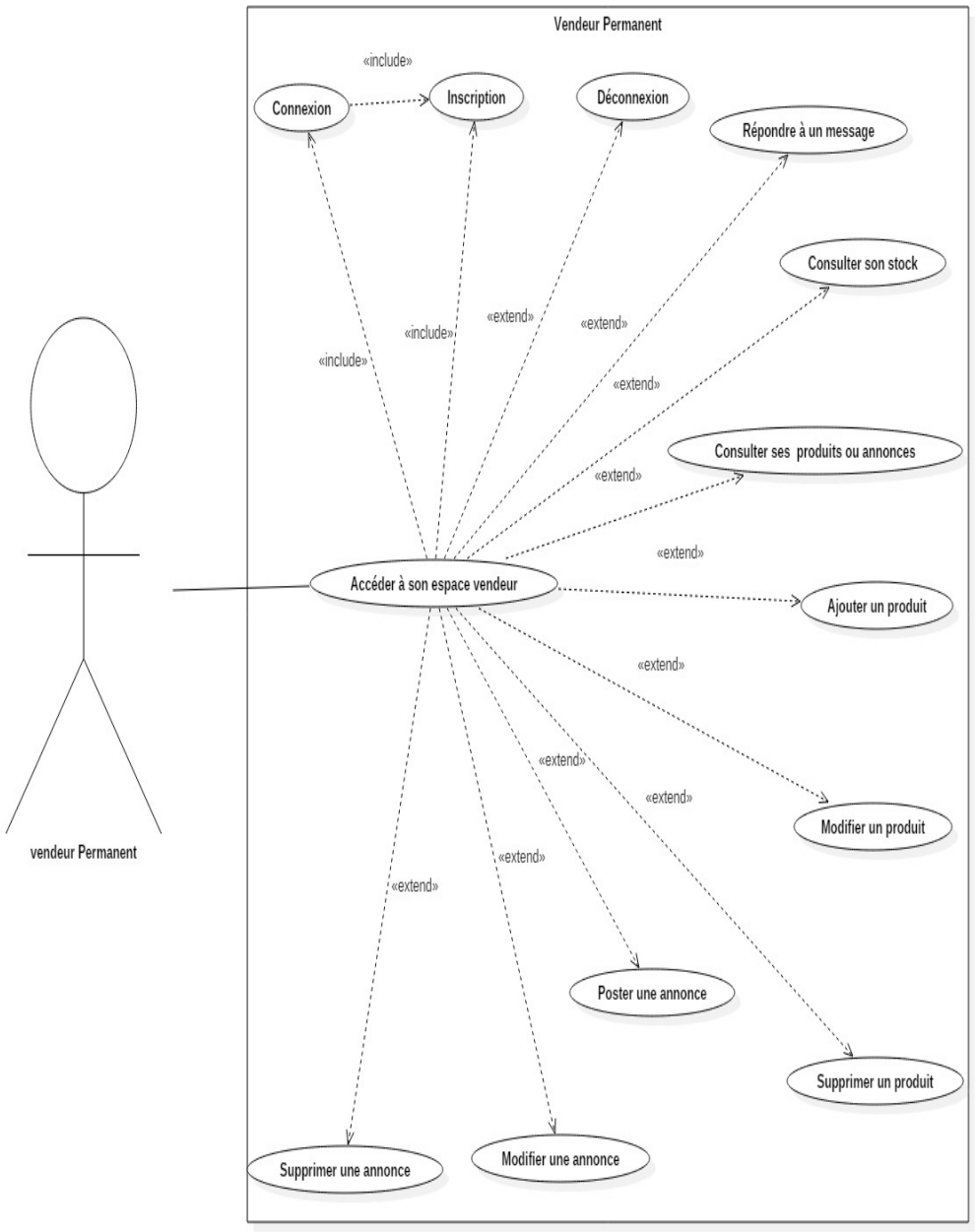


Figure 2.3.4: *Diagramme de cas d'utilisation « vendeur permanent »*

- Vendeur occasionnel / Service / Entreprise de service:

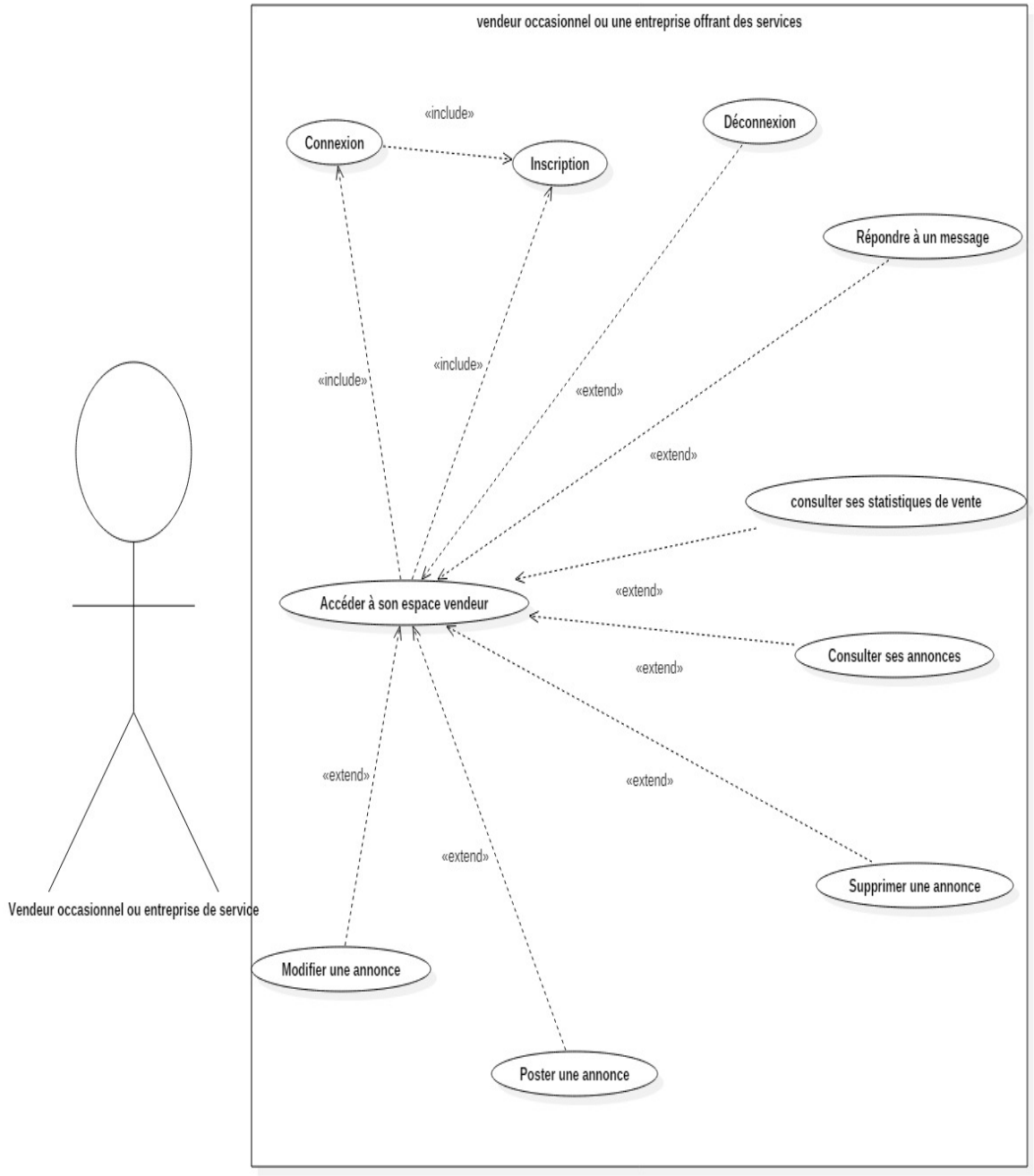


Figure 2.3.5: *Diagramme de cas d'utilisation « vendeur occasionnel »*

- client:

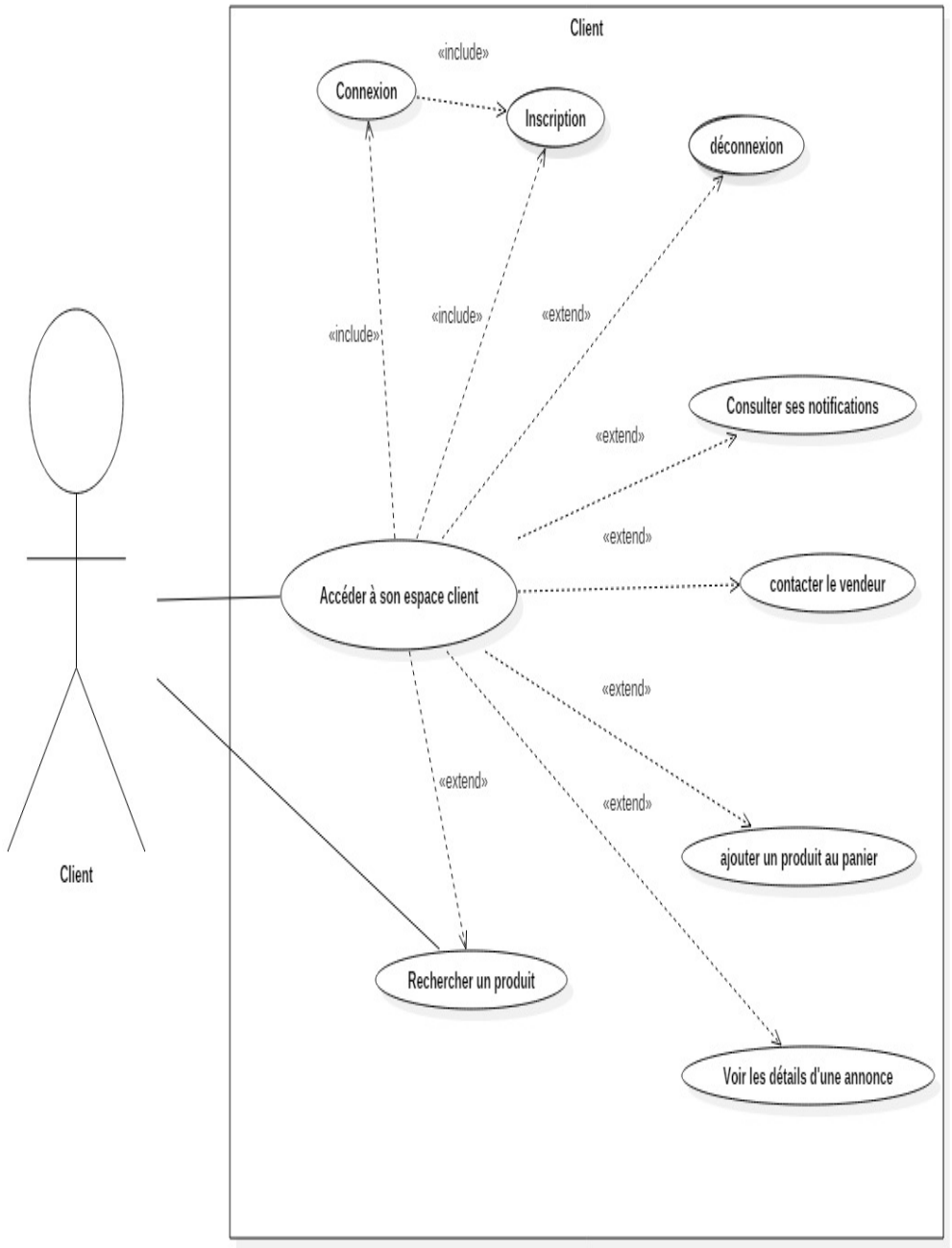


Figure 2.3.6: *Diagramme de cas d'utilisation « client »*

2.3.1.4 Les diagrammes d'activité:

Vendeur Permanent :

- Inscription du vendeur :

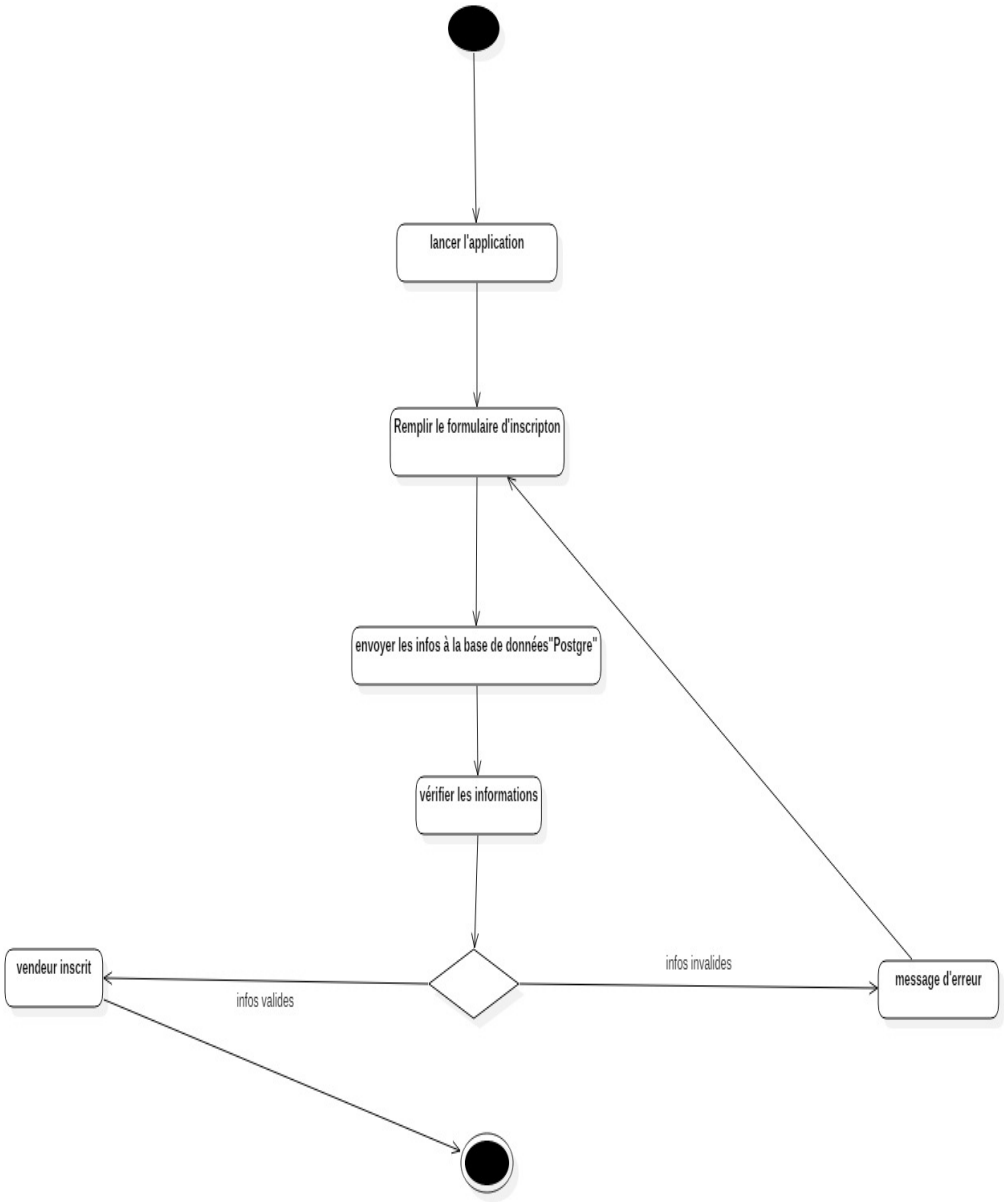


Figure 2.3.7: Diagramme d'activité *«Inscription du vendeur Permanent »*

- Connexion du vendeur :

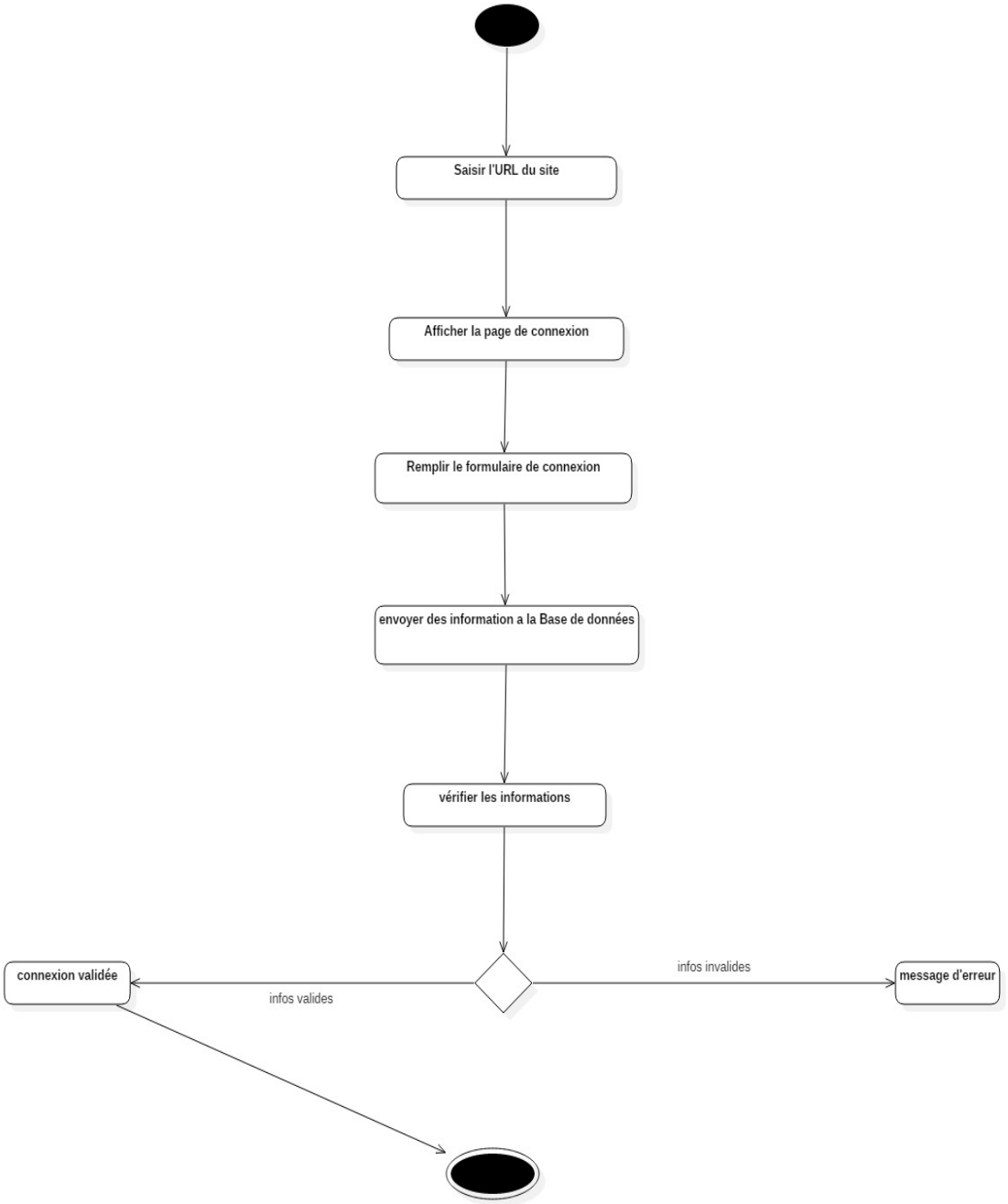


Figure 2.3.8: *diagramme d'activité « connexion du vendeur Permanent »*

- Ajouter un produit :

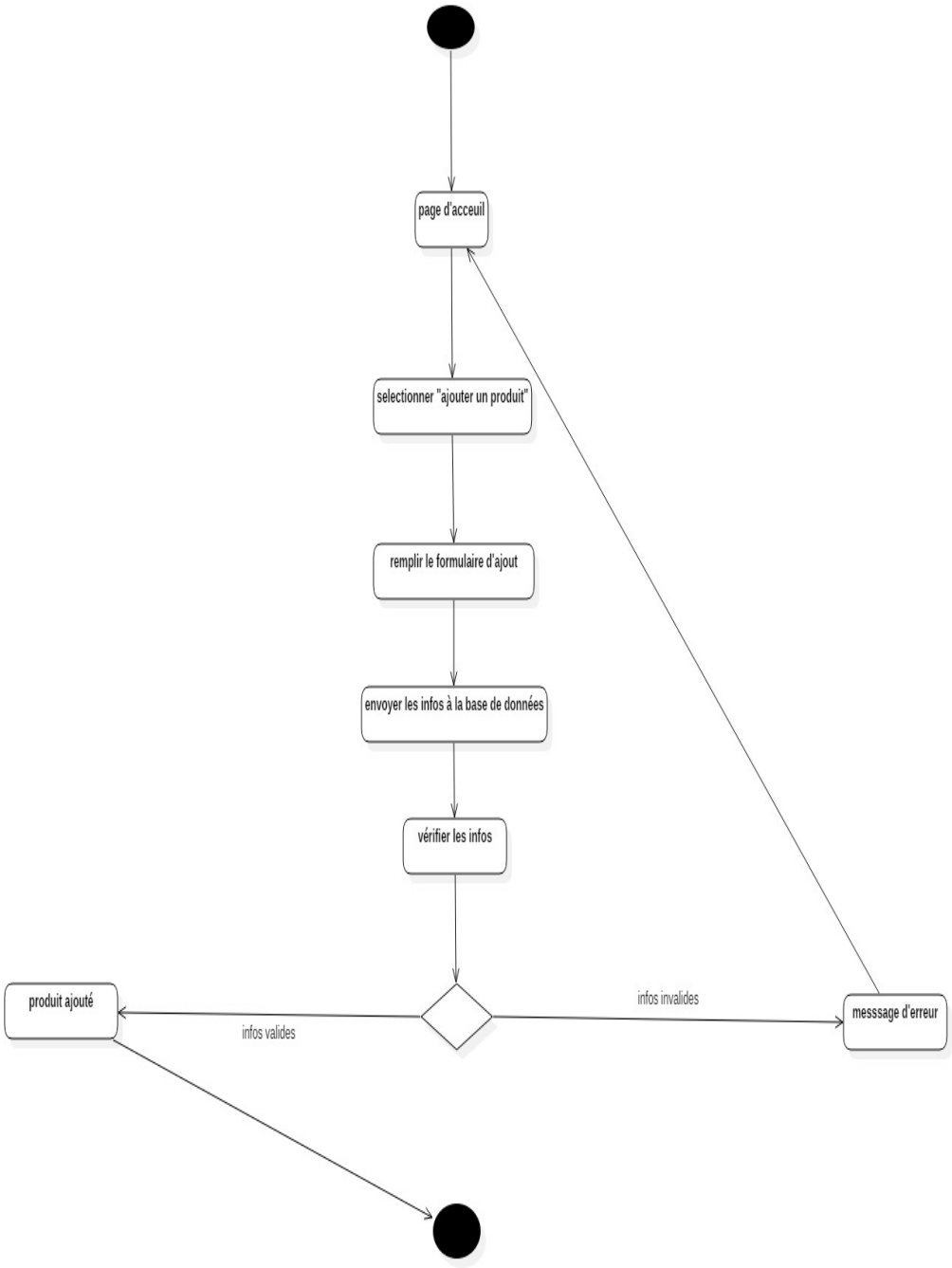


Figure 2.3.9: *Diagramme d'activité « ajouter un produit vendeur Permanent »*

- Poster une annonce :

