|  |  |  |
| --- | --- | --- |
| GTI | **REPUBLIQUE TUNISIENNE**  \*\*\*\*\*  MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE  \*\*\*\*\*  DIRECTION GENERALE DES ETUDES TECHNOLOGIQUES  \*\*\*\*\*  INSTITUT SUPERIEUR DES ETUDES TECHNOLOGIQUES DE CHARGUIA  \*\*\*\*\*  **Département Technologies de l’Informatique** | Institut Supérieur des Etudes Technologiques de Charguia ... |

**RAPPORT DE**

|  |
| --- |
| Stage de perfectionnement |

Elaboré par

**Yosr AMAMOU**

**Encadré par :**

**Mr** Mohamed Slaimi

**Société d’accueil :** Générale Tunisienne de l’informatique (G.T.I)

**Année Universitaire : 2023/2024**

Remerciements

Aux termes de ce travail, nous saisissons l’occasion pour réserver ces quelques lignes pour remercier chaque personne qui a contribué au succès de notre stage.

Tout d’abord, nous voudrons remercier Monsieur M. Riadh Frikha, le directeur général de GTI pour son accueil et sa confiance précieuse dès notre arrivée dans l’entreprise.

Nous remercions énormément notre tuteur Monsieur Mohamed Slaimi, ingénieur logiciel au niveau de la GTI, qui a fait preuve d’un esprit très généreux depuis le premier jour en donnant de l’importance à notre candidature, qui a cru en nos capacités, qui nous a attribué des missions valorisantes et qui nous a fait confiance durant toute la période de stage.

Son expertise et ses conseils avisés ont été des éléments essentiels qui ont permis de donner forme à notre projet. Grâce à sa disponibilité, sa patience et son suivi, nous avons pu surmonter les obstacles techniques et relever les défis rencontrés lors du développement de l'application.

Nous sommes reconnaissants pour sa capacité à partager ses connaissances et son expérience, ce qui a enrichi notre apprentissage et a élargi nos compétences dans le domaine du développement.

Nous remercions également Monsieur Aymen Ben Nasr et Mademoiselle Rania Kacem, ingénieurs logiciels, pour leurs conseils avisés qui nous ont permis d’acquérir des compétences dans le domaine du développement, pour leurs disponibilités et leurs bonnes humeurs. Ils ont toujours été à l’écoute et ont participé à rendre notre stage plus enrichissant et captivant. Leurs expertises et leurs soutiens ont été des éléments clés dans la résolution de certaines difficultés rencontrées lors du développement de l’application.

Nous remercions toute l’équipe GTI qui fut bien accueillante et collaborative.

Ainsi que nos reconnaissances amicales à l’ensemble du personnel de GTI pour l’agréable environnement de travail qu’ils nous ont offert et du temps qu’ils nous ont consacré pour enrichir notre apprentissage et faciliter nos tâches.

Nous remercions tous ceux qui ont collaboré de près ou de loin à l’accomplissement de ce travail.

**Sommaire**

[Introduction générale 1](#_Toc159797689)

[Chapitre 1 : Présentation du cadre du projet 2](#_Toc159797690)

[I. Présentation de la société 2](#_Toc159797691)

[II. Présentation du domaine métier 3](#_Toc159797692)

[III. Etude de l’existant 4](#_Toc159797693)

[1. Description de l’existant 4](#_Toc159797694)

[2. Critique de l’existant 4](#_Toc159797695)

[3. Solution proposée 5](#_Toc159797696)

[1. Besoins fonctionnels 6](#_Toc159797697)

[2. Besoins non fonctionnels 8](#_Toc159797713)

[Chapitre 2 : modélisation comportementale de l’application 11](#_Toc159797728)

[I. Identification des acteurs 11](#_Toc159797729)

[II. Présentation du diagramme de cas d’utilisation 11](#_Toc159797733)

[1. Diagramme du cas d’utilisation général 12](#_Toc159797734)

[2. Description textuelle des cas d’utilisation 12](#_Toc159797737)

[1. Diagrammes de séquence système 15](#_Toc159797783)

[2. Diagrammes de séquence objet 18](#_Toc159797785)

[3. Diagramme d’activité 25](#_Toc159797796)

[Chapitre 3 : Modélisation structurelle de l’application 27](#_Toc159797800)

[I. Diagramme de classes 27](#_Toc159797801)

[1. Descriptif du diagramme 27](#_Toc159797802)

[2. Représentation du diagramme 28](#_Toc159797803)

[II. Modélisation de la base de donnée 29](#_Toc159797804)

[1. Modèle relationnel 29](#_Toc159797805)

[Chapitre 4 : Réalisation 31](#_Toc159797806)

[I. Environnement de développement 31](#_Toc159797807)

[1. Environnement matérielle 31](#_Toc159797808)

[2. Environnement logiciel 31](#_Toc159797809)

[3. Présentation de l’environnement logiciel 32](#_Toc159797810)

[II. Création du JWT à partir d’une Authentication ? 38](#_Toc159797811)

[III. Principales interfaces graphiques 41](#_Toc159797812)

[1. Interfaces graphiques 41](#_Toc159797813)

**Liste des figures**

**Listes des tableaux**

# Introduction générale

# Chapitre 1 : Présentation du cadre du projet

Ce premier chapitre vise à offrir une vue d'ensemble du cadre du projet, mettant en lumière les aspects essentiels de la société de stage et du domaine métier dans lequel elle opère. À travers cette présentation, nous plongeons dans l'univers de la technologie financière, explorant les défis et les opportunités qui ont conduit à la conception de l'application « e-crédit ».

## Présentation de la société

## 

La générale tunisienne de l’informatique (GTI) est un acteur majeur en matière de logiciels et services informatiques dédiés aux institutions financières.

Cette société a des relations suivies avec le secteur bancaire et les intermédiaires en bourse tel que : BH banque, BNA banque, BIAT, UIB et spécialement ATTIJARIWAFA BANK.

Grâce à ses solutions, plusieurs acteurs bancaires de références optimisent leur efficacité, leur souplesse et leur capacité de réaction face aux enjeux de leurs activités.

La GTI est responsable de tous travaux intègres ou sectoriels d’étude, de réalisation, de planification et d'organisation en informatique et en générale, dans tous les domaines de l’activité bancaire notamment, ainsi que la définition et la conception (conseil de direction, diagnostic des systèmes de gestion, étude d’opportunité, bilan informatique, organisation, robotique…).

La générale tunisienne de l’informatique est présente dans plus de 8 pays dans le monde et leurs services sont déployés dans plusieurs pays dans le monde.

Etablie en Tunisie depuis plus de 40 ans, la GTI a été créé avec la volonté d’importants acteurs financiers de la place pour miser sur le développement de l’informatique depuis 1982, la banque du Sud (actuellement Attijaribank) était l’actionnaire majoritaire.

Depuis 2019, la GTI appartient au groupe ATTIJARIWAFA BANK, premier groupe bancaire et financier du Maghreb présent dans 25 pays : en Afrique, en Europe et en Asie.

Ce tableau illustre des informations générales sur la société :

Tableau 1:informations générales sur la société

|  |  |
| --- | --- |
| Raison sociale | La générale tunisienne de l’informatique |
| Directeur générale | M. Riadh Frikha |
| Forme juridique | Société anonyme |
| Capital | 400.000 dt |
| Secteur | Secondaire |
| Nombre d’employés | 70 |
| Actionnaires majoritaires | ATTIJARIWAFA BANK |
| Personne chargée des relations publiques | Le directeur de projet M. Amine Arous et le PMO M. Anouar Sdiri |
| Adresse | Immeuble new Tower, centre urbain nord,  Tunis |
| CP/Ville/Pays | 1073/Tunis/Tunisie |
| Tél. | 71951703 |
| Fax | 71285361 |
| E-Mail | gti.informatique@gtiinfo.com.tn |
| Site web | www.gtiinfo.com.tn |
| Registre national des entreprises (RNE) | 0014303L |
| Matricule Fiscale | 0014303L/A/M/000 |

## Présentation du domaine métier

La compréhension du secteur financier est crucial pour notre projet « e-crédit ». En explorant les méandres des demandes de crédit, nous plongeons au cœur des opérations bancaires où les transactions, les évaluations de risques et les décisions de prêt se nouent et se dénouent, c’est là que notre application trouve sa place.