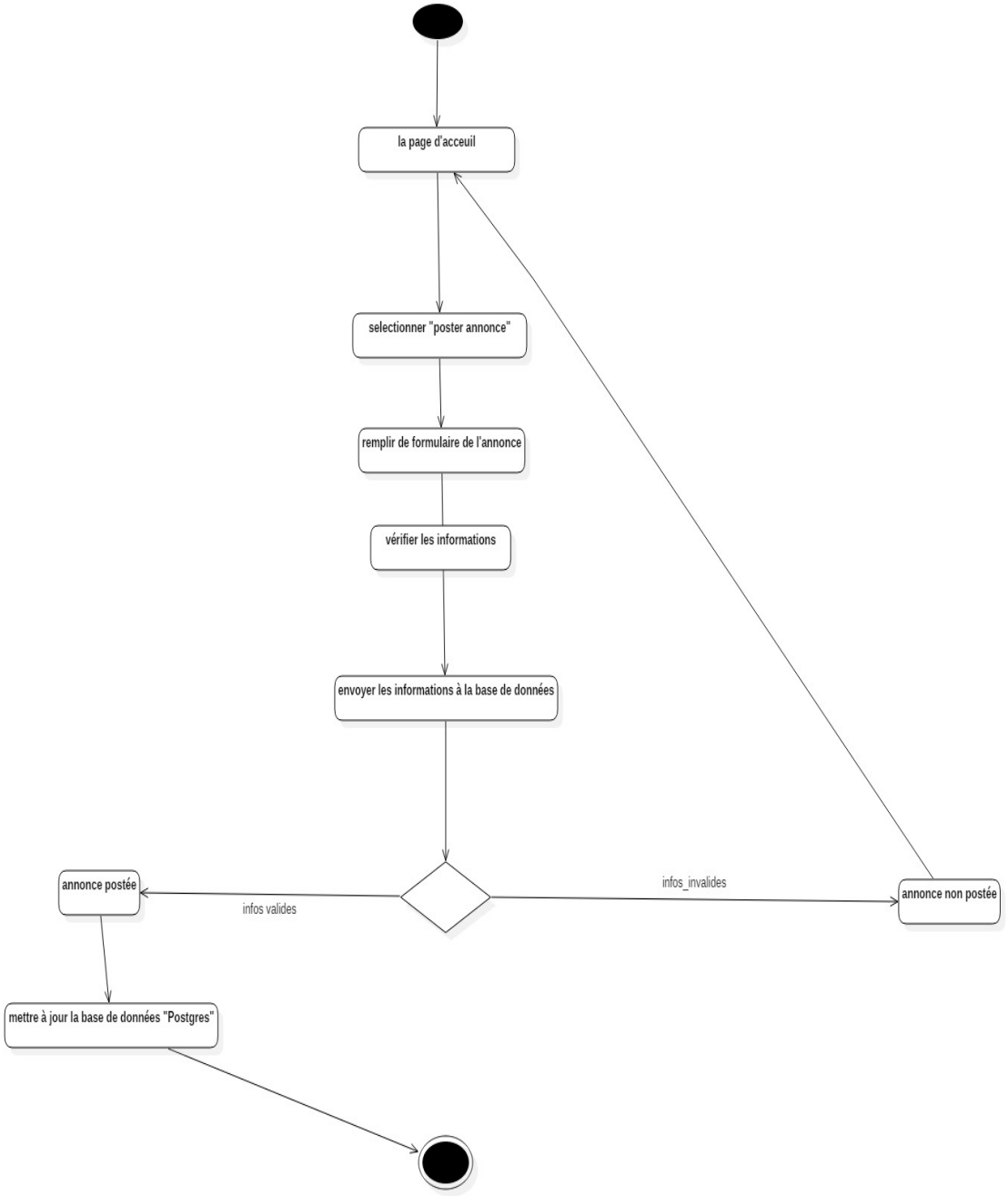


Figure 2.3.10: *Diagramme d'activité « poster une annonce vendeur Permanent»*

- Modifier un produit :

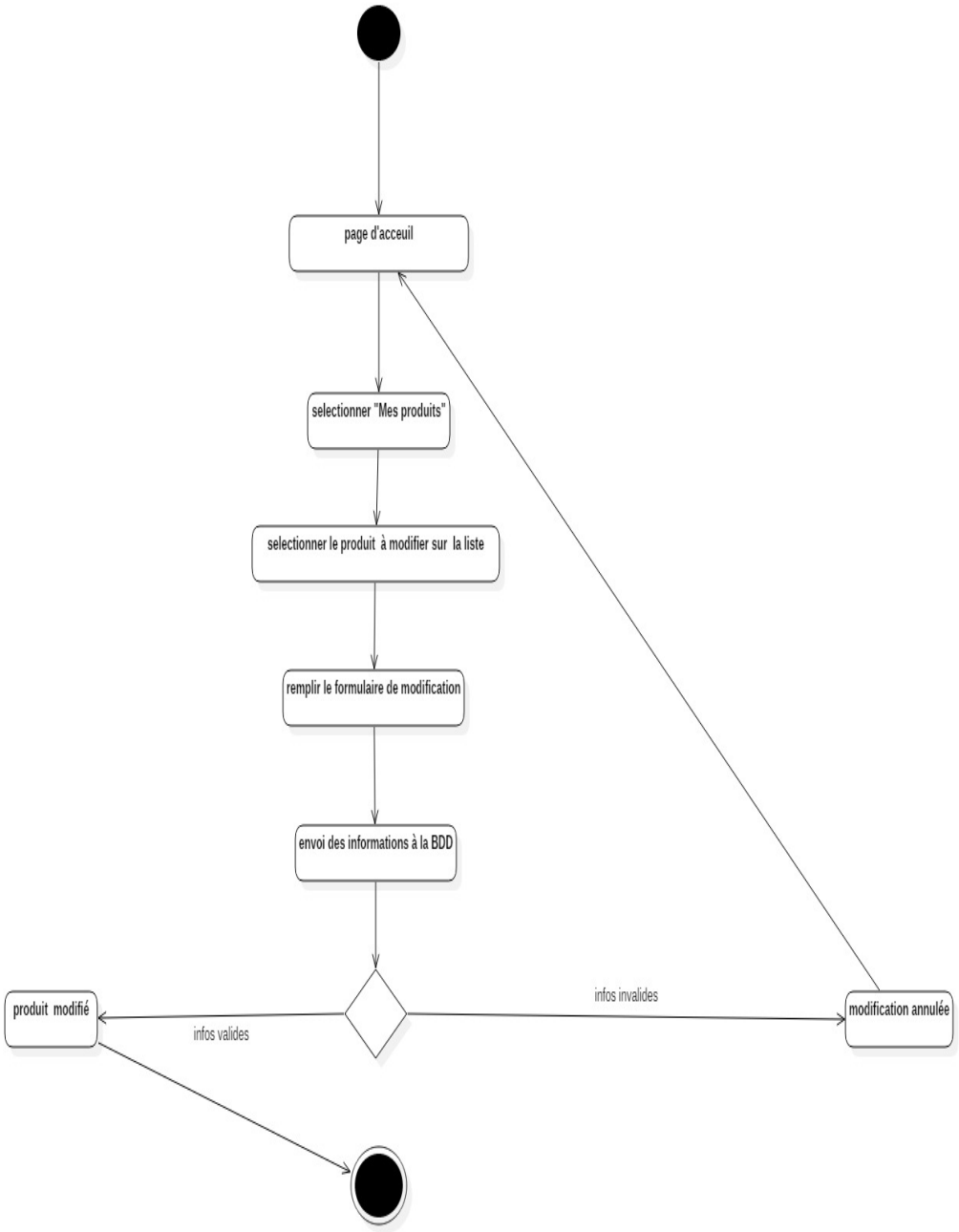


Figure 2.3.11: *Diagramme d'activité « modifier un produit vendeur Permanent»*

- Supprimer une annonce :

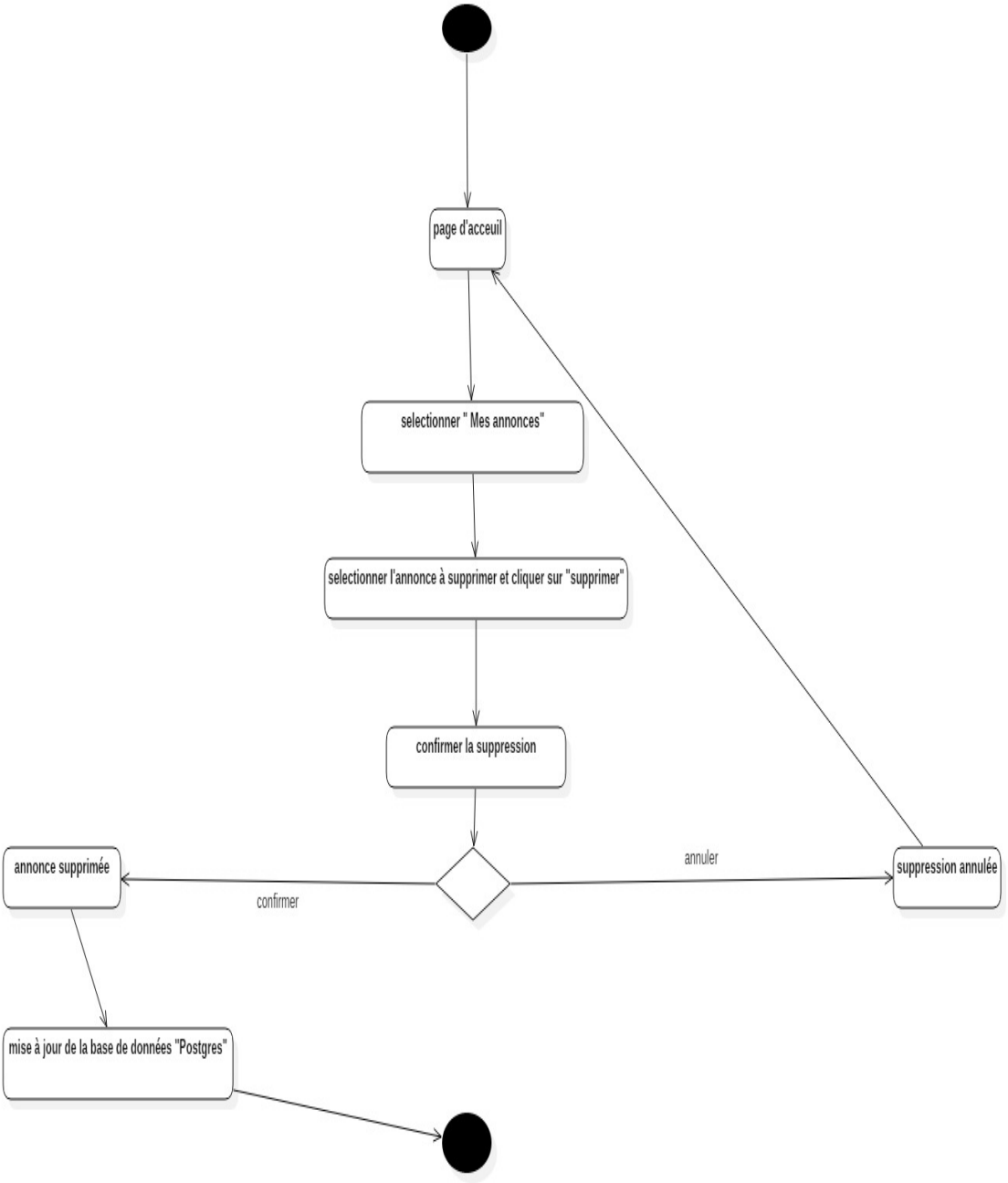


Figure 2.3.12: *Diagramme d'activité « supprimer une annonce vendeur Permanent»*

Vendeur occasionnel / Service / Entreprise de service:

- Inscription du vendeur :

Figure 2.3.13: Diagramme d'activité *«Inscription du vendeur Occasionnel»*

- Connexion du vendeur :

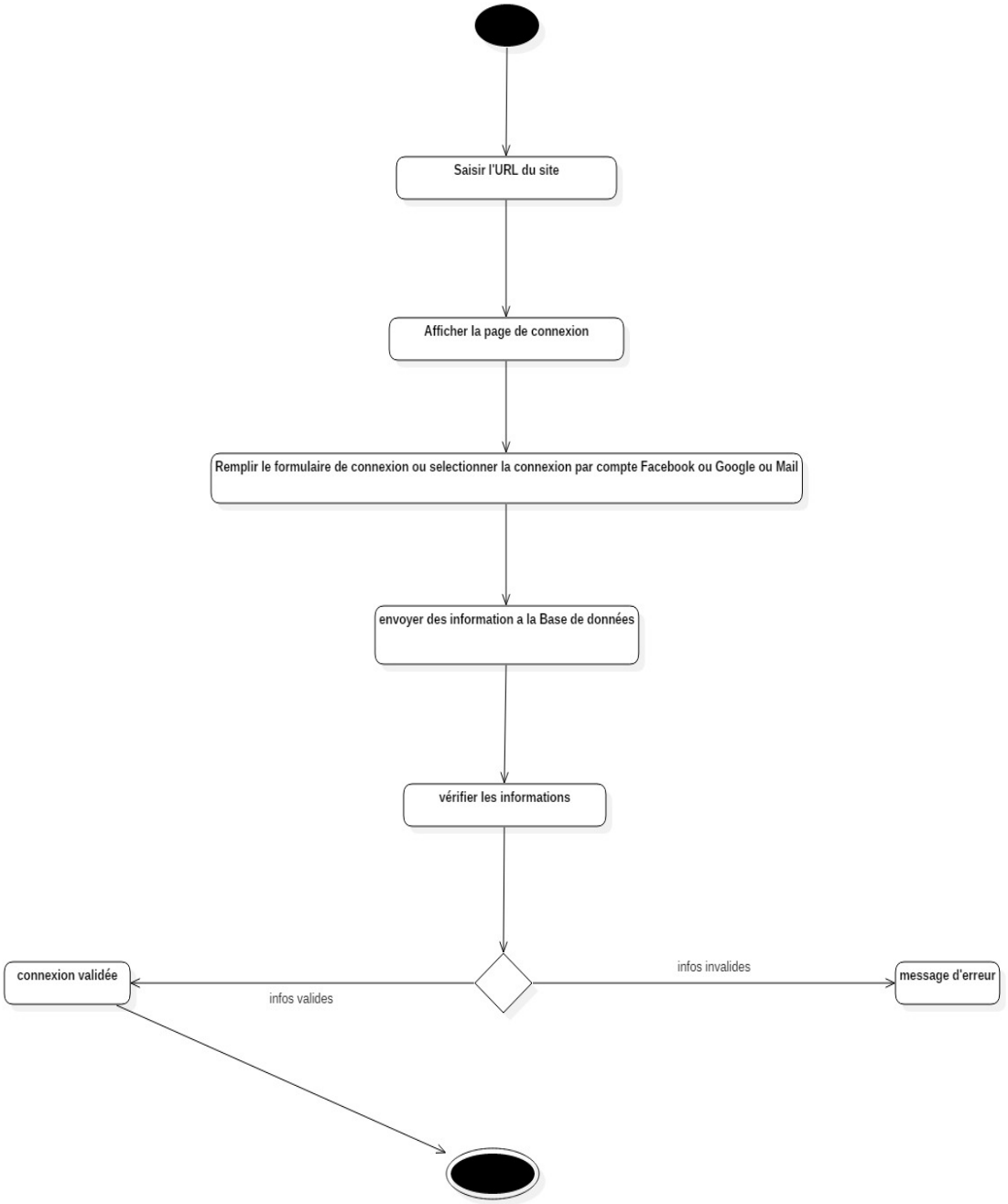


Figure 2.3.14: diagramme d'activité *«Connexion du vendeur Occasionnel»*

- Poster une annonce:

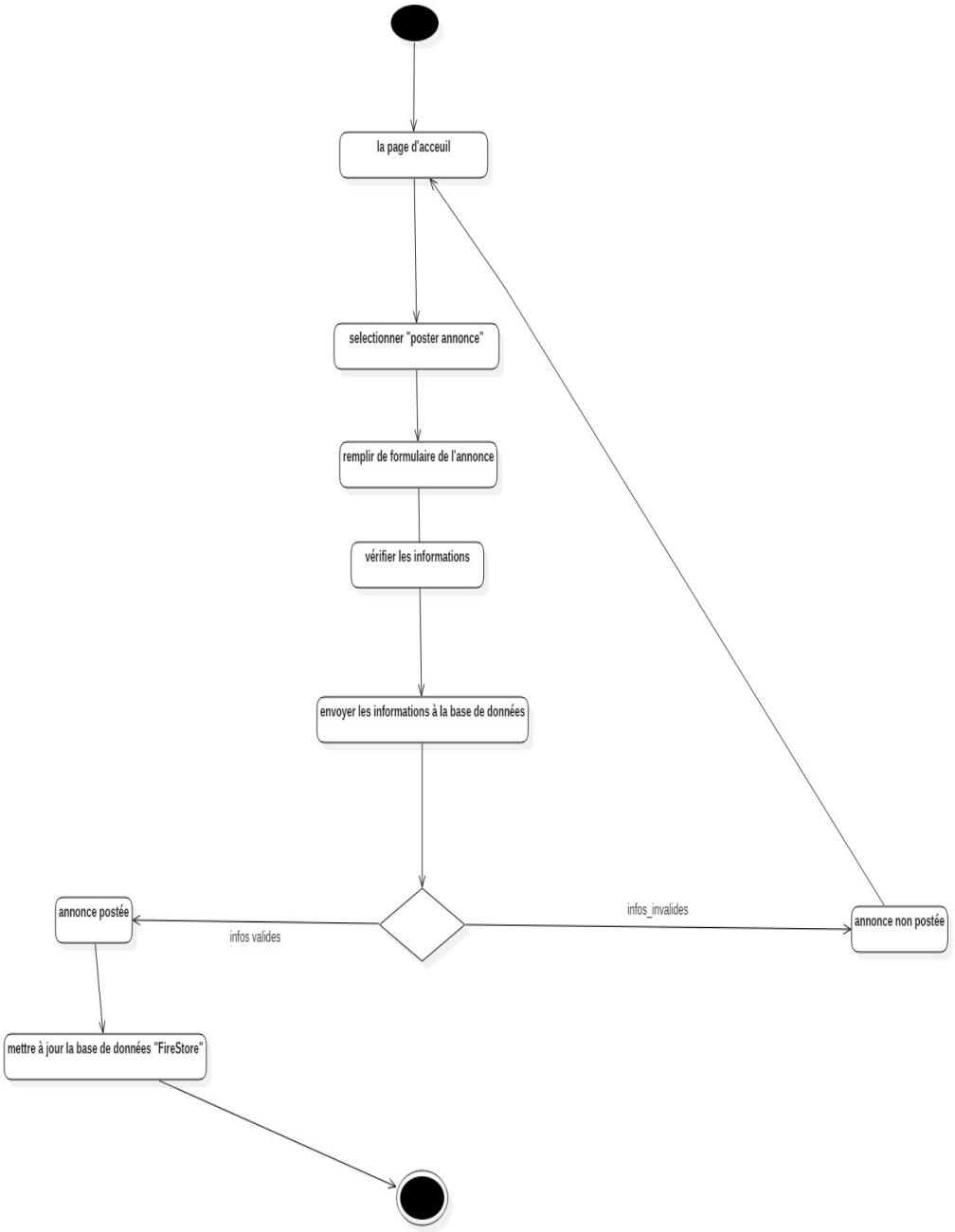


Figure 2.3.15: Diagramme d'activité *«Poster une annonce vendeur Occasionnel»*

- Modifier une annonce:

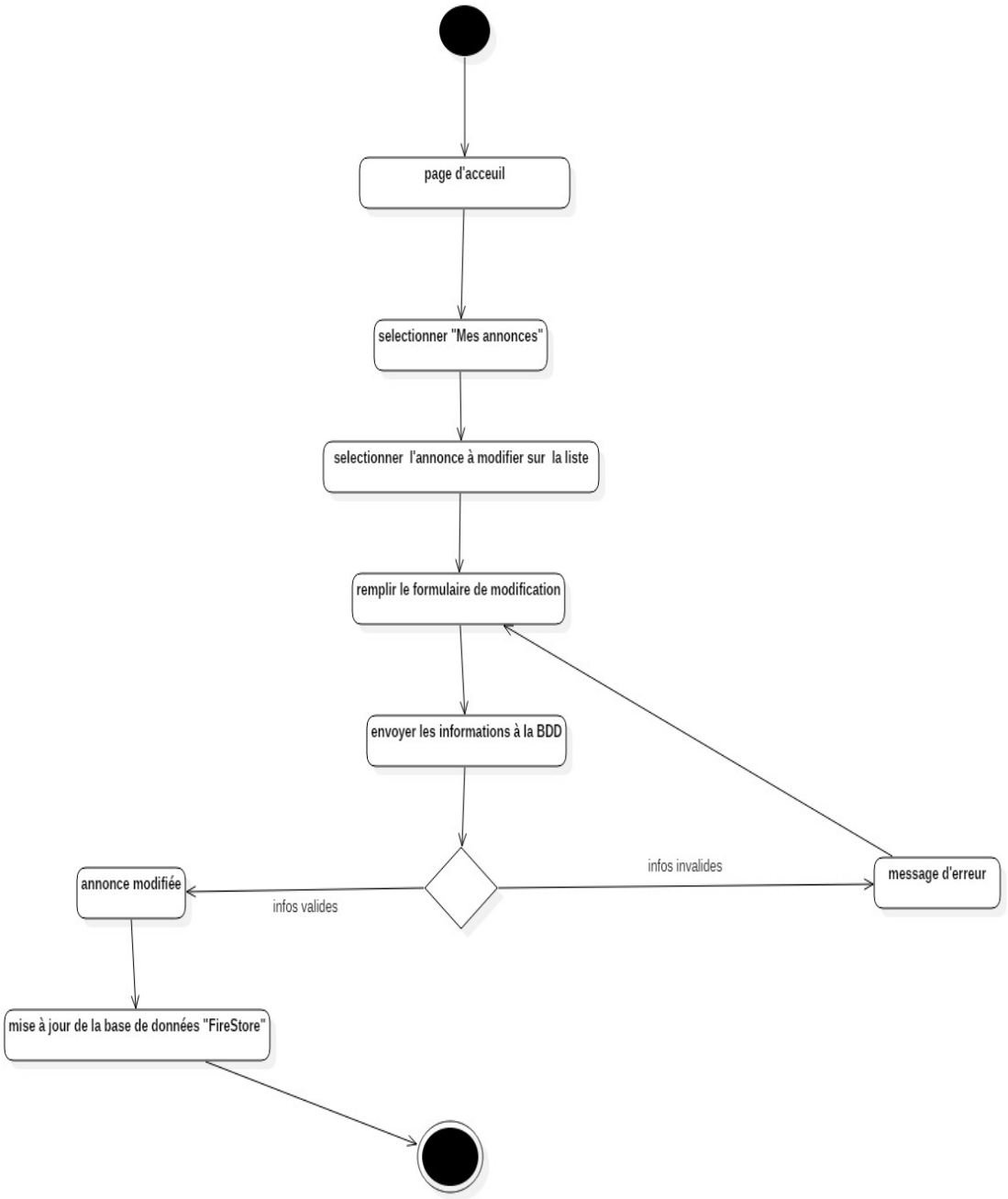


Figure 2.3.16: diagramme d'activité *«Modifier une annonce vendeur Occasionnel »*

Client:

- Inscription du client:

Figure 2.3.17: diagramme d'activité *«Inscription du client»*

- Connexion du client:

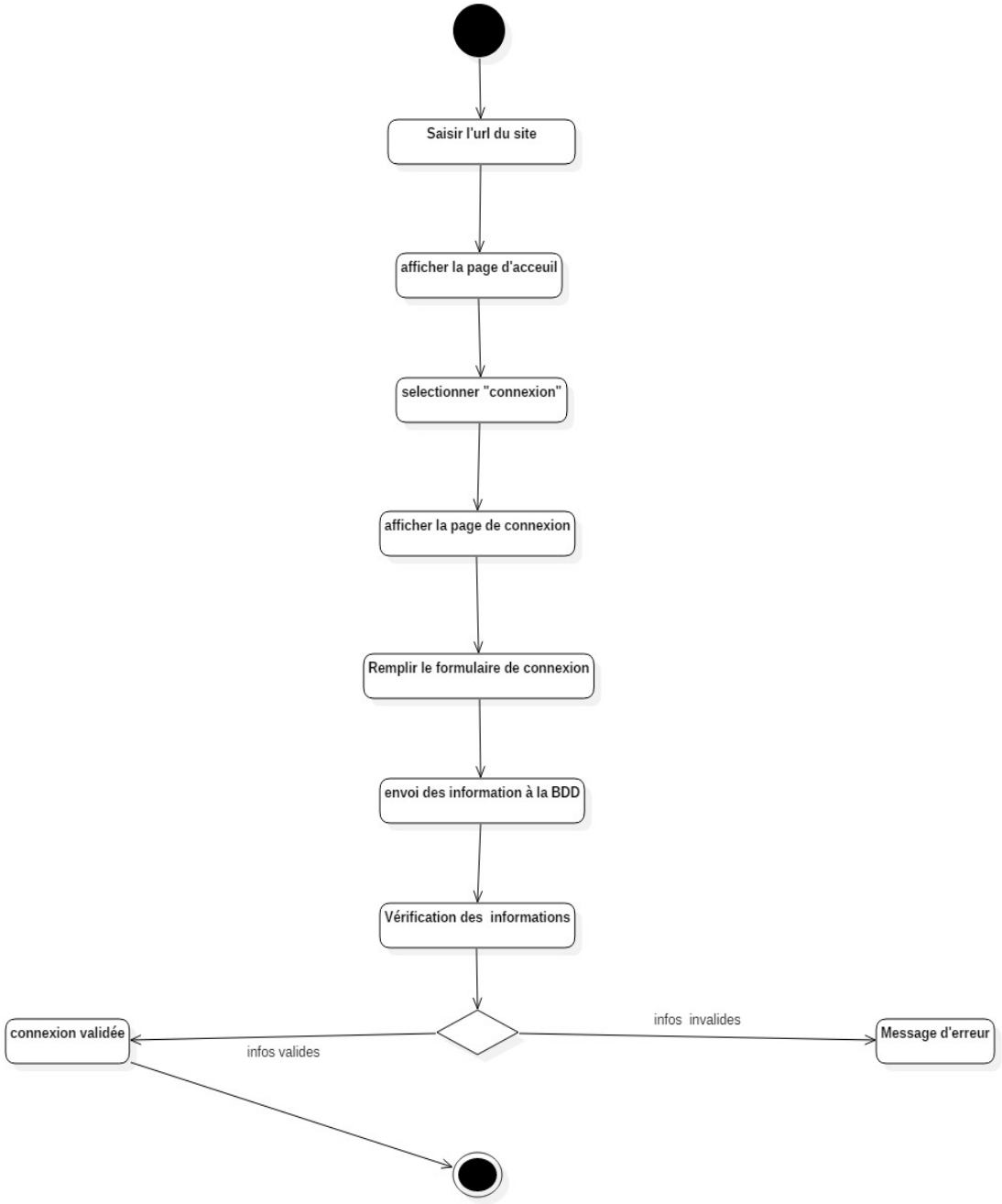


Figure 2.3.18: diagramme d'activité *«Connexion du client»*

- Rechercher un produit:

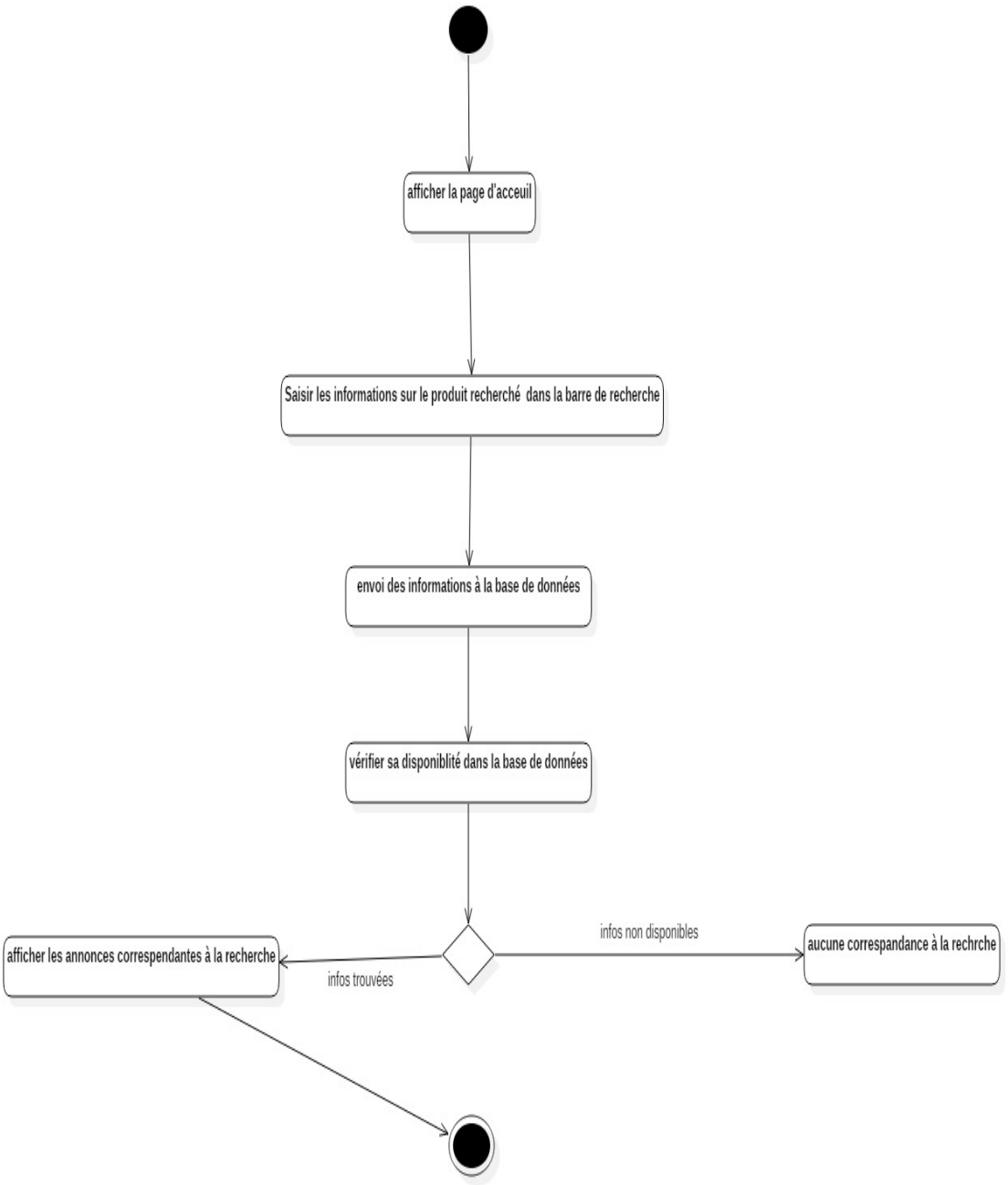


Figure 2.3.19: diagramme d'activité *«rechercher un produit»*

- Contacter le vendeur:

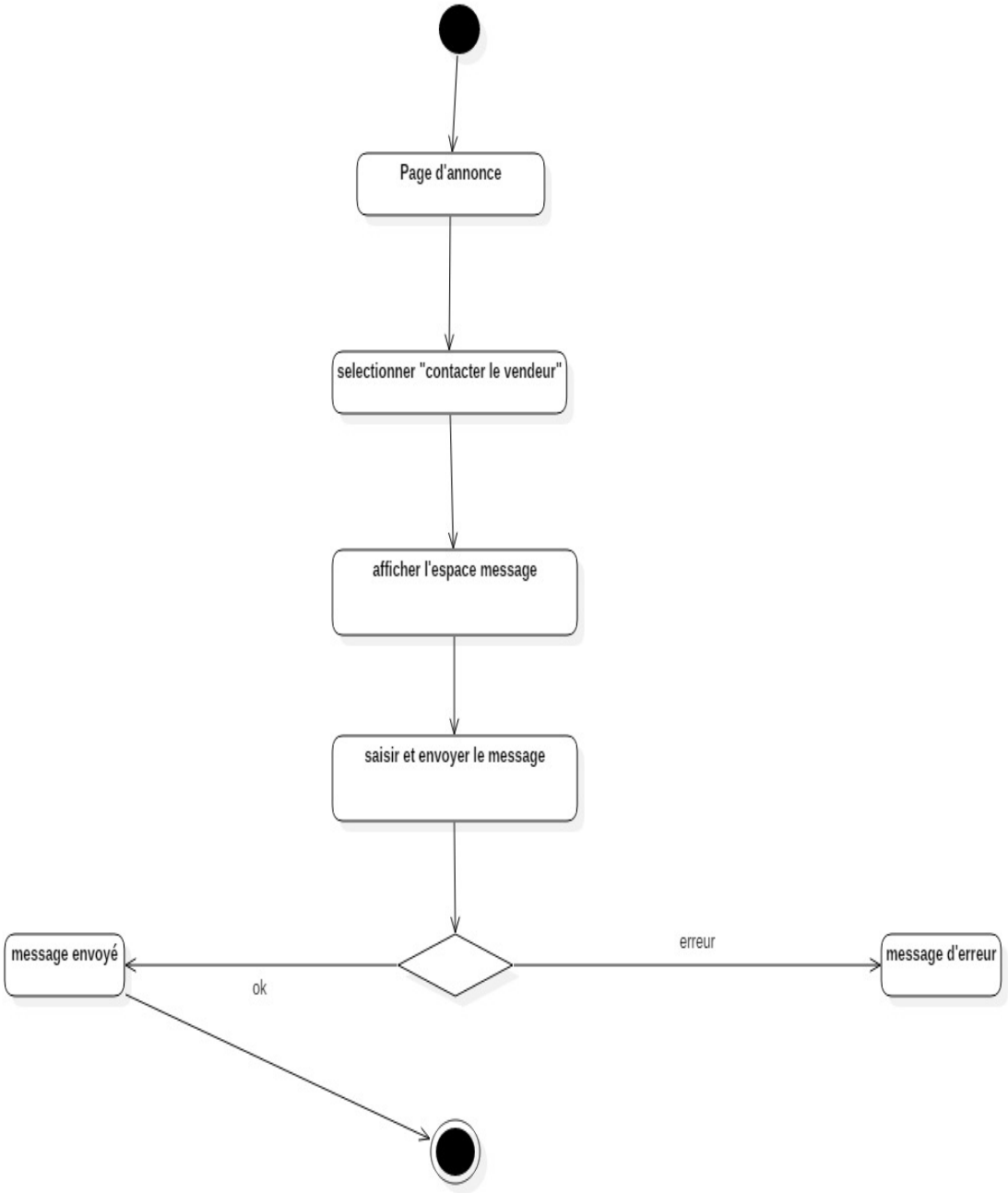


Figure 2.3.20: diagramme d'activité *«contacter le vendeur»*

2.3.1.5 Les diagrammes de séquence:

Vendeur Permanent :

- Inscription vendeur:

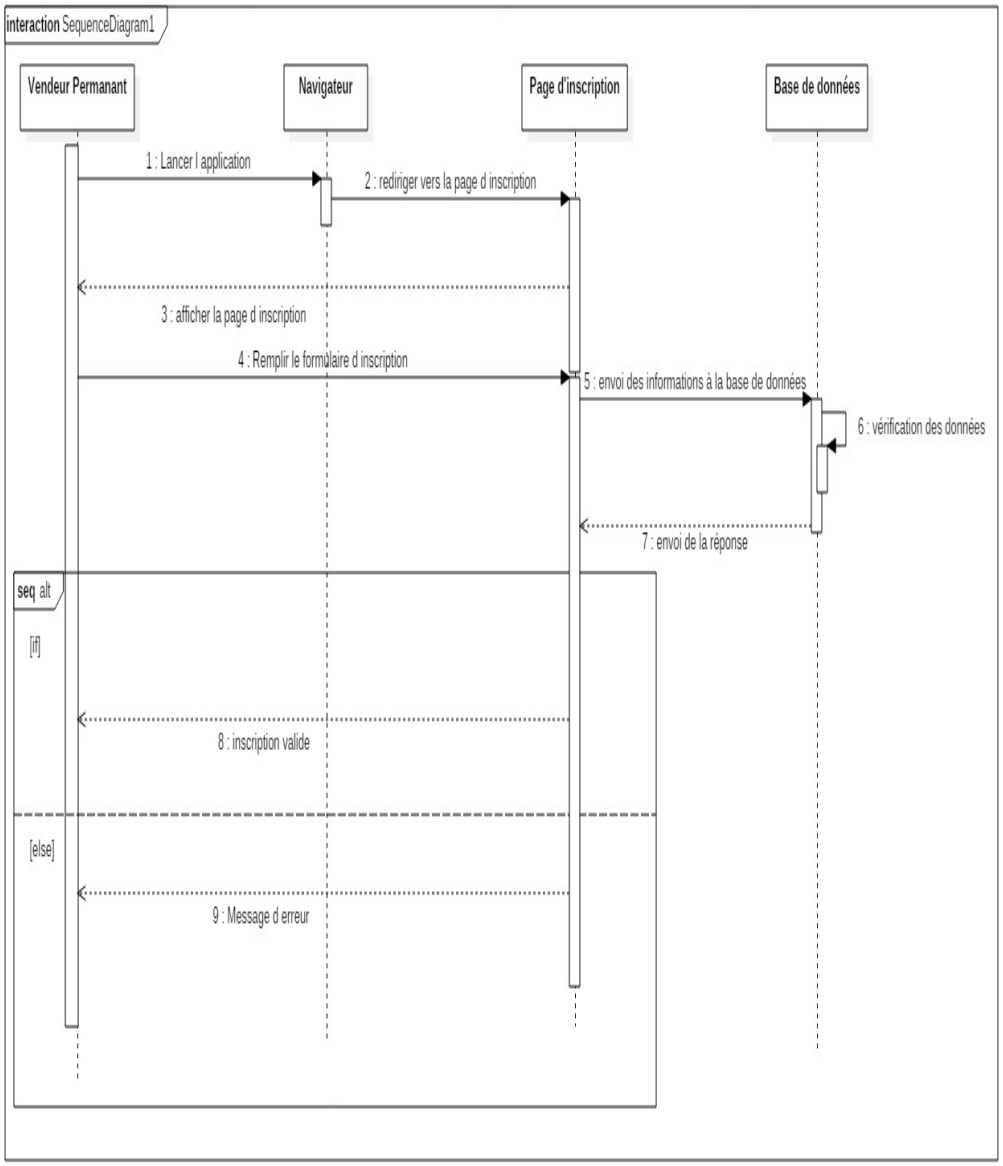


Figure 2.3.21: *Diagramme de séquence « inscription du vendeur »*

- Authentification vendeur :

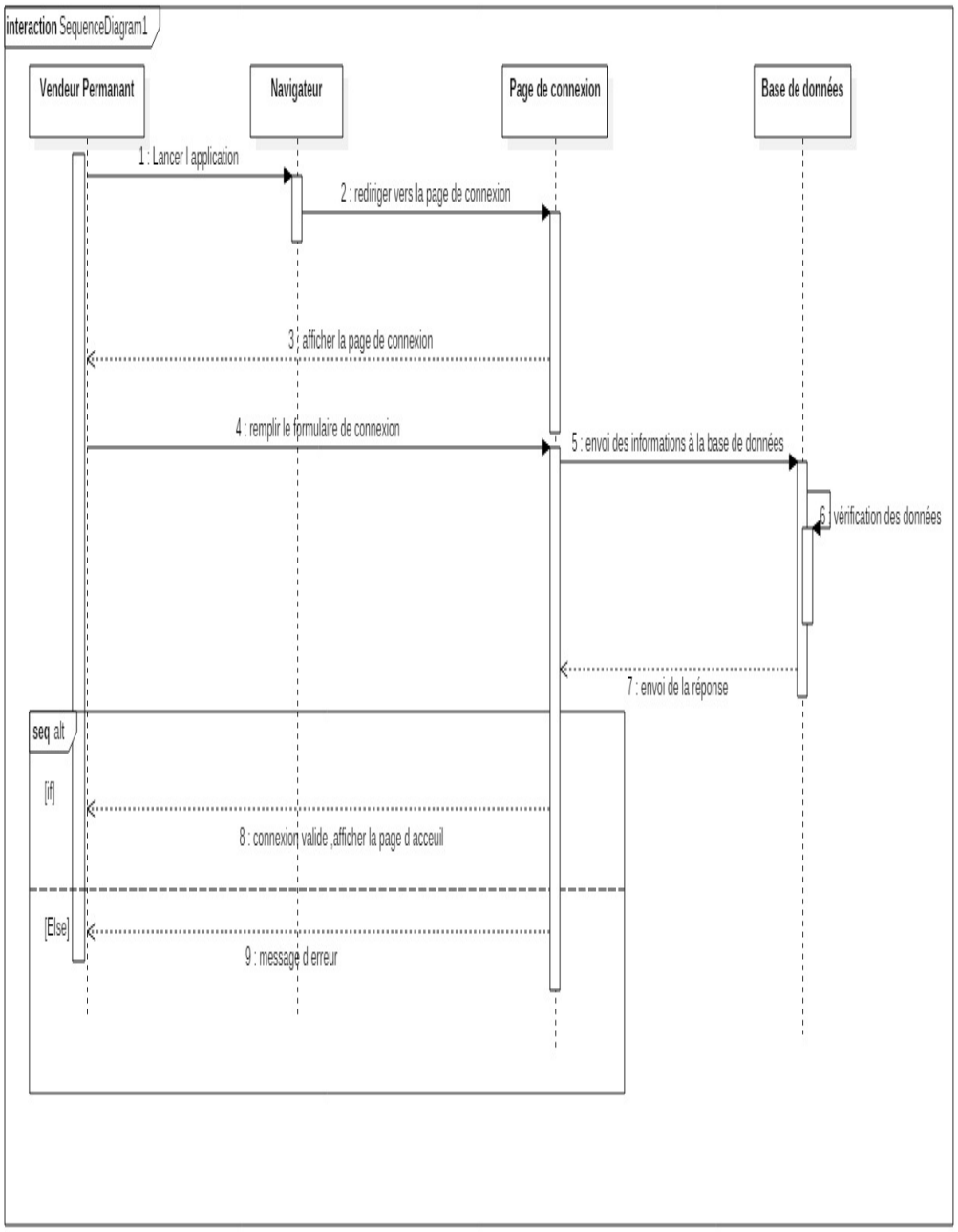


Figure 2.3.22: *Diagramme de séquence « authentification du vendeur »*

- Ajouter un produit :

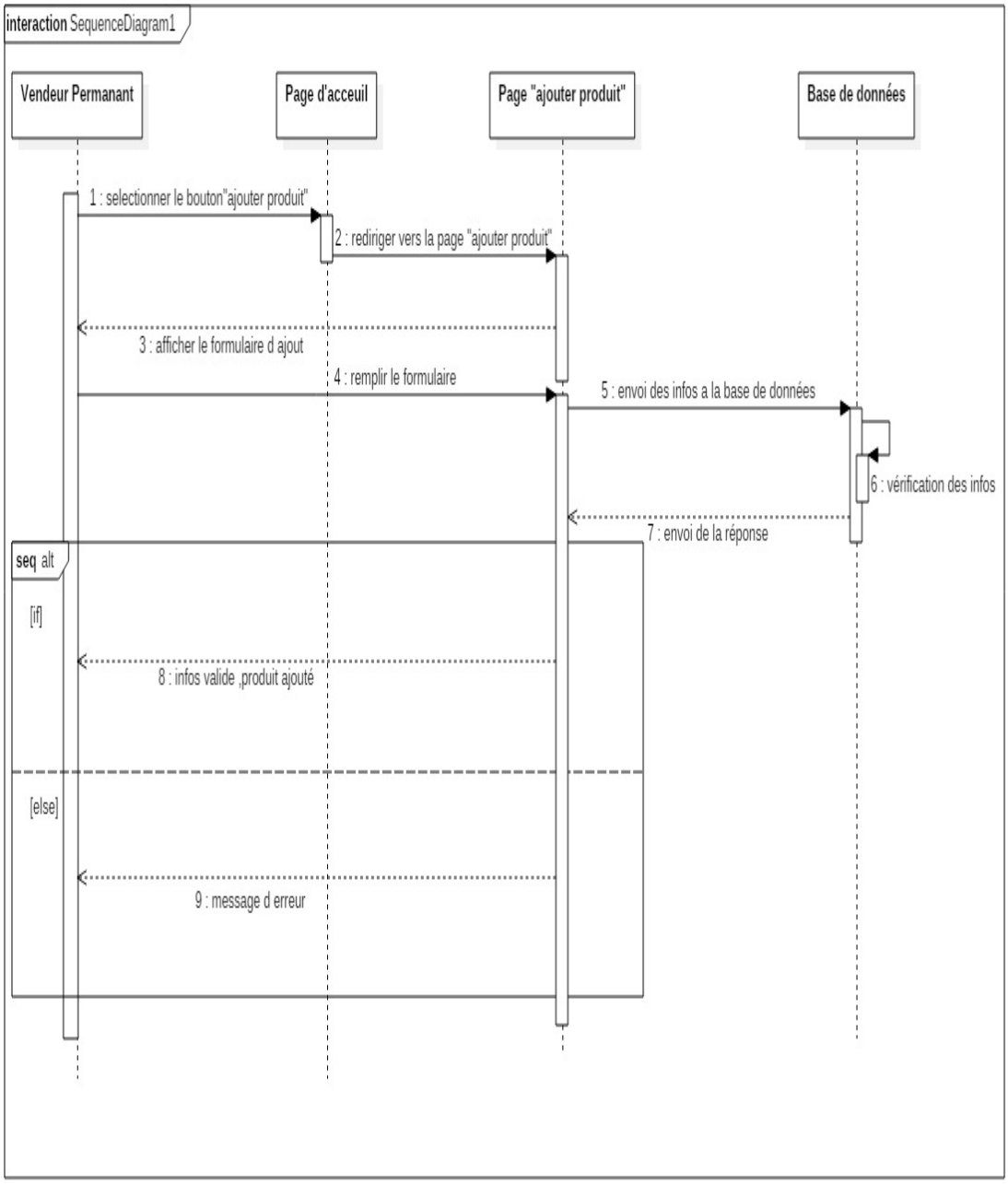


Figure 2.3.23: *Diagramme de séquence « ajouter produit »*

- Poster une annonce :