« E-crédit » vise à simplifier la gestion des demandes de crédit en ligne, offrant aux clients une expérience fluide et sécurisée tout en simplifiant le processus pour les institutions financières.

En s’appuyant sur une connaissance approfondie des pratiques bancaires et des règlementations financières, notre objectif est de proposer une solution efficace qui répond aux besoins des utilisateurs.

## Etude de l’existant

L’exploration du paysage des applications de crédit en ligne révèle une diversité d’offres et de fonctionnalités déjà présentes sur le marché financier et spécialement au niveau de la société GTI. Cette étude approfondie permet de saisir les subtilités des solutions existantes, d’identifier leurs forces et leurs faiblesses, et de définir les fondements d’une innovation significatif avec l’application « e-crédit ».

### Description de l’existant

La société GTI propose une application « Aréna Risque » dédiée pour le secteur bancaire et qui couvre le parcours du traitement de demande de crédit. L’application « Aréna risque » offre une gamme de fonctionnalités attrayantes, tels que la soumission électronique des demandes, le suivi en temps réelle de leurs statuts, ainsi que la fourniture d’informations détaillés sur les offres de crédit disponibles. De plus, cette solution intègre des outils de simulations de prêt et des conseils financiers pour guider les clients dans leurs décisions.

L’application « Aréna Risque » présente certaines subtilités dans le processus de demande de crédit, notre solution « e-crédit » vise à couvrir ces aspects de manière plus complète en offrant une réponse proactive aux défis et aux opportunités identifiées par le processus de demande de crédit.

### Critique de l’existant

Malgré ces fonctionnalités apparemment robustes, des lacunes subsistent dans l’expérience utilisateur et dans la prestation des services. Parmi ceux-ci :

* + Interface complexe et peu conviviale : La complexité des interfaces utilisateurs peut créer des obstacles à l’engagement, particulièrement pour les utilisateurs moins familiarisés avec les technologies numériques. Les formulaires longs et mal organisés peuvent décourager les utilisateurs de remplir les informations nécessaires, réduisant ainsi le nombre de demandes soumises,
  + Utilisation de logiciels et technologies obsolètes : L’évaluation de la solution actuelle au niveau de la société GTI met en lumière une forte dépendance à l’égard des logiciels et des technologies héritées. Cette solution héritée présente des limitations importantes. Son architecture basée sur des Frameworks tels que JSF (Java Server Faces), struts et d’autres bibliothèques vieillissantes la rend moins agile et plus difficile à mettre à jour pour répondre aux évolutions rapides du marché financier. Ces technologies, bien qu'elles aient été largement utilisées dans le passé, peuvent maintenant entraîner des problèmes de performances médiocres, des pannes fréquentes et une difficulté à s'adapter aux besoins changeants du marché. Par ailleurs, la solution actuelle au niveau de la société peine à répondre aux besoins spécifiques des institutions financières et de leurs clients en raison de leurs manques de flexibilité et d’adaptabilité.
* Sécurité insuffisante : Un autre défi réside dans la sécurité des données. Les technologies anciennes utilisées au niveau de l’application « Aréna risque » tels que JSF et Hibernate sont souvent sujettes à des failles de sécurité exposant ainsi les informations sensibles des clients a des risques potentiels de violation des données et de cyberattaques.

### Solution proposée

### 

Face à ces enjeux, notre application « e-crédit » s’inscrit dans une démarche de transformation positive de l’expérience de demande de crédit en ligne. En mettant l’accent sur la simplicité, la sécurité et l’innovation, « e-crédit » vise à redéfinir les normes du secteur.

La solution « e-crédit » que nous élaborons pour résoudre le défi des demandes de crédit électroniques représente l’apogée de la sophistication technologique :

* L’interface utilisateur, conçue avec un souci obsessionnel du détail, guide les clients à travers chaque étape du processus de demande de crédit de manière intuitive, tandis que le système de gestion des demandes fournit aux responsables de la banque une visibilité complète et en temps réel sur l'ensemble du flux de travail.
* En intégrant une architecture à la fois robuste et flexible, basée sur des technologies de pointe tels que Angular et Spring boot, notre solution offre une expérience utilisateur incomparable marquée par sa simplicité d’utilisation.
* La sécurité des données, au cœur de notre approche, est assurée par des mesures avancées de protection et de cryptage, offertes par Spring Security, garantissant la confidentialité absolue des informations personnelles des clients à chaque instant.

En réduisant les délais de traitement, en automatisant les tâches répétitives et en optimisant les processus internes, notre solution permet aux institutions financières de libérer leur plein potentiel et de se concentrer sur l'essentiel : offrir des services exceptionnels à leurs clients.

Par sa capacité à s'adapter aux besoins changeants du marché financier et à anticiper les évolutions futures, notre solution offre aux institutions financières un avantage concurrentiel indéniable. Elle incarne l'excellence technologique et l'engagement envers l'innovation, ouvrant la voie à une nouvelle ère de performance et de réussite dans le secteur financier.

1. Cahier des charges

### Besoins fonctionnels

Les besoins fonctionnels représentent l’ensemble des fonctionnalités indispensables pour garantir le bon fonctionnement et l’utilité de l’application « e-crédit ». La solution permet aux clients de soumettre une demande de crédit en remplissant un formulaire en ligne suivant différentes étapes. Les demandes sont ensuite envoyées au chargé de la banque pour être instruites et confirmées ou infirmées. Ce dernier peut lui-même passer des demandes de crédit. Ceux-ci sont sécurisés par une authentification obligatoire.

#### Authentification

L’utilisateur (client ou chargé de la banque) doit s’authentifier en saisissant un e-mail et un mot de passe.

#### Rôle du client

* **Demande de crédit**

Le **client** doit saisir son CIN. Le système génère, alors, automatiquement le nom, le prénom, la date de naissance, la situation familiale et les numéros des comptes relatifs à ce client. Dans ce cas, le client doit sélectionner le numéro du compte pour traiter la demande de crédit. A partir de ce numéro, la devise et la date d’ouverture de ce compte seront générées automatiquement par le système.

Le client doit remplir son dossier crédit. Il doit sélectionner le type de crédit désiré et saisir le montant qu'il souhaite emprunter. Le système applique un taux d'intérêt spécifique sur chaque type de crédit, pouvant varier d'un type à l'autre. Le client sélectionne l’unité de crédit. A partir du type et de l’unité choisis, le système calcule le nombre d’échéance.

Le client doit fournir des garanties pour sécuriser sa demande de crédit et assurer le remboursement, il a la possibilité d'ajouter, de modifier ou de supprimer les garanties proposées selon ses besoins.

Le client doit saisir le nom du suivi de sa demande de crédit (la banque ou l'institution financière assigne un agent chargé de suivre son dossier) ainsi que la date à laquelle la relation a été établie avec ce dernier.

Le système affiche, alors, les documents nécessaires pour la demande de crédit selon le type de crédit choisie. Le client doit importer ces fichiers et indiquer leurs états de traitement, confirmant ainsi s'ils ont été soumis ou non,

Le client peut présenter des observations ou des remarques supplémentaires.

* **Consultations des demandes de crédit**

Le client peut consulter ses propres demandes de crédit et suivre leur état en temps réel.

#### Rôle du chargé de la banque

#### 

Le **chargé de la banque** s’occupe d’instruire les demandes crédit déposées par les clients. Il peut être aussi délégué par certains clients pour leur faire le processus de la demande crédit.

* **Consultations des demandes de crédit**

Le chargé de la banque peut consulter toutes les demandes de crédit en attente dans le système. Il peut prendre des décisions éclairées concernant la validation ou le rejet des demandes en cours.

Ces besoins fonctionnels visent à simplifier le processus de demande de crédit pour les clients tout en fournissant aux responsables de la banque les outils nécessaires pour gérer efficacement les demandes.

### Besoins non fonctionnels

Les besoins non fonctionnels de l’application « e-credit » definissent les aspects qualificatifs et les contraintes techniques qui sous-tendent son developpement. Bien que les besoins fonctionnels decrivent ce que l’application doit faire, les besoins non fonctionnels specifient comment elle doit le faire et quels criteres de qualites elle doit respecter.