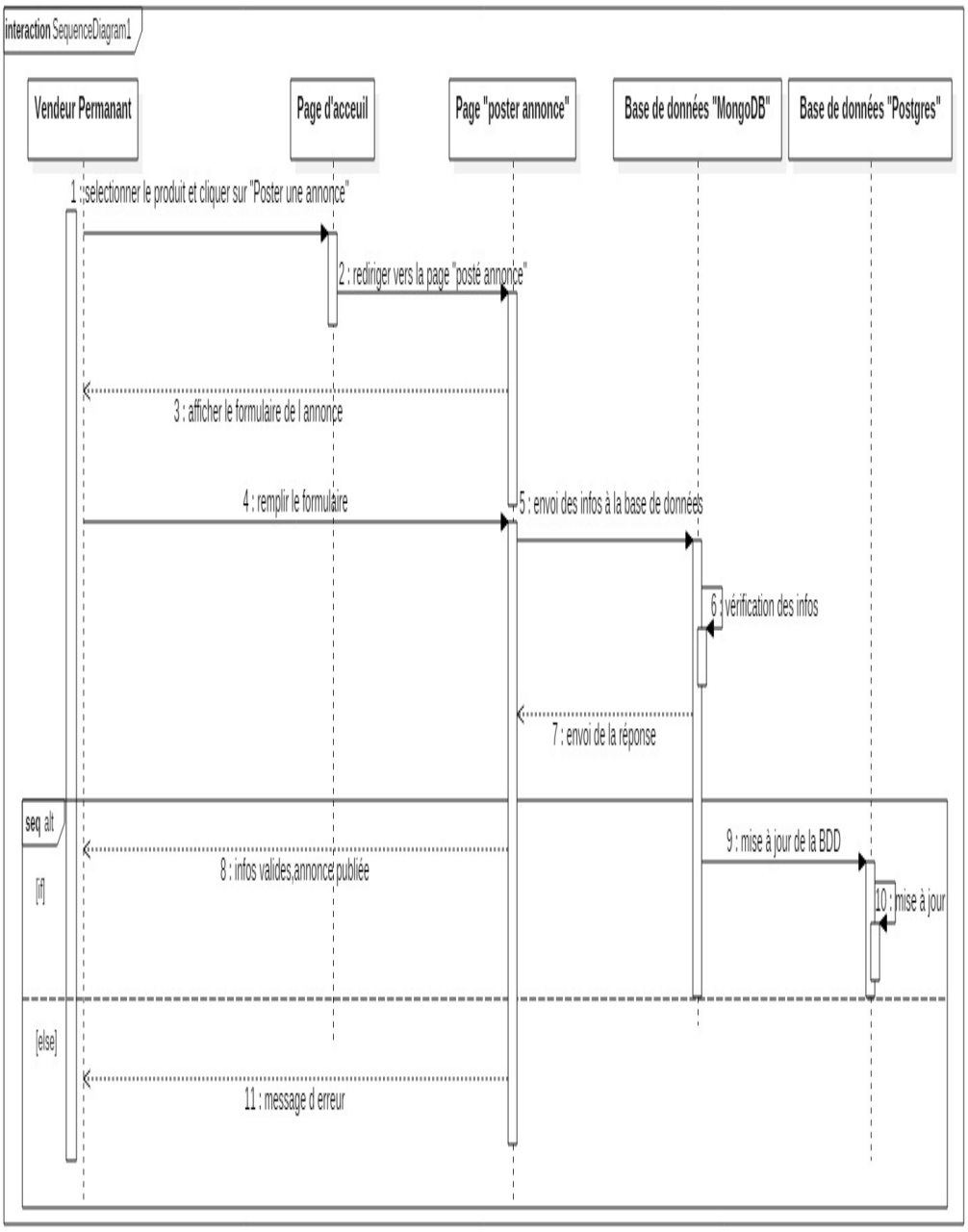


Figure 2.3.24: *Diagramme de séquence « poster une annonce »*

- Supprimer un produit :

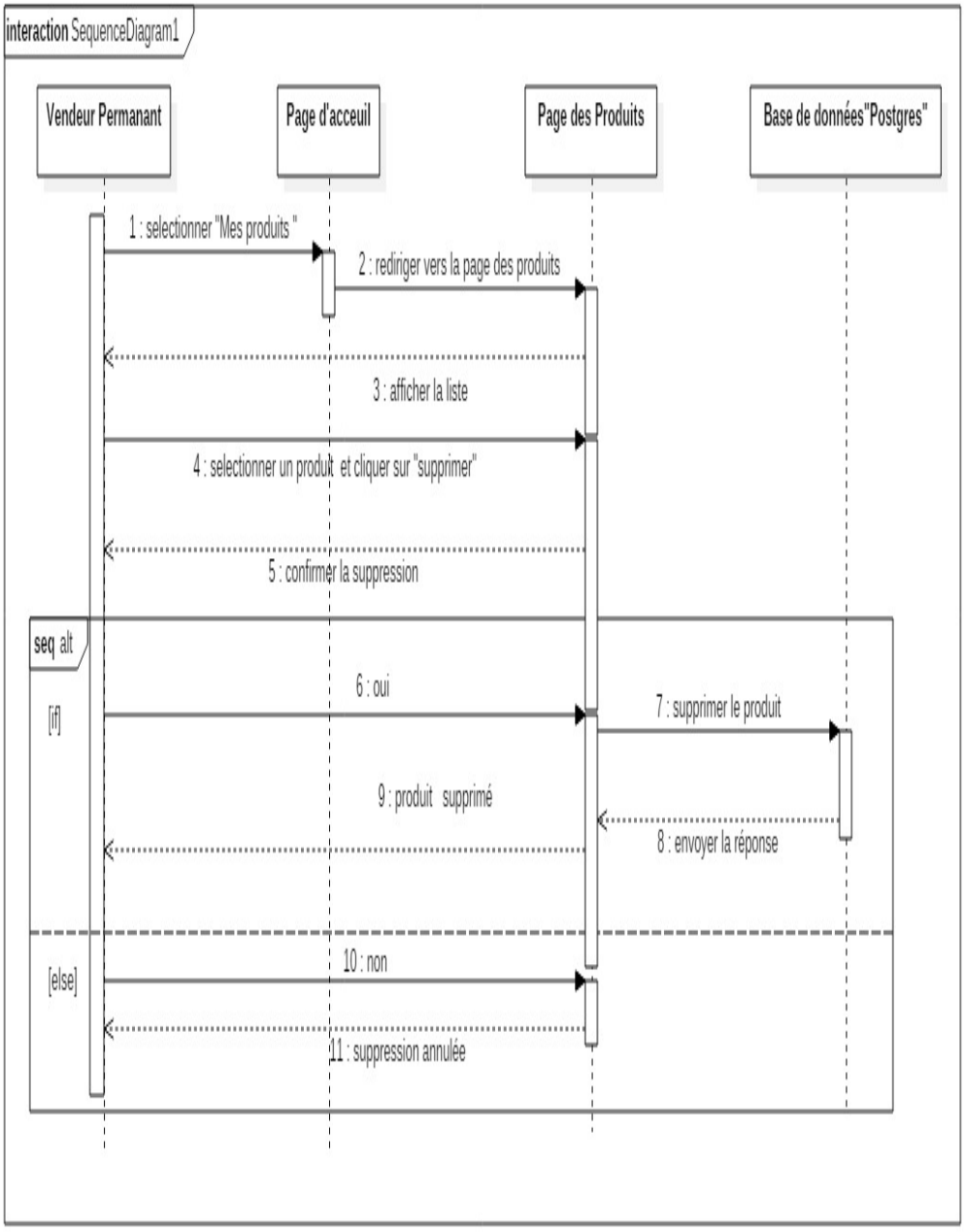


Figure 2.3.25: *Diagramme de séquence « supprimer un produit vendeur permanent »*

Vendeur occasionnel / Service / Entreprise de service :

- Inscription du vendeur :

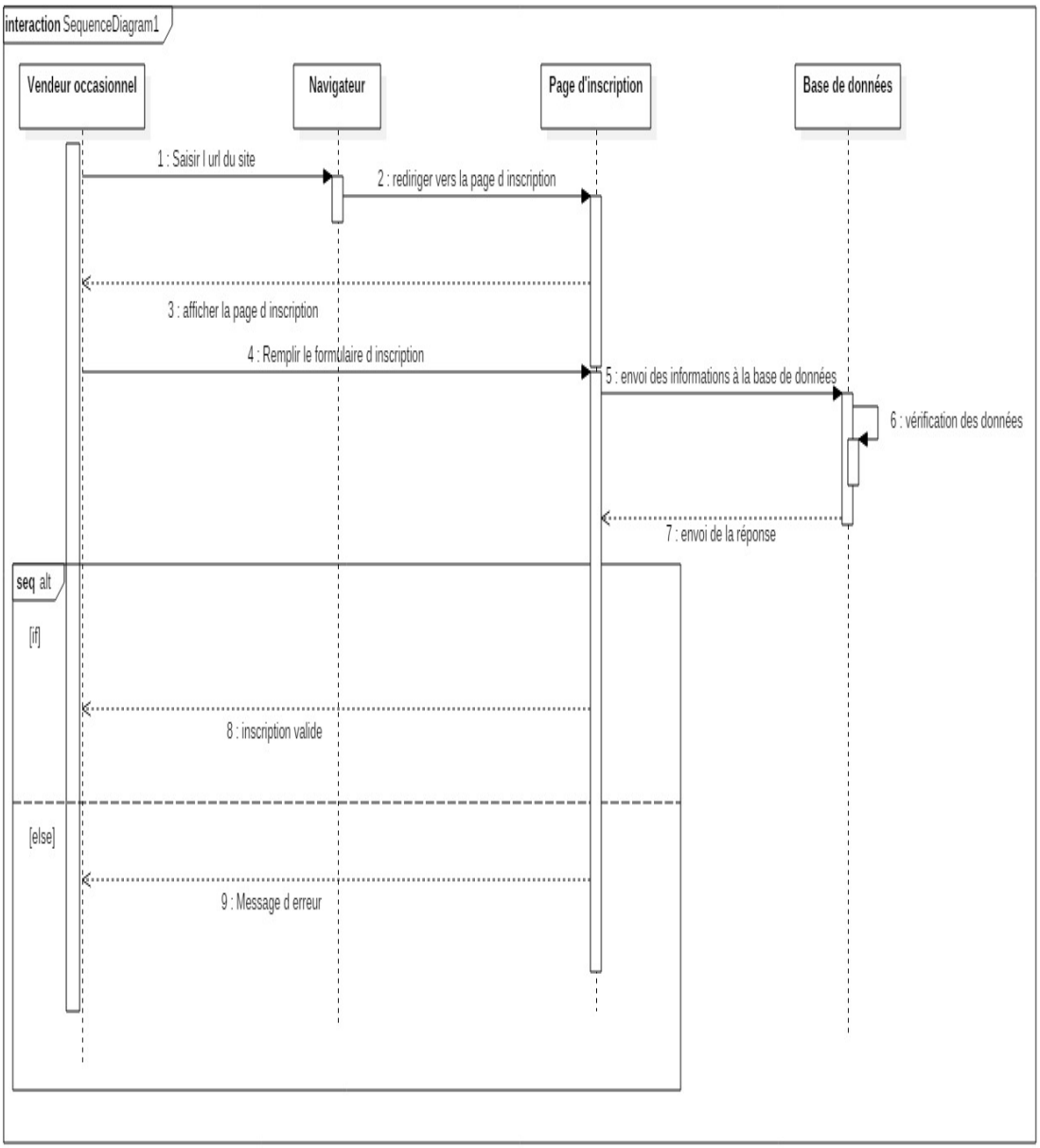


Figure 2.3.26: *diagramme de séquence «Inscription du vendeur »*

- Connexion du vendeur :

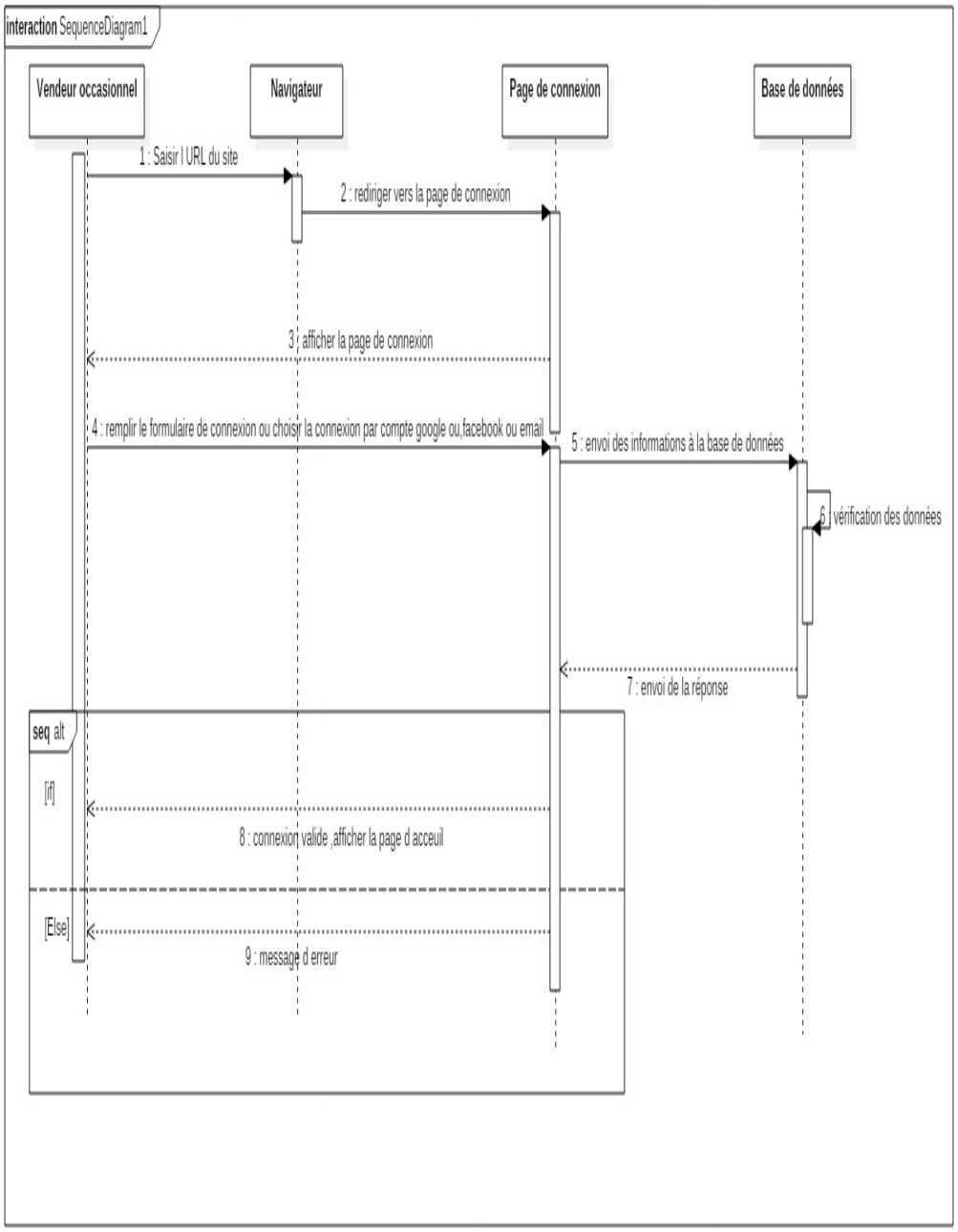


Figure 2.3.27: *diagramme de séquence «connexion du vendeur »*

- Poster une annonce:

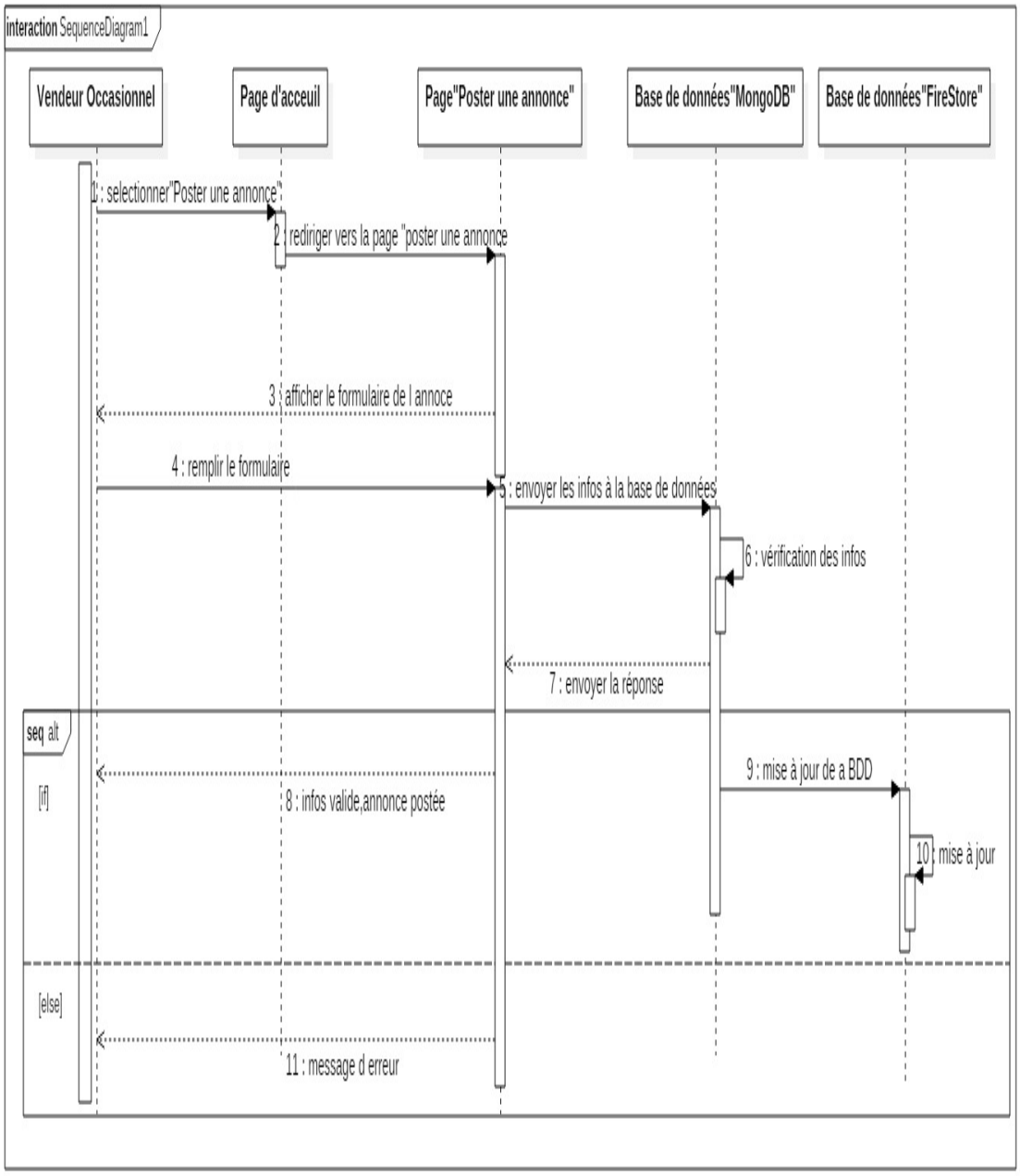


Figure 2.3.28: *Diagramme de séquence «poster une annonce »*

- Modifier une annonce:

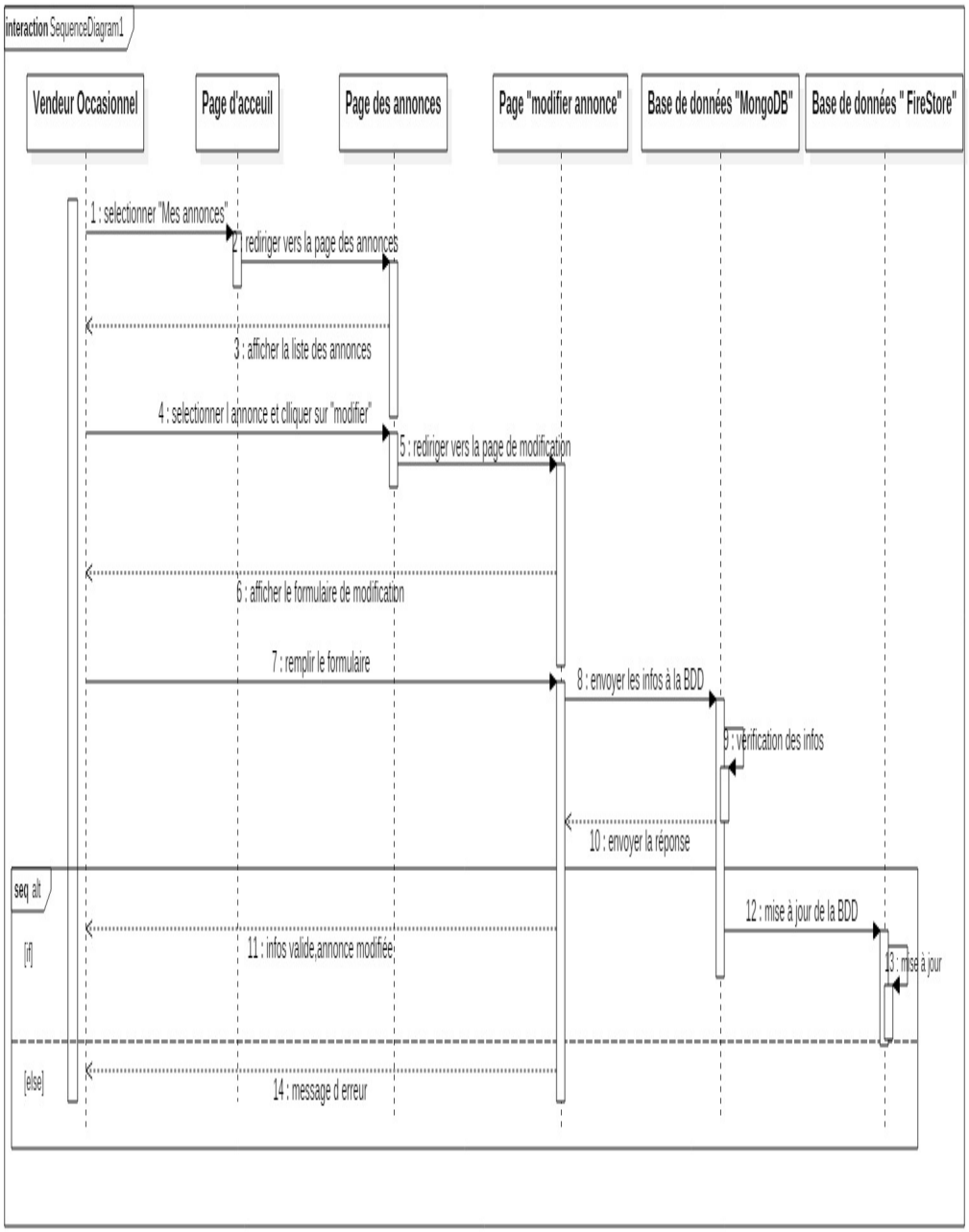


Figure 2.3.29: *Diagramme de séquence «Modifier une annonce »*

- Supprimer une annonce:

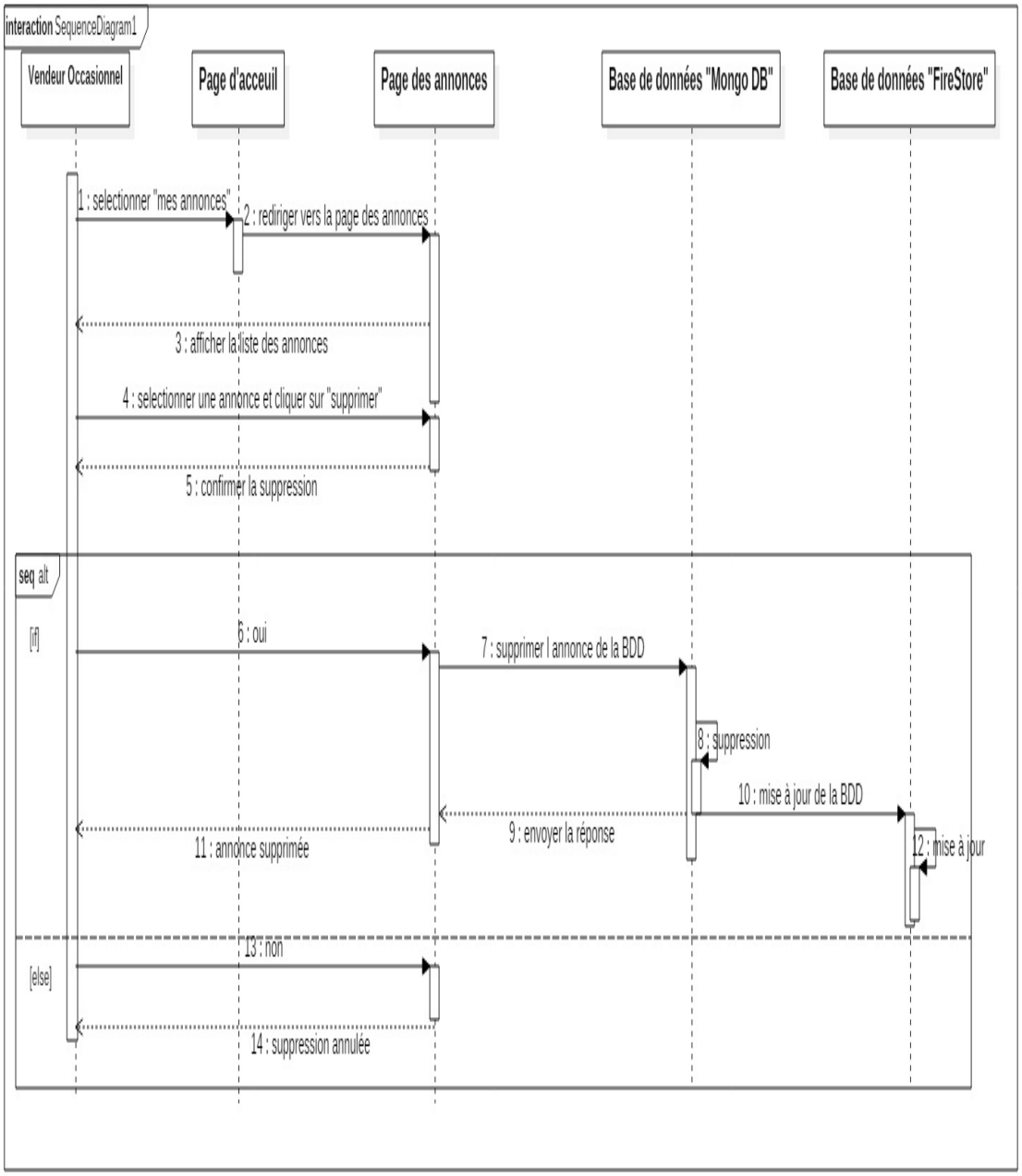


Figure 2.3.30: *Diagramme de séquence «Supprimer une annonce »*

Client:

- Inscription du client:

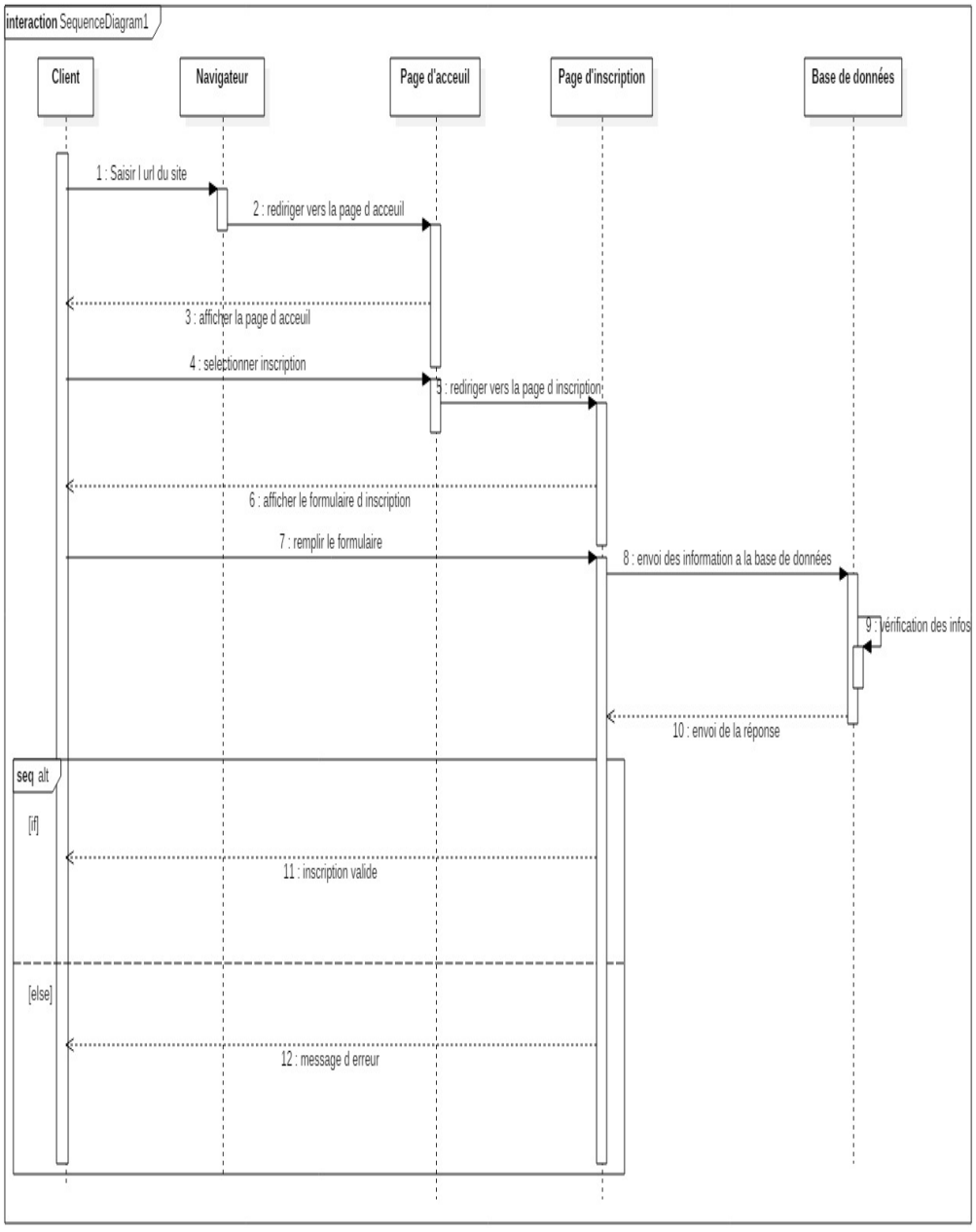


Figure 2.3.31: *diagramme de séquence «inscription du client »*

- Connexion du client :

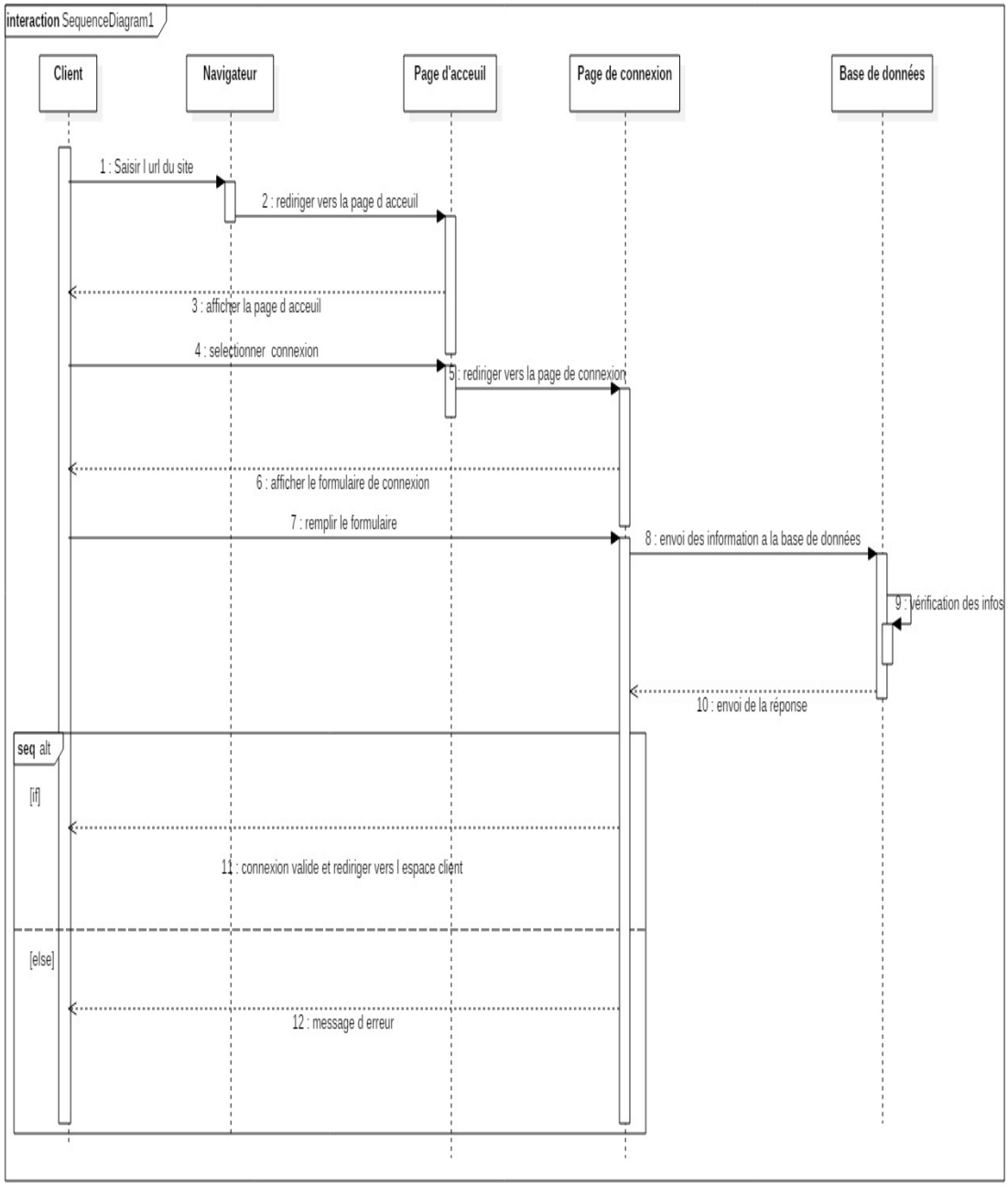


Figure 2.3.32: *diagramme de séquence «connexion du client »*

- Rechercher un produit:

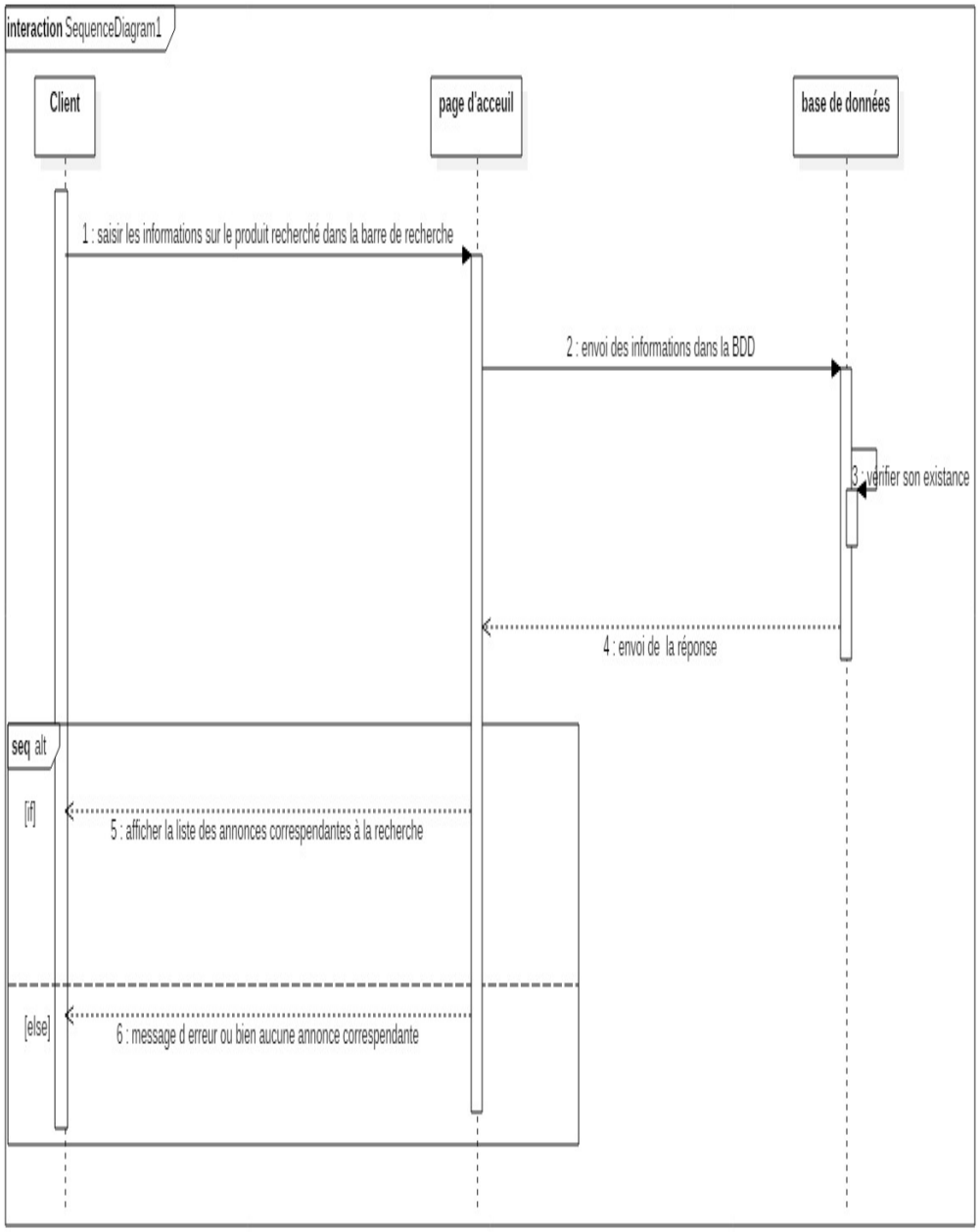


Figure 2.3.33: diagramme de séquence *«Rechercher un produit»*

- Contacter vendeur:

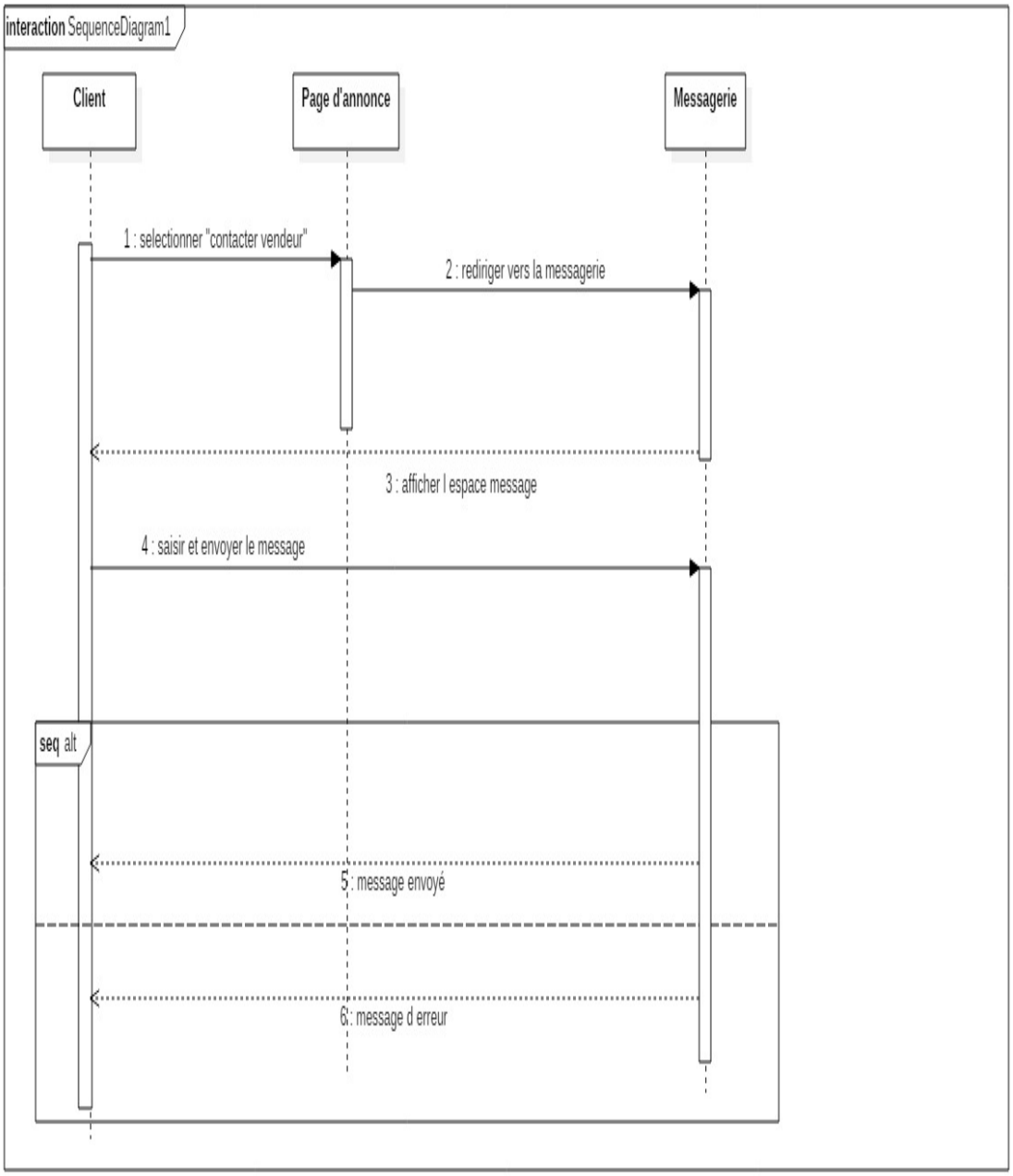


Figure 2.3.34: diagramme de séquence *«Contacter le vendeur»*

2.3.1.6 Diagramme de classe:

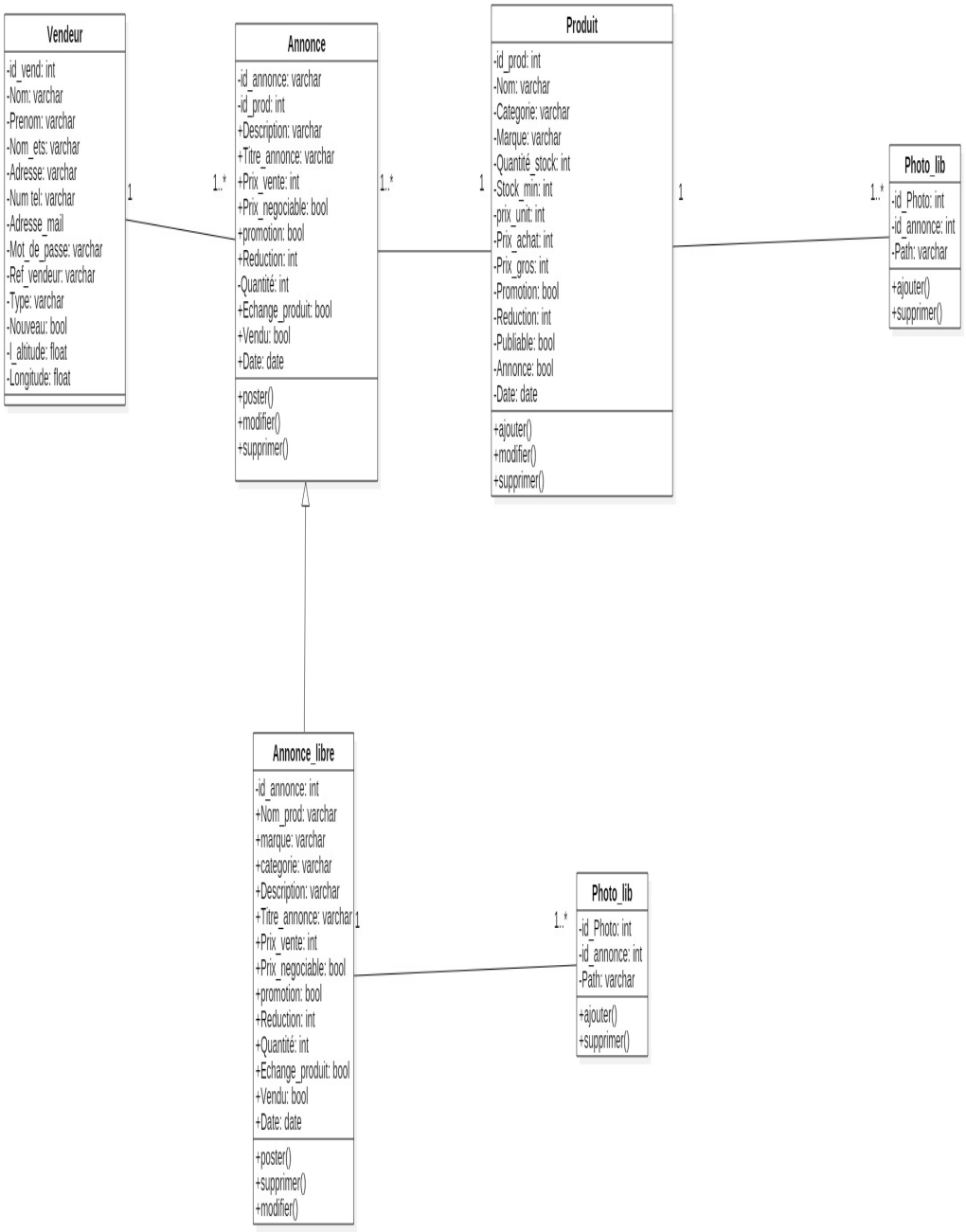


Figure 2.3.35: Diagramme de classe du vendeur permanent

2.3.2 Modélisation de la base données Firebase:

- Les collections:

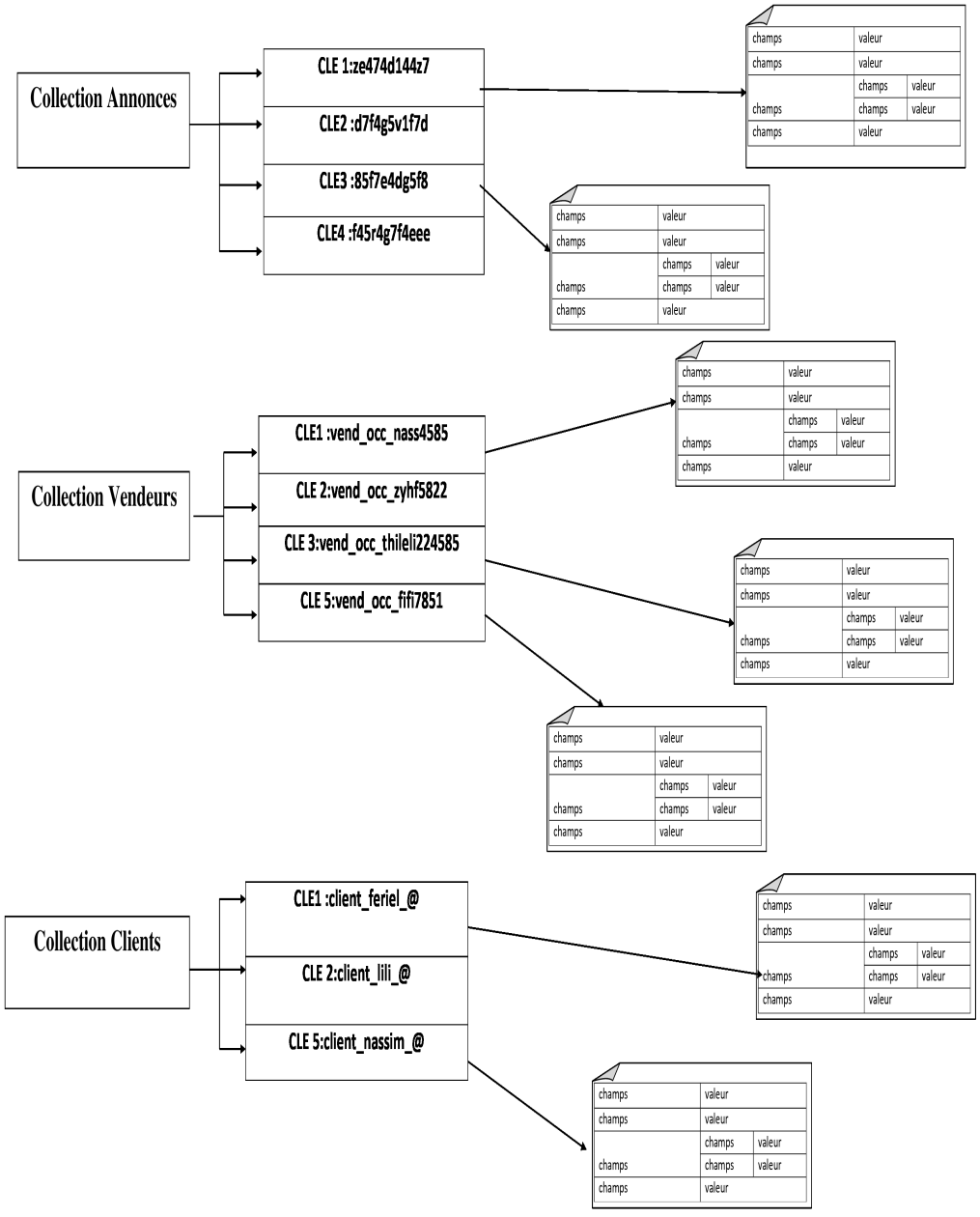
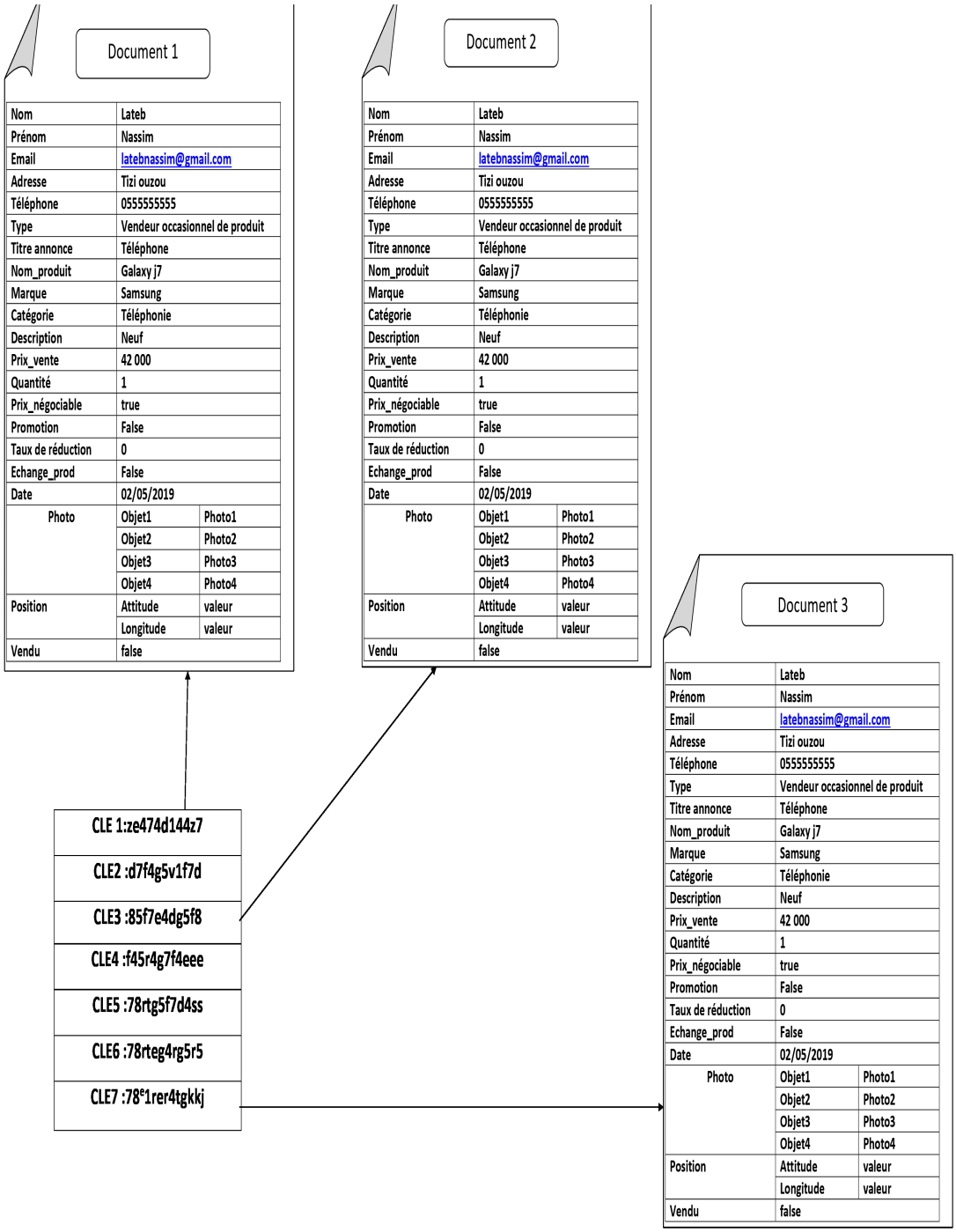


Figure 2.3.36: Modélisation de la base données Firebase

1. La collection “Annonce”:



Nom	Lateb	
Prénom	Nassim	
Email	latebnassim@gmail.com	
Adresse	Tizi ouzou	
Téléphone	0555555555	
Type	Vendeur occasionnel de produit	
Titre annonce	Téléphone	
Nom_produit	Galaxy j7	
Marque	Samsung	
Catégorie	Téléphonie	
Description	Neuf	
Prix_vente	42 000	
Quantité	1	
Prix_négociable	true	
Promotion	False	
Taux de réduction	0	
Echange_prod	False	
Date	02/05/2019	
Photo	Objet1	Photo1
	Objet2	Photo2
	Objet3	Photo3
	Objet4	Photo4
Position	Attitude	valeur
	Longitude	valeur
Vendu	false	

Figure 2.3.37: Modélisation de la collection Annonce

2. La collection "Vendeur":

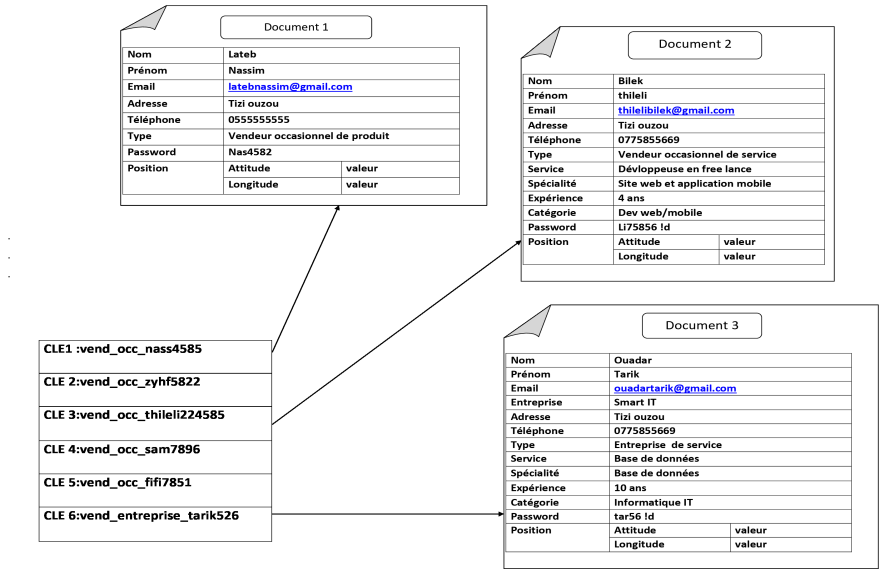


Figure 2.3.38: Modélisation de la collection Vendeur

3. La collection "Client":

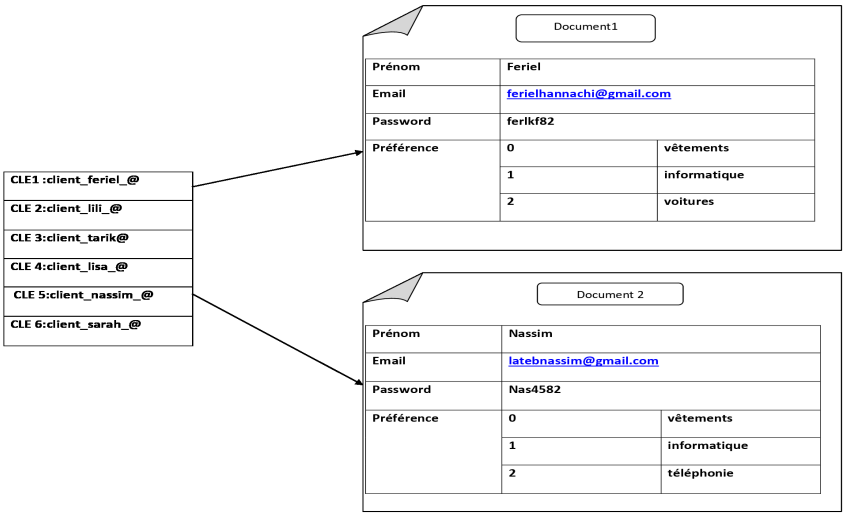


Figure 2.3.39: Modélisation de la collection Client

2.3.3 Modélisation de la base données MongoDB:

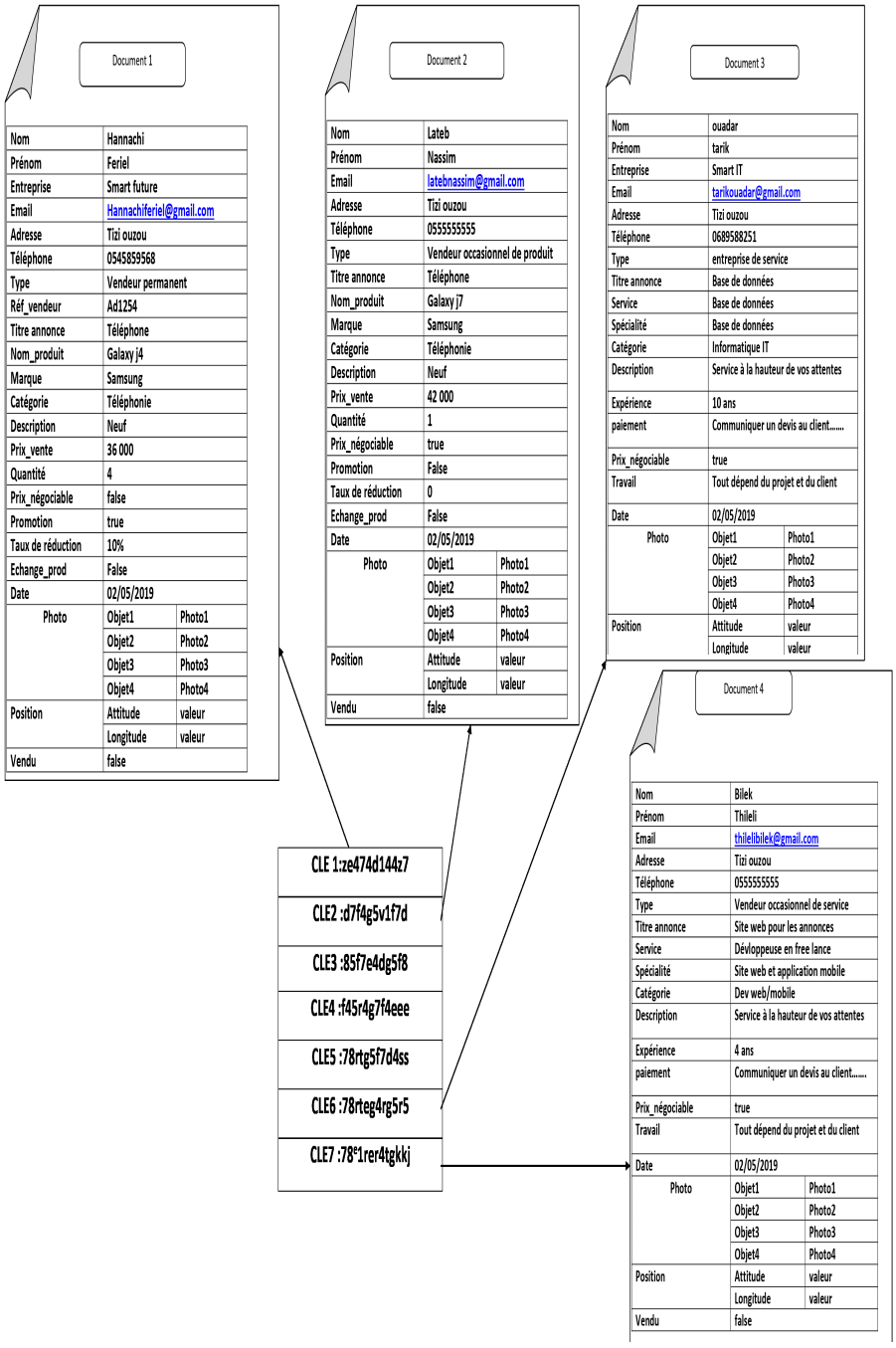


Figure 2.3.40: Modélisation de la base de données orientée document"MongoDB"

2.4 Conclusion

Dans ce chapitre nous avons définit le formalisme de modélisation UML et nous avons fait la description de ses différents diagrammes ainsi que l'analyse et la conception de notre application. Après avoir terminé la conception, notre application est prête à être codée ,nous allons nous intéresser à l'implémentation de notre système en se basant sur la conception détaillée de ce chapitre.

Chapitre 3

Implémentation et Réalisation

3.1 introduction

Dans ce chapitre nous allons présenter notre application :l'environnement logiciel de travail , les différentes étapes de son implémentation.Ces étapes seront présentées sous forme de capture d'écran de l'application.

3.2 Présentation de notre travail

Notre travail consiste à réaliser une plateforme distribuée temps réel, qui contiendra plusieurs applications : une pour chaque type de vendeurs(Permanent, Occasionnel de produit, Occasionnel offrant un service et une entreprise offrant des service) et un site web pour les clients ,chacun de ces vendeurs aura la possibilité de poster une annonces d'un produit à vendre sur le site web , tout en assurant la sécurité des données.

Chaque vendeur aura toutes les fonctionnalités nécessaires afin qu'il fasse un suivi en ligne de son annonce, il pourra donc la consulter, modifier ou la supprimer.

Le vendeur permanent aura aussi la possibilité de gérer son stock de produits dans son magasin.

Un client pourra contacter le vendeur pour avoir plus de détails sur l'annonce .

3.3 Environnement logiciel

- Outil de modélisation : Unified Modeling Language (UML),c'est un langage visuel constitué d'un ensemble de diagrammes, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par le logiciel, etc.

- Editeur de texte : Visual studio code,Visual Studio Code est un éditeur de code open-source, gratuit et multi-plateforme , développé par Microsoft. VSC est développé avec Electron et exploite des fonctionnalités d'édition avancées . Principalement conçu pour le développement d'application avec JavaScript, TypeScript et Node.js, l'éditeur peut s'adapter à d'autres types de langages grâce à un système d'extension bien fourni.comparé à d'autres éditeurs de texte , le niveau visé est plutôt intermédiaire/avancé

- Système de gestion de base de données :Navicat for PostgreSQL ,est une suite logicielle graphique de gestion et de développement de bases de données produits par PremiumSoft CyberTech Lt.Son interface utilisateur graphique est similaire à celle d'Explorer et il supporte des connexions multiples vers des bases de données locales et distantes.

- Serveur :Nodejs , un environnement de développent permettant de créer facilement des applications réseau rapides et évolutives.

- Langages de programmation :
  - html5:Un langage de balisage qui sert à l'écriture de l'hypertexte indispensable à la mise en forme d'une page Web.
  - css3:Un langage de feuille de style utilisé afin de décrire la présentation d'un document HTML.
  - javascript:Un langage de script léger,permet à une page web d'étre interactive et dynamique.

3.4 Schéma global du système

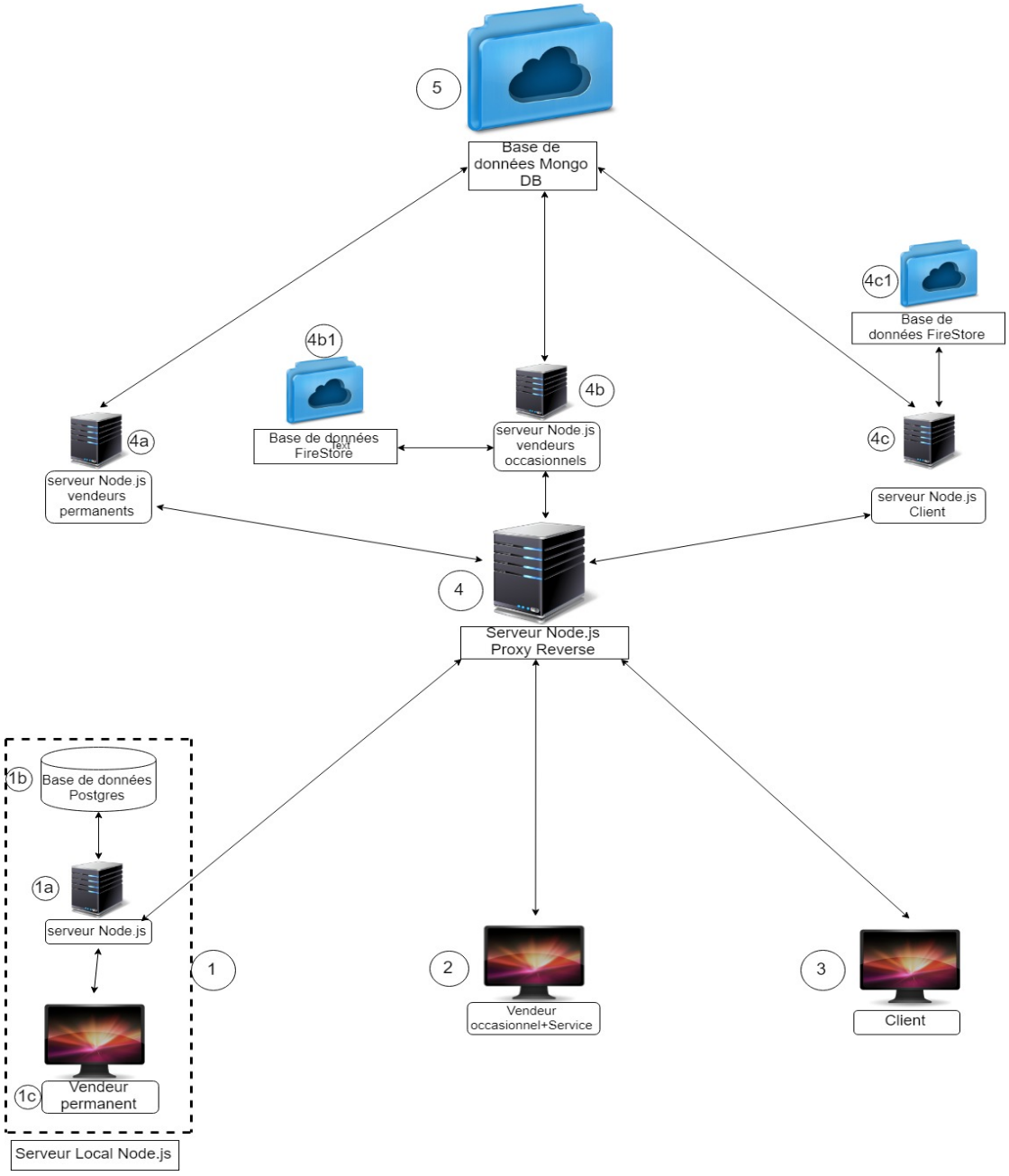


Figure 3.4.1: Schéma global du système

Interprétation du schéma:

- [1] : C'est un pc qui contient application desktop du Vendeur permanent et tous ses outils : serveur local nodejs, le serveur de base de données postgres et un navigateur web.
  - [1a] : Serveur nodejs, contient le code métier de l'application.
  - [1b] : Serveur de base de données SQL postgres.
  - [1c] : Navigateur web
- [2] : Application vendeur occasionnel (de produit,de service,entreprise de service), représente un client léger qui utilise un navigateur web.
- [3] : Site web des annonces pour les clients. Tout comme le vendeur occasionnel, c'est un client léger qui utilise un navigateur web.
- [4] : Serveur Node.js qui fait office d'un proxy inverse « reverse proxy ».
  - [4a] ,[4b], [4c] : Représente respectivement les serveurs node.js des applications « 1 », « 2 », « 3 ».
    - [4b1] , [4c1] :Base de données NOSQL cloud firestore utilisées pour les vendeurs occasionnels et les clients .
- [5] : Service cloud base de données NOSQL Mongo DB, où sont stockées toutes les annonces postées par les vendeurs.

Exemples de chemins des opérations CRUD (create, read, update, delete) :

Vendeur permanent :

- [1a-1c-4-4a-5-1b/erreur] : Pour effectuer les opérations CRUD,le server local Node.js du vendeur permanent contactera d'abord le reverse proxy server qui redirigera ces requêtes vers un autre server Node.js qui s'occupera d'exécuter d'abord les requêtes sur la base NoSQL MongoDB .La réponse prend le chemin inverse vers le server local Node.js : si la réponse est un succès : le server local effectuera les opérations sur sa base de donnée local Postgres, sinon on affiche un message d'erreur (opération échouée).

Vendeur occasionnel :

- [2-4-4b-5-4b1/erreur] : Toutes les opérations CRUD sont directement transférées vers le reverse proxy server qui les redirigera vers le server Node.js, ce dernier exécutera les requêtes d'abords sur la base MongoDB : si la réponse d'exécution est un succès le server Node.js exécutera ces opérations dans la base NoSQL FireStore, sinon le server renvois un message d'erreur au vend occasionnel.

Client :

1. Inscription et connexion [3-4-4c-4c1] : Pour ces deux opérations le client envois les requêtes au reverse proxy qui s'occupera de les transférer au server Node.js, ce dernier exécutera les requêtes sur la base NoSQL FireStore.

2. Recherche[3-4-4c-5] :Le client envois sa requête vers le reverse proxy qui la redirigera vers le server Node.js, ce dernier exécute la recherche sur la base mongo pour récupérer les documents nécessaires, si la réponse est vide, le server Node.js envois un message d'erreur au client (aucun résultat trouver), sinon il envois un tableau de document spécifique à la recherche qui seront affichés sur une carte avec les informations nécessaires.

Remarque : Chaque application a son propre server Node.js dédié, les requêtes passeront par le reverse proxy qui se charge de la redirection vers le server Node.js respective à chaque application.

3.5 Sécurité adaptée

3.5.1 Vendeur permanent

Coté serveur:

Lors de l'authentification du vendeur , on envoi l'email et le mot de passe au server, il les chiffre en utilisant un algorithme symétrique et une clé déclarée au préalable, ensuite une requête est lancée pour récupérer l'email et le mot de passe dans la base de donnée Postgres ,étant chiffrée aussi avec le même algorithme et la même clé ,on fait la comparaison entre hash :

- Si le hash est différent donc les données saisi par le vend sont incorrectes, on le redirige vers la page de connexion,
- Sinon on procède à la création du token à l'aide du module JWT comme suit:
  - Let token = jwt.sign ({email:cryptedEmail, password:cryptedPass}, config1.secret,{algorithm:'HS256'});
  - On déclare un var token où on va stocker le token final sous forme de string,
  - L'object jwt prend trois paramètres :
    1. Les données à utiliser sous forme d'un objet Json qui représente le hash de l'email, et le hash du mot de passe.
    2. La clé secrète sauvegardée dans un fichier.
    3. L'algorithme utilisé et supporté par le JWT.