Pour cela, nous presentons un apercu des principeaux aspects des besoins fonctionnels pour l’application « e-credit » :

* + **Convivialité et ergonomie**
* Concevoir une interface utilisateur intuitive et conviviale pour une expérience utilisateur optimale,
* Assurer une navigation fluide à travers les différentes fonctionnalités de l’application,
* Veiller à ce que le formulaire et les interactions utilisateur soient clairs et faciles à comprendre,
* **Performance et réactivité**
* Optimiser le temps de réponse de l’application pour garantir une expérience utilisateur fluide et efficace,
* Minimiser le temps de chargements des pages et des fonctionnalités pour une utilisation plus agréable
* **Sécurité et confidentialité**
* Mettre en œuvre des mesures de sécurité robustes pour protéger les données utilisateur et les transactions financières,
* Utiliser des protocoles de cryptage et des techniques d’authentification avancées pour garantir la confidentialité des informations sensibles,
* **Contraintes techniques**
* Utiliser des technologies modernes et évolutives telles que Angular et Spring boot pour assurer la flexibilité et la maintenabilité de l’application,
* Veiller à ce que l’architecture de l’application soit extensible et évolutive pour répondre aux besoins futurs.

En respectant ces besoins non fonctionnels, l’application « e-crédit » pourra offrir une expérience utilisateur de haute qualité tout en garantissant la sécurité et la confidentialité de l’utilisateur.

En conclusion, ce chapitre inaugural nous a plongé dans l'univers de la transformation numérique dans le secteur financier, jetant les bases de notre projet « e-crédit ». Avec la société GTI comme partenaire et une vision claire de nos objectifs, nous sommes prêts à franchir les prochaines étapes vers l'innovation et l'excellence. Dans les chapitres à venir, nous approfondirons notre exploration pour concrétiser notre vision d'une expérience de demande de crédit en ligne simplifiée, sécurisée et efficace.

# Chapitre 2 : modélisation comportementale de l’application

## Identification des acteurs

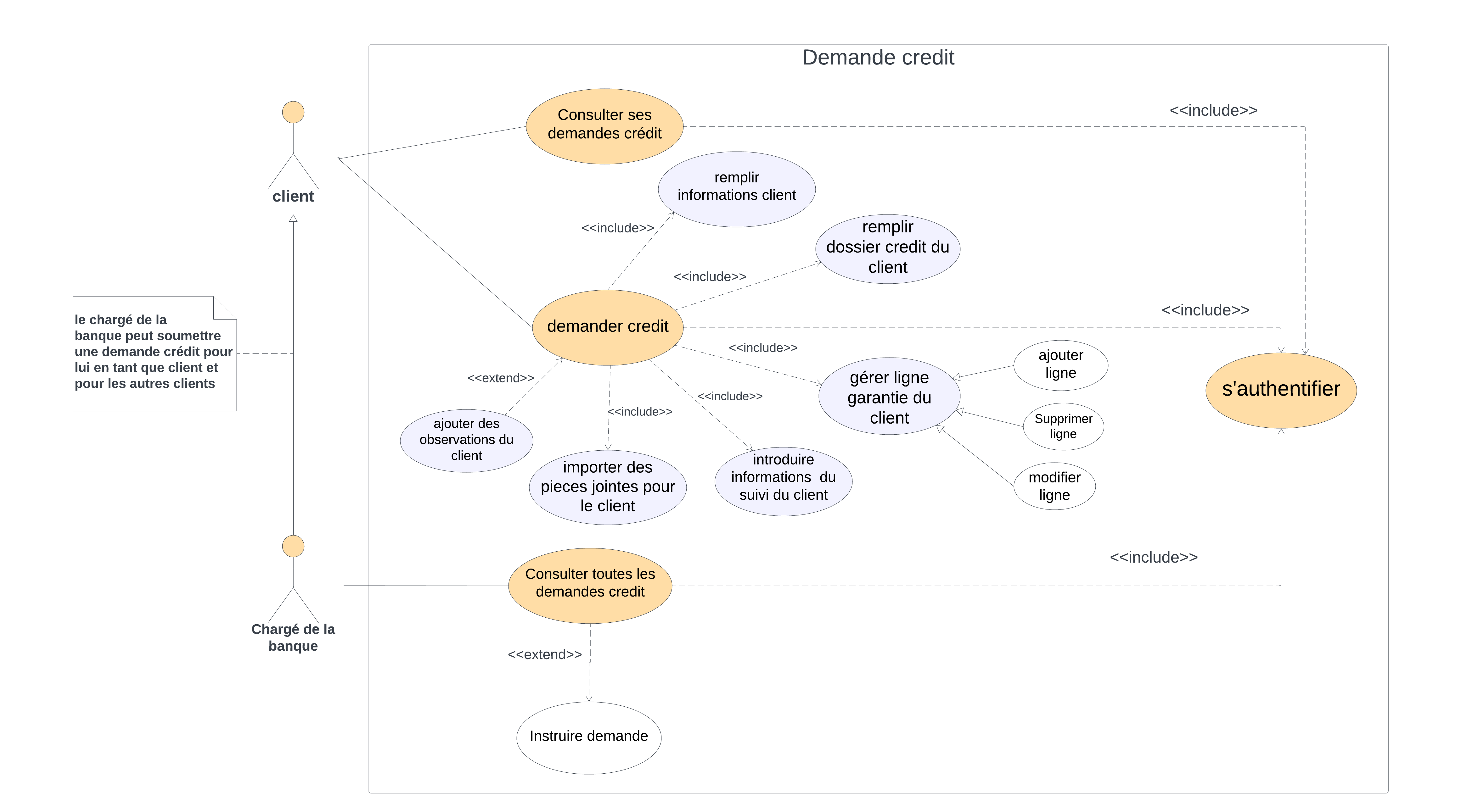
Le système est sollicité par deux acteurs principeaux :

* Client : Le client se connecte avec son e-mail et son mot de passe, remplit un formulaire en ligne (ses informations personnelles, son dossier crédit, ses garanties, les informations de son suivi, les pièces jointes nécessaires, les observations). Une fois le formulaire soumis, il peut suivre en temps réel l'état de sa demande (en cours de traitement, confirmée ou rejetée) et se déconnecter.
* Chargé de la banque : Le chargé de la banque se connecte avec son e-mail et son mot de passe. Il peut visualiser la liste des demandes soumises par les clients. Après examen, le chargé de la banque a le pouvoir de prendre des décisions concernant l'approbation, le rejet ou la mise en attente des demandes pour des vérifications supplémentaires. De plus, le chargé de la banque a la capacité d'instruire des demandes de crédit pour lui-même ainsi que pour d'autres clients.

## Présentation du diagramme de cas d’utilisation

### Diagramme du cas d’utilisation général

Dans cette section, Nous proposons le diagramme général du cas d’utilisation à travers le graphique ci-dessous pour fournir une vue globale de l’élément fonctionnel de notre solution finale :



### Description textuelle des cas d’utilisation

Les descriptions textuelles permettent de fournir une explication détaillée et narrative des interactions entre les acteurs (utilisateurs) et le système informatique. Nous complétons certaines cas d’utilisations par des descriptions textuelles afin de mieux comprendre les fonctionnalités du système.

#### Description textuelle du cas d’utilisation « s’authentifier »

Tableau 2 description Textuelle de l'authentification

|  |  |
| --- | --- |
|  | Sommaire d’identification |
| titre | **S’authentifier** |
| Resumé | Ce cas d’utilisation permet à l’utilisateur de s’authentifier pour bénéficier des services de l’application. |
| Acteurs | Le chargé de la banque (acteur principal) ou le client (acteur principal) |
|  | **Description des scénarios** |
| Préconditions | 1. L’application « e-credit » est installée et fonctionnelle 2. L’utilisateur dispose d’un compte valide |
| Scénario nominal  Scénario alternatif | 1. L’utilisateur accéde ẚ l’ecran d’authentification. 2. L’utilisateur saisit son e-mail et son mot de passe. 3. Le système vérifie les informations saisies par l’utilisateur. 4. Si l’utilisateur est identifié, le système affiche l’interface de l’accueil.   **A1.** Email ou mot de passe non valide   1. Le système affiche un message d’erreur «Votre identifiant ouvotre mot de passe est incorrect » et signale à l’utilisateur de recommencer. 2. L’utilisateur a la possiblité de réessayer. |
| Postcondition | * L’utilisateur est authentifié et accéde à sa page d’accueil. |

#### Description textuelle du cas d’utilisation « demander crédit »

Tableau 3 Description textuelle du cas d’utilisation « Demander crédit »

|  |  |
| --- | --- |
|  | Sommaire d’identification |
| titre | **Demander un credit en ligne** |
| Resumé | Ce cas d’utilisation permet au client de déclencher une crédit en ligne en fournissant les informations requises. |
| Acteurs | le client (acteur principal) |
|  | **Description des scénarios** |
| Préconditions | * Client authentifié, * Service web disponible, * Serveur de base de donnee disponible. |
| Scénario nominal  Scénario alternatif | 1. Le client accéde a l’ecran de demande credit. 2. Le système affiche un formulaire de demande de credit avec les champs necessaires a remplir. 3. Le client saisit les informations requises dans le formulaire notamment le CIN du client, son numéro du compte, les details de son dossier credit (le type de crédit, le montant, l’unité), les garanties proposées (la nature du garantie, le type de garantie, la valeur du garantie, la devise du garantie), le nom de son suivi et la date à laquelle la relation a été etablie avec son suivi, les pièces jointes necessaires et les observations facultatives. 4. Le client valide la demande en cliquant sur le bouton ‘enregistrer’, 5. Le système enregistre la demande de credit et la soumet pour examen par le chargé de la banque.   **A1.** Service web ou serveur de base de donnée indisponible   1. Le système affiche un message d’erreur «une erreur inattendu s’est produite». |
| Postcondition | * Le client attend qu’un administrateur approuve sa demande. |

## Analyse des cas d’utilisation

### Diagrammes de séquence système

#### Diagramme de séquence système « authentification »

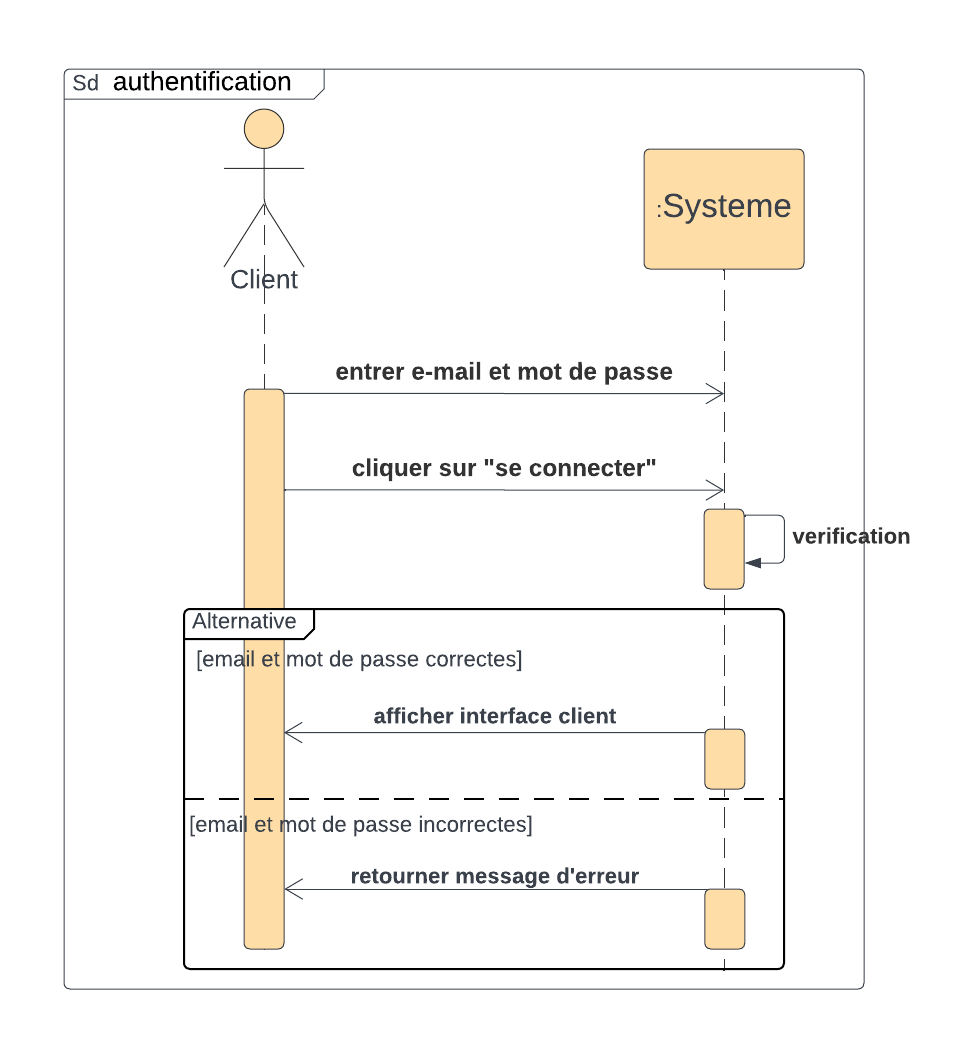
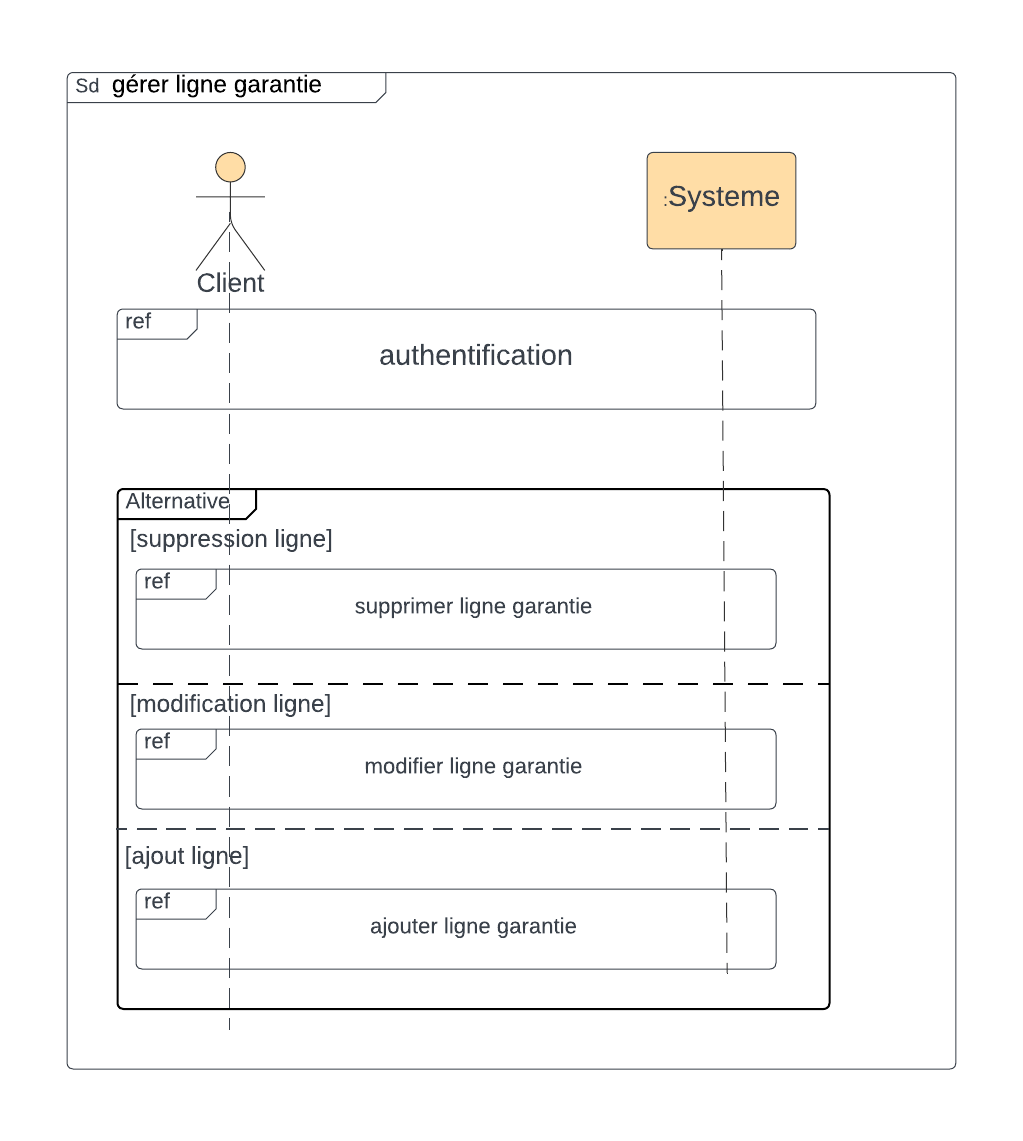