    Une fois crée, le token est chiffré une deuxième fois en utilisant un autre algorithme et une autre clé, ensuite il sera stocké dans un ficher texte à la racine du server.

    Une fois enregistré, on utilisera le module cron pour exécuter des taches à chaque période précise, dans notre cas à chaque 30 min on vérifie si le fichier texte n'est pas vide, ensuite on récupère de la base de données postgres l'email et le mot de passe chiffrés pour générer un nouveau token et le stocker dans le ficher (mise a jour du ficher texte avec un nouveau token)

Coté client:

Lors du chargement de chaque page html, un socket est envoyer au server pour vérifier le ficher texte:

- Si le fichier n'est pas vide le socket renvois le code, la page peut être chargée.
- Si le ficher est vide, l'utilisateur s'est pas connecté donc le token n'a pas était crée, la page ne sera pas chargéet il sera redirigé vers la page de connexion.

Remarque:

Un autre cron job est chargé de mettre à jour le ficher texte en mettant son contenu à vide chaque 8h, en d'autre terme,après son authentification, la session d'un utilisateur dure 8 h.

3.5.2 Vendeur occasionnel / Client

Coté serveur:

Pour gérer les sessions des utilisateurs du " vendeur occasionnel", on a élaboré un algorithme qui utilise:

- Jwt (module nodejs jsonwebtoken).
- Cookies (module nodejs cookie-parser).
- Cron job (module nodejs cron-job).
- Crypto (module nodejs pour le chiffrement symétrique).
- Crypto-random-string (module nodejs pour générer des clés pour l'utiliser dans le chiffrement).
- Socket.io (module nodejs pour l'envoie de données entre Client-Server en temps réel).

Déroulement de l'algorithme:

Lors de l'authentification, l'émail et mot de passe de l'utilisateur, sont chiffrés avec le module crypto en utilisant une clé secrète déjà déclarée au préalable: (private key) et un algorithme de chiffrement symétrique déjà déclarer aussi. Ensuite on fait le test sur l'existence correcte des identifiants saisis :

dans le cas où ils existent:

1. On va créer en premier une clé de type string en utilisant « crypto-random-string » de taille(20).

2. Ensuite, on génère le premier token jwt en utilisant comme données d'entrée : "l'émail ,le mot de passe chiffré et comme clé celle qui est générée dans (1).

3. En dernier lieu, en stock le token crée sous forme de string dans un cookie avec le nom " session".

4. on appelle la fonction « codeGen(token) » , on lui passe le token comme paramètre d'entrée, cette fonction une fois appelée , elle déclenche une tache répétitive chaque 30 min , cette tache consiste à :

   a) Sélectionner un algorithme de chiffrement de façon aléatoire d'un tableau déclaré au début de la fonction {Var tab= ['HS256','HS384','HS512']};

   b) Une fois un algorithme est sélectionné, on génère une autre clé de taille(20) de type string toujours avec le module" crypto-random-string" ;

   c) On calcul un nouveau token en lui comme donnée l'ancien token passé à la fonction, la nouvelle clé fraichement générée et enfin l'algorithme aléatoirement saisi du tableau.

   d) Un socket va envoyer le nouveau token créer chaque 30 mins au client pour mettre à jour le cookie.

Coté Client:

Au début de chaque page html, on récupère le cookie créer lors de la connexion, on teste:

- s'il est : null /undefined ou bien égale à 0 : on le redirige vers la page de connexion sinon on fait rien.

Après des que le socket coté server va envoyer le nouveau code on met à jour le cookie avec le nouveau code

- la déconnexion du client : la déconnexion va d'abord déclencher une fonction deconnect() : qui se chargera de:

  1. mettre le cookie à une valeur ==0 : //document.cookie='session=0';
  2. elle va rediriger l'utilisateur vers la page de connexion;
  3. elle va envoyer un socket stop au server pour arrêter la tâche répétitive et cesser de générer un nouveau token;

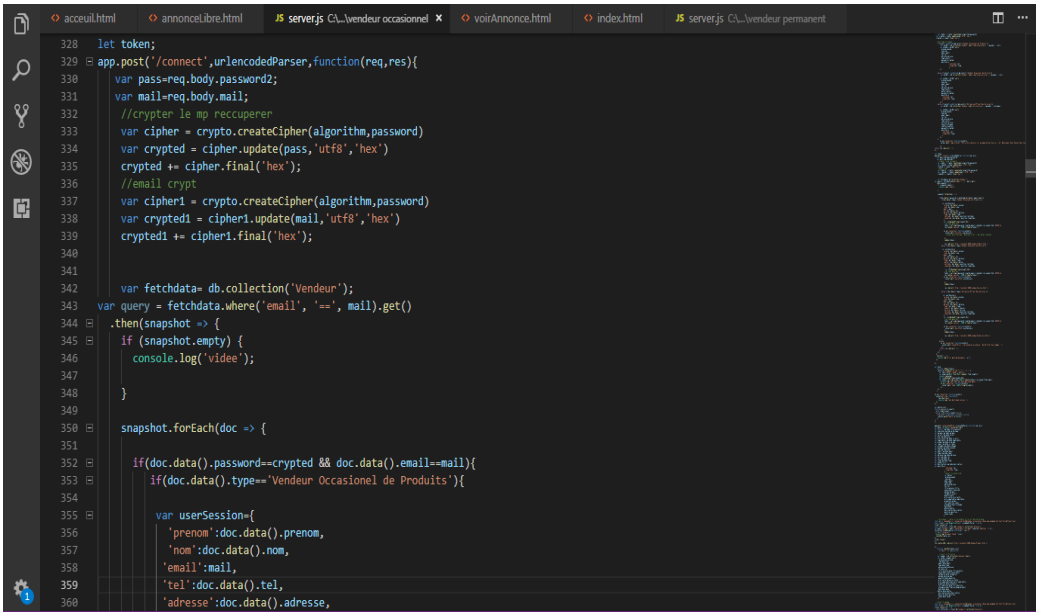


Figure 3.5.1: Algorithme de sécurité vendeur occasionnel/client

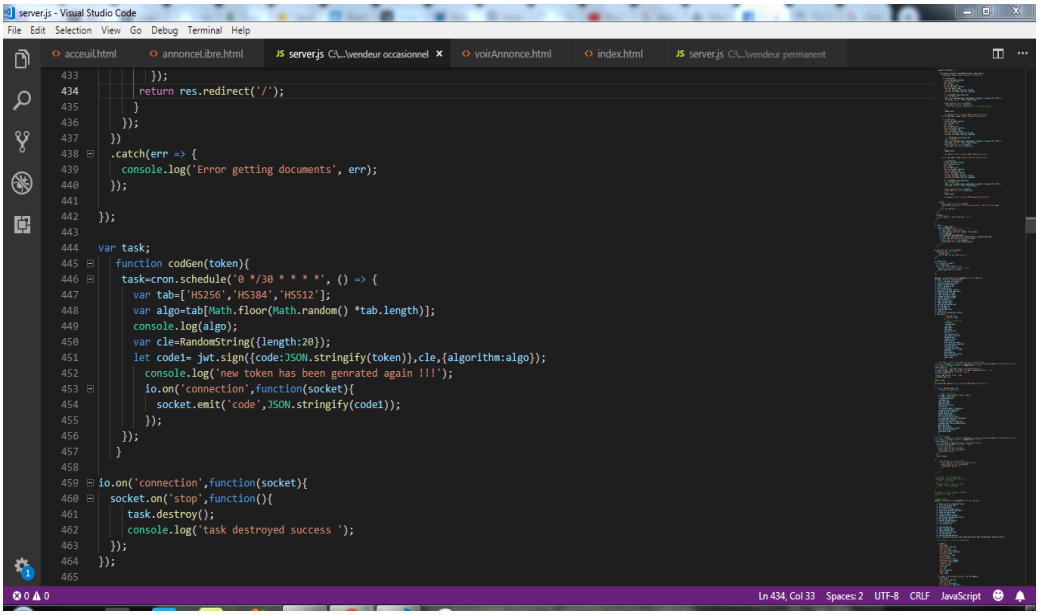


Figure 3.5.2: Algorithme de sécurité vendeur occasionnel/client

3.6 Base de données postgres

- Table produits:

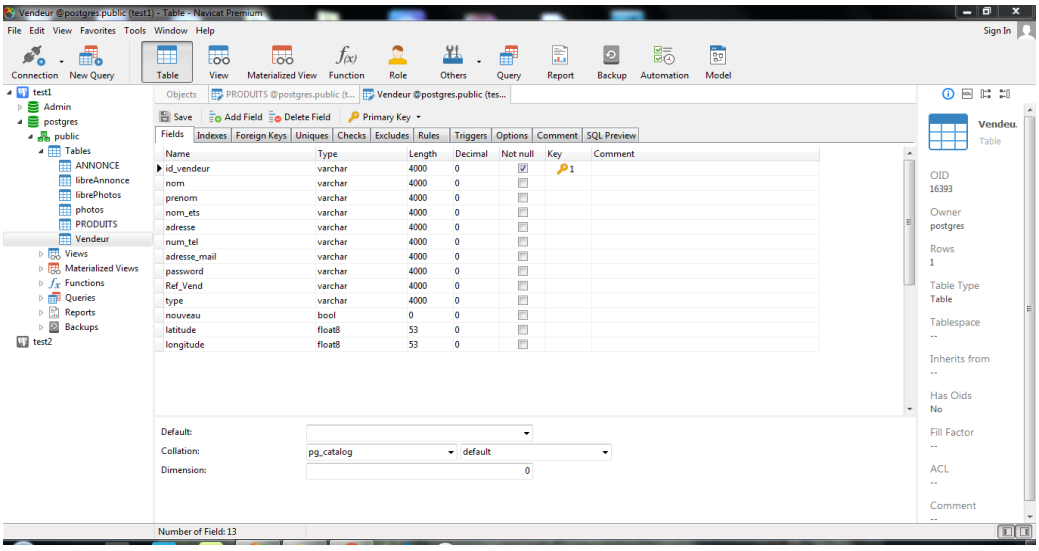


Figure 3.6.1: Table produits:

- Table vendeurs:

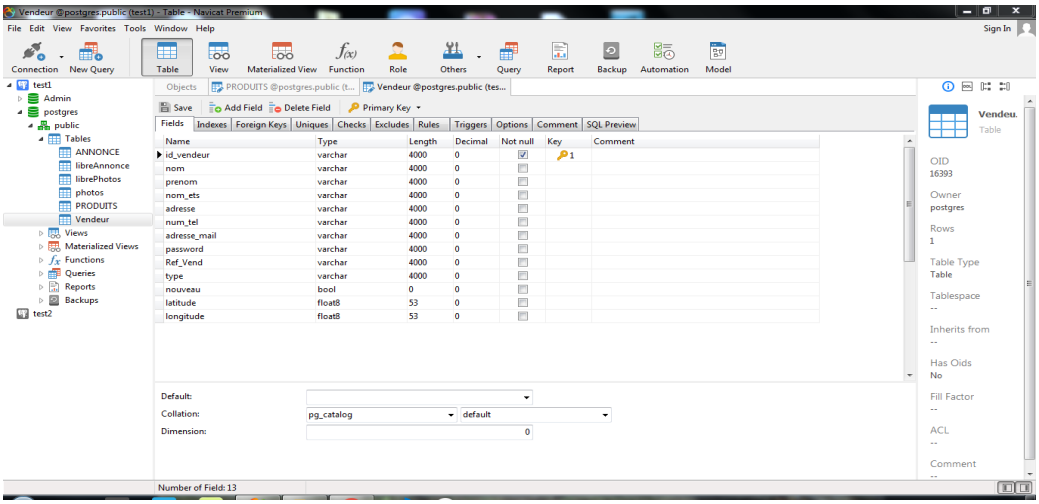


Figure 3.6.2: Table vendeurs:

3.7 Base de données Firebase

- Collection annonces:

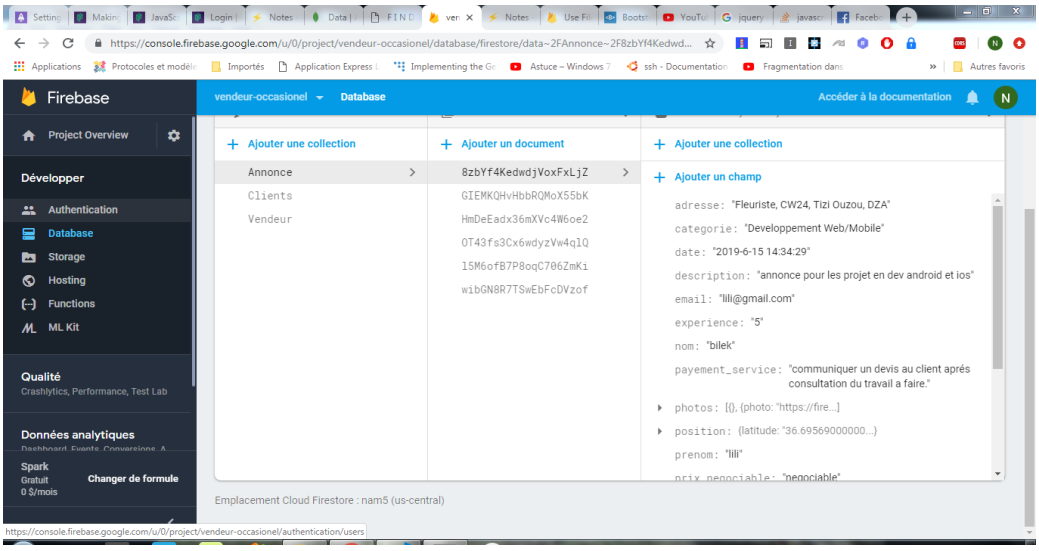


Figure 3.7.1: Base de données Firebase collection “annonce”

- Collection clients:

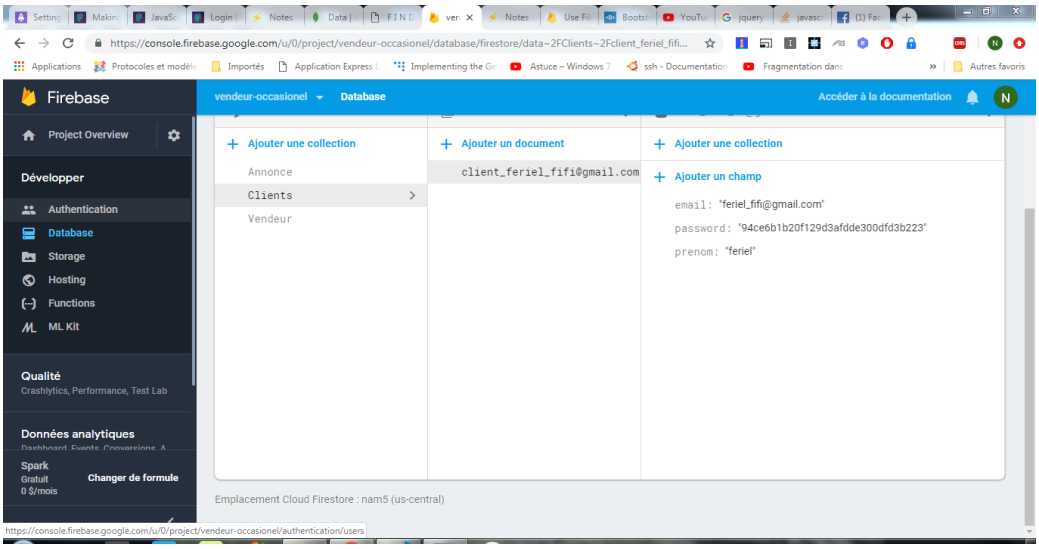


Figure 3.7.2: Base de données Firebase collection “client”

3.8 Base de données MongoDB

- Collection clients:

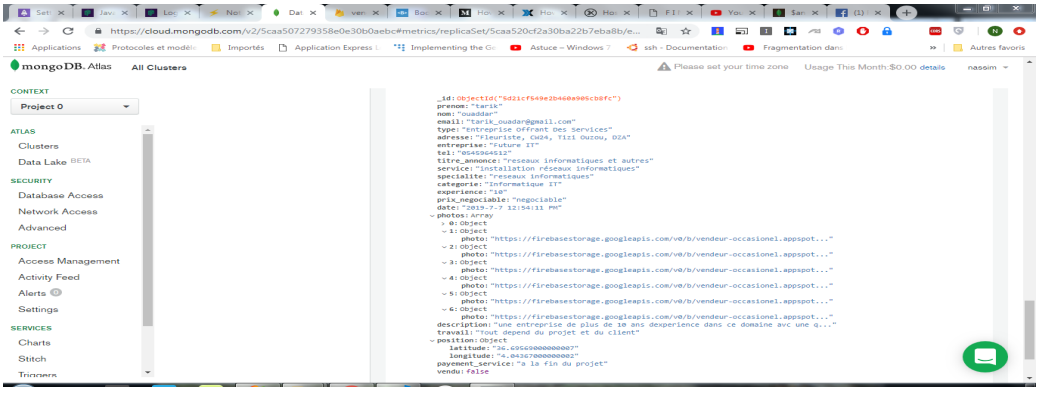


Figure 3.8.1: Exemple de document de la base de données MongoDB”

3.9 les interfaces

3.9.1 Vendeur permanent

- Réinitialisation de mot de passe :
  - lorsque le vendeur se connecte pour la première fois après son inscription, par mesure de sécurité il doit changer le mot de passe attribué par le livreur de l'application:

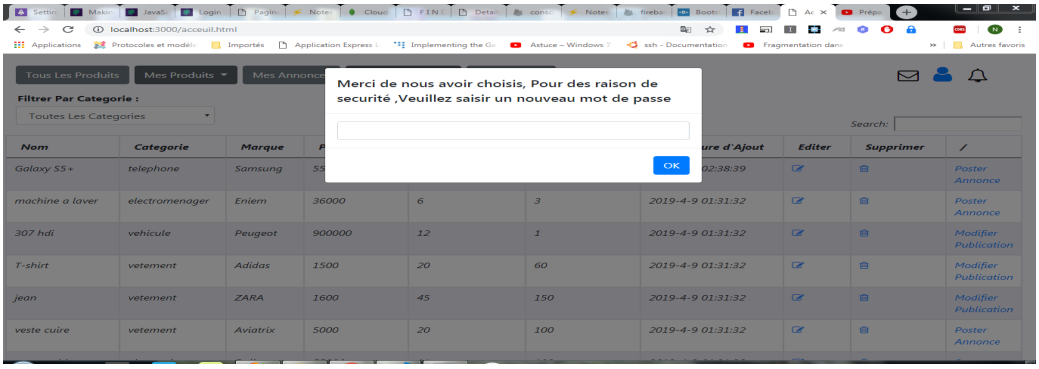


Figure 3.9.1: "Réinitialisation de mot de passe" vendeur permanent

- Interface d'acceuil:

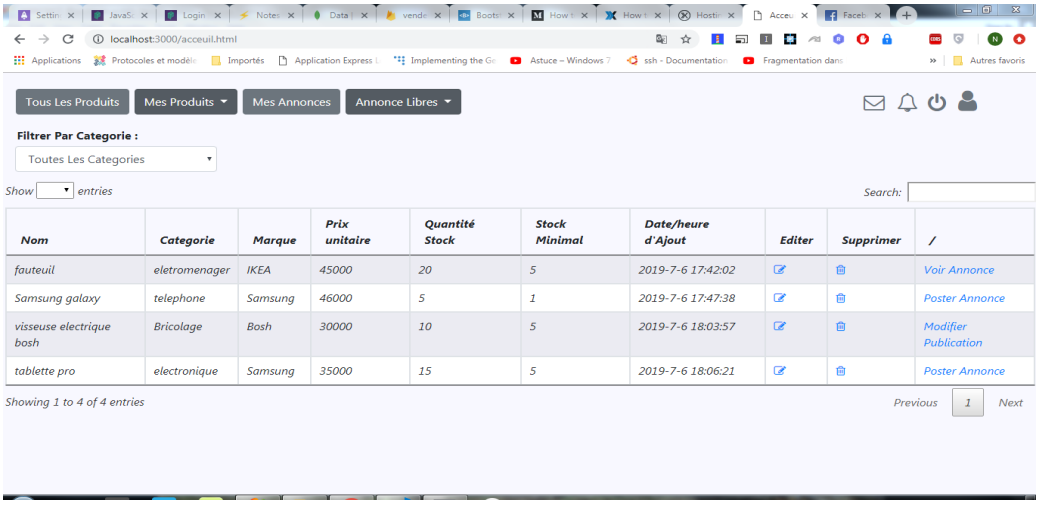


Figure 3.9.2: Interface d'acceuil du vendeur permanent

- Modifer une annonce:

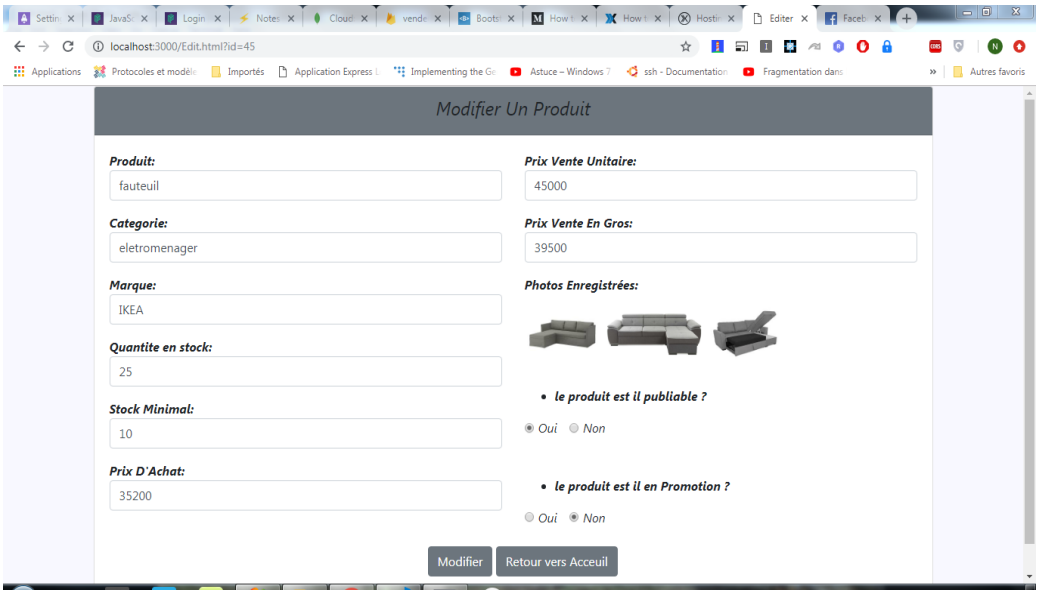


Figure 3.9.3: interface “Modifier une annonce” vendeur permanent

3.9.2 Vendeur occasionnel

- connexion du vendeur occasionnel:

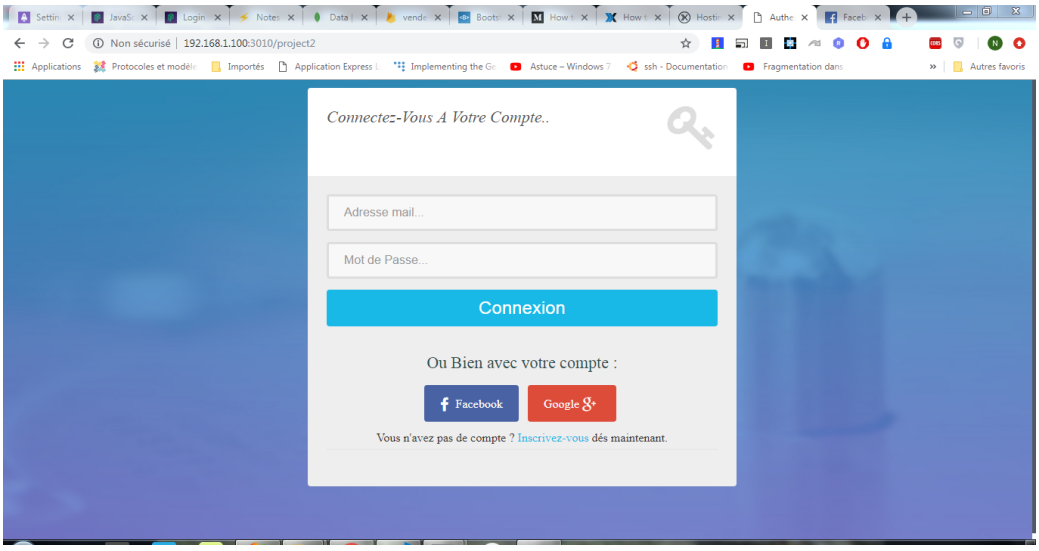


Figure 3.9.4: connexion du vendeur occasionnel

- Interface d'acceuil du vendeur occasionnel offrant un service:

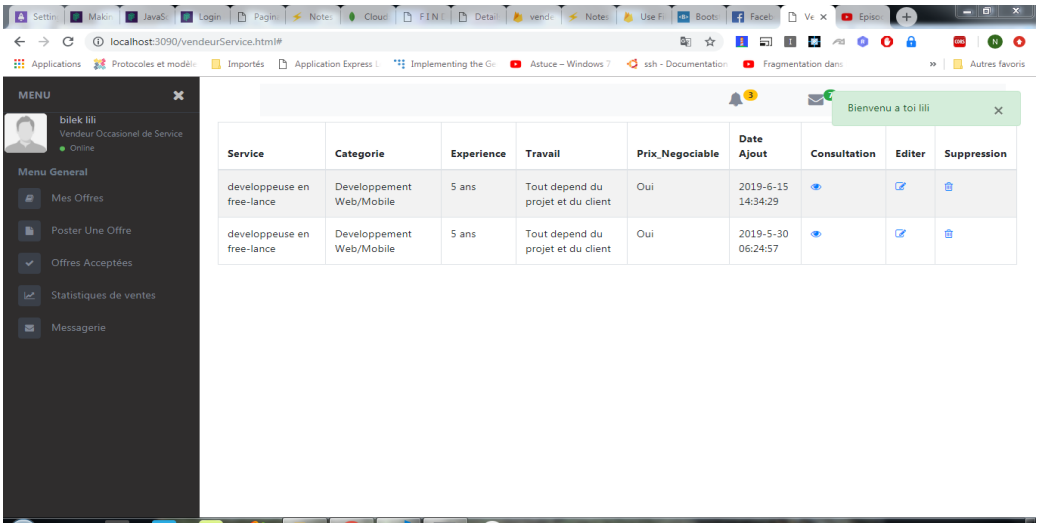


Figure 3.9.5: Interface d'acceuil du Vendeur occasionnel offrant un service

- Page des annonces du vendeur occasionnel de produits :

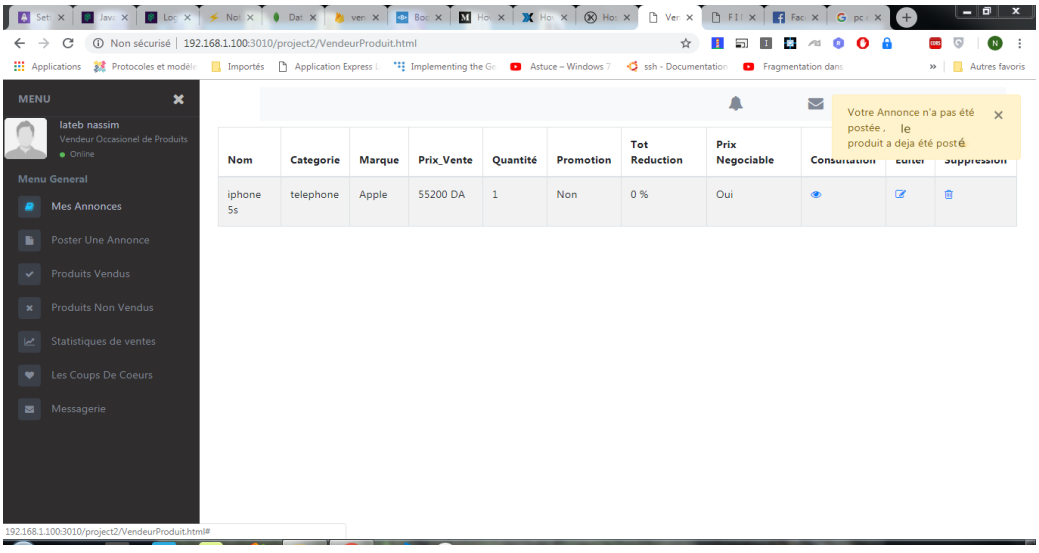


Figure 3.9.6: interface des annonces du vendeur occasionnel de produits

3.9.3 client

- page d'acceuil des visiteurs :

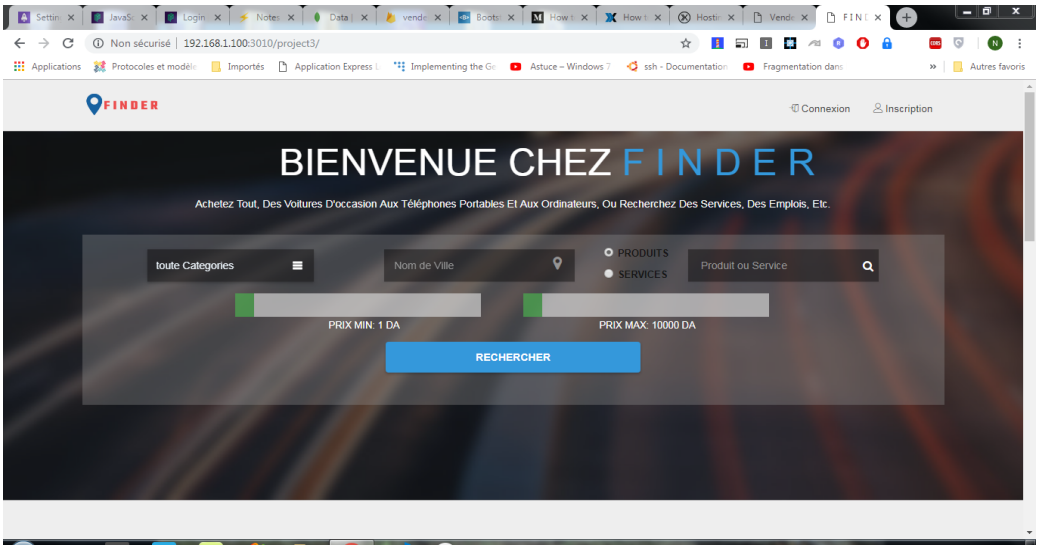


FIGURE 3.9.7 : page d'acceuil d'un visiteur

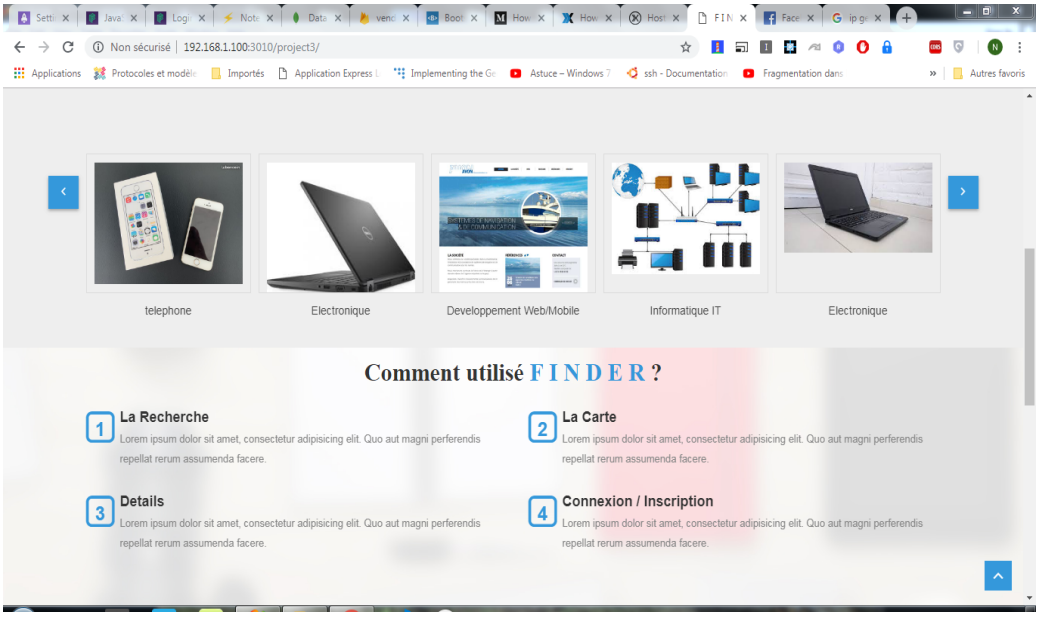


FIGURE 3.9.8 : slide des produits mis en avant

- recherche d'un produit :

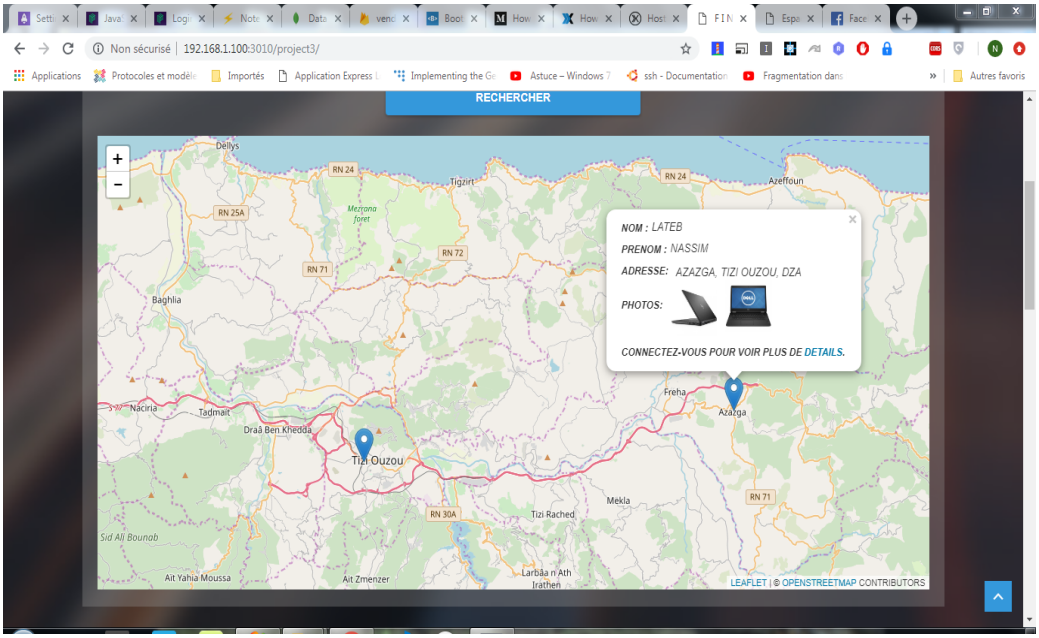


FIGURE 3.9.9 : Rechercher un produit ou une annonce

- connexion du client :

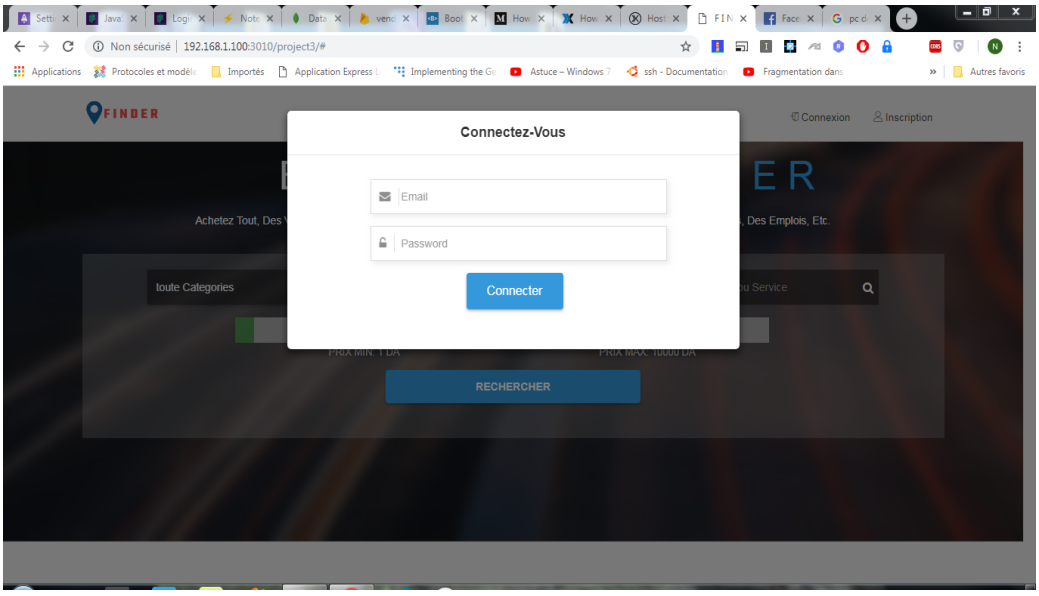


FIGURE 3.9.10 : Page de connexion pour le client

- consulter une annonce :

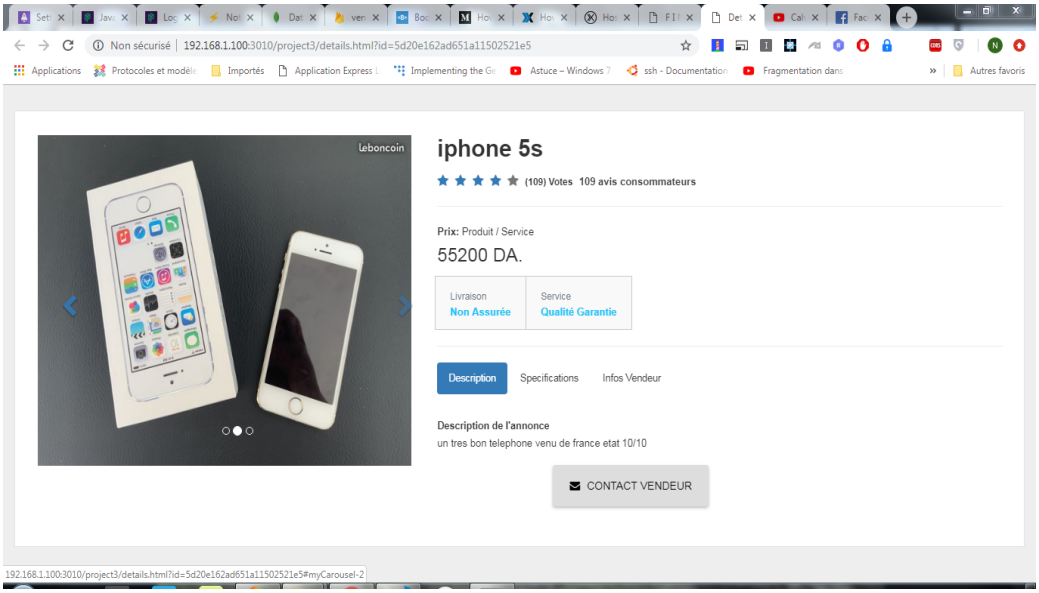


FIGURE 3.9.11 : Consulter le détail d'une annonce

- contacter le vendeur :

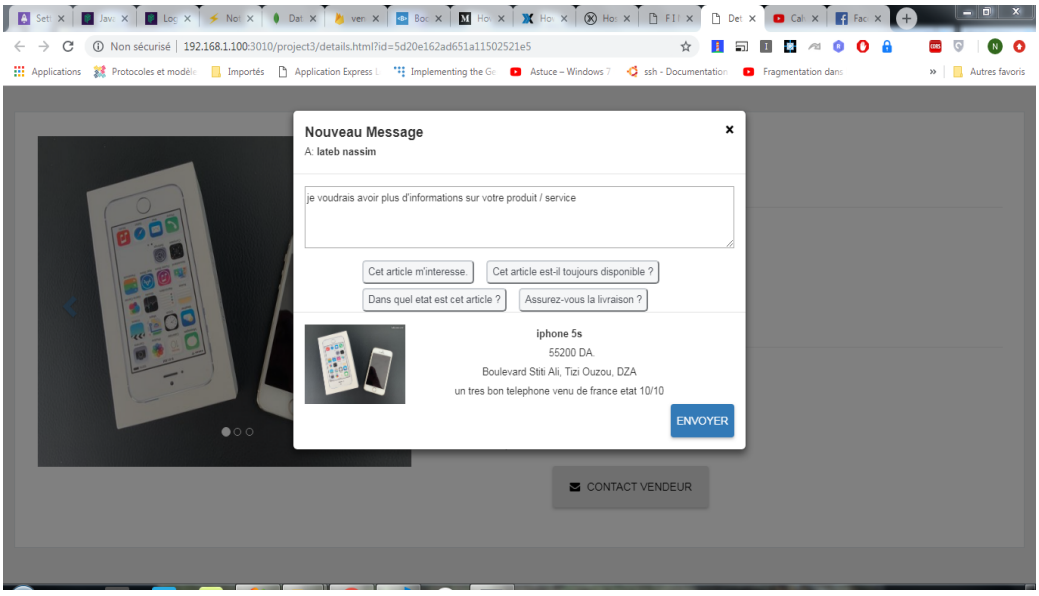


FIGURE 3.9.12 : Page de contact du vendeur

3.10 Conclusion

Nous avons vu à travers ce chapitre les différents outils utilisés pour l'implémentation de notre application, les tables de base données utilisés pour la réalisation de notre projet et les différentes interfaces qui composent notre application, pour terminer ensuite avec la présentation de notre environnement.

Conclusion générale

Conclusion et perspectives

Au terme de notre travail, nous pouvons faire le bilan de notre étude en insistant sur les points suivants :

D'abord le présent projet est notre deuxième expérience ,après le projet réalisé en fin de cycle de licence, il nous a permis de mettre en œuvre et d'appliquer une bonne partie de nos connaissances acquises tout au long de notre cursus, comme il nous a incité à approfondir nos recherches et à nous frotter au monde réel de notre domaine, ce qui nous a encouragé à persévérer dans notre choix du thème.

Apres élaboration de cahier de charge, nous avons opté pour l'utilisation de la plateforme J2EE mais par la suite nous avons rencontré un problème majeur, qui est la lenteur du système, alors que dans ce genre de thème, il est primordial d'utiliser un environnement plus rapide qui assurera la scalabilité et minimiser les temps de réponse des requêtes, c'est pour cela que notre choix s'est penché sur le Node.js, un environnement de développent basé sur le moteur V8 de Google chrome,un outil open source créé par Google qui analyse et exécute du code JavaScript très rapidement.

Par ailleurs ce travail a eu pour objectif primordial à rependre à toutes les exigences et les besoins qu'on a fixés et qui se présentent comme suit :

- Réaliser une plateforme e-commerce distribuée et temps réel.
- Intégrer les différents types de base de données: SQL,NoSQL,Cloud
- Base de données multimédia.
- Elaboration et implémentation des algorithmes de sécurité personnalisés.

Cependant quoiqu'elle présente diverse fonctionnalités, quelques améliorations et extensions peuvent être apportés à notre application, en guise de perspective, les possibilités d'évolutions de ce projet sont nombreuses :

- En premier lieu, intégrer une API de e-payement,ou bien développer notre propre API.
- Ajout au panier.
- Hébergement et déploiement de la plateforme.
- Rajouter quelques fonctionnalités : notification des produits prochainement disponibles, intégrer un live où le vendeur pourra faire un live pour présenter son produit dans forum, faire les statistique de vente pour les vendeurs.
- Réalisation d'une API REST pour le vendeur permanent qui sera utilisée par une application mobile.
- Application mobile pour les vendeurs et le client.
- Offrir aux clients la possibilité d'imprimer les détails d'une annonce d'un service , afin qu'il garde trace .

D'autre part on espère que ce mémoire pourra servir comme référence pour les promotions futures en Informatique ou comme documentation pour tirer des renseignements utiles et des compléments à leurs études vu sa richesse en informations dans ce domaine choisi.

Webographie

- [1] https://www.mongodb.com/nosql-explained?lang=fr-fr
- [2] https://www.digora.com/fr/blog/definition-base-nosql-datastax-mongodb
- [3] https://www.mongodb.com
- [4] https://www.json.org/json-fr.html
- [5] https://blog.postgresql.fr/
- [6] https://nodejs.dev/
- [7] https://www.lebigdata.fr/base-de-donnees#ftoc-heading-11
- [8] https://www.ibm.com/fr-fr/cloud/learn/what-is-cloud-database
- [9] https://whatis.techtarget.com/definition/Nodejs
- [10] https://www.tutorialsteacher.com/nodejs/nodejs-process-model
- [11] https://www.memoireonline.com/02/17/9572/m_Mise-en-oeuvre-dun-systeme-distribue-pour-lidentification-et-le-suivi-du-casier-judiciaire14.html
- [12] https://sebsauvage.net/comprendre/proxy/index.html
- [13] https://fr.vpnmentor.com/blog/les-differents-types-de-vpn-et-quand-les-utiliser/
- [14] https://firebase.google.com/docs/firestore/
- [15] https://stackoverflow.com/
- [16] https://www.npmjs.com/
- [17] https://medium.com/
- [18] http://tutorialpoint.com/
- [19] https://openclassrooms.com/fr/courses/2035826-debutez-lanalyse-logicielle-avec-uml/2035851-uml-c-est-quoi

- [20] http://dspace.univ-tlemcen.dz/bitstream/112/8812/1/Etude-comparative-des-bases-de-donnees-NoSQL.pdf
- [21] https://lear.inrialpes.fr/~douze/enseignement/2014-2015/cours_chap1_2pp.pdf,Bases de données multimédia , ENSIMAG 2014-2015 Matthijs Douze & Karteek Alahari
- [22] http://lsds.hesge.ch/distributed-systems/,Nabil Abdennadher, nabil.abdennadher@hesge.ch 2015/2016 Semestre d'Automne
- [23] http://ecariou.perso.univ-pau.fr/cours/sd-l3-old/cours-1-intro.pdf.Introduction aux Systèmes Distribués Introduction générale ,Eric Cariou, Université de Pau et des Pays de l'Adour Département Informatique,Eric.Cariou@univ-pau.fr
- [24] http://www.redcad.org/members/tarak.chaari/cours/coursSD.pdf,tarak.chaari@isecs.rnu.tn Maître assistant à l'institut supérieur d'électronique et de communication
- [25] https://support.zendesk.com/hc/fr/articles/203663816-Activation-de-la-connexion-unique-avec-JWT-Token-Web-JSON-,8 mai 2019
- [26] https://www.ekino.com/articles/introduction-aux-json-web-tokens
- [27] https://riptutorial.com/node-js/example/21767/setting-cookies-with-cookie-parser
- [28] https://www.linuxtricks.fr/wiki/cron-et-crontab-le-planificateur-de-taches
- [29] http://sdz.tdct.org/sdz/les-sockets.html
- [30] http://www.i3s.unice.fr/~tettaman/Classes/L2I/ProgSys/11_IntroSockets.pdf
- [31] https://www.globalsign.fr/fr/centre-information-ssl/definition-certificat-ssl/
- [32] https://www.lemagit.fr/definition/Pare-feu-dapplications-Web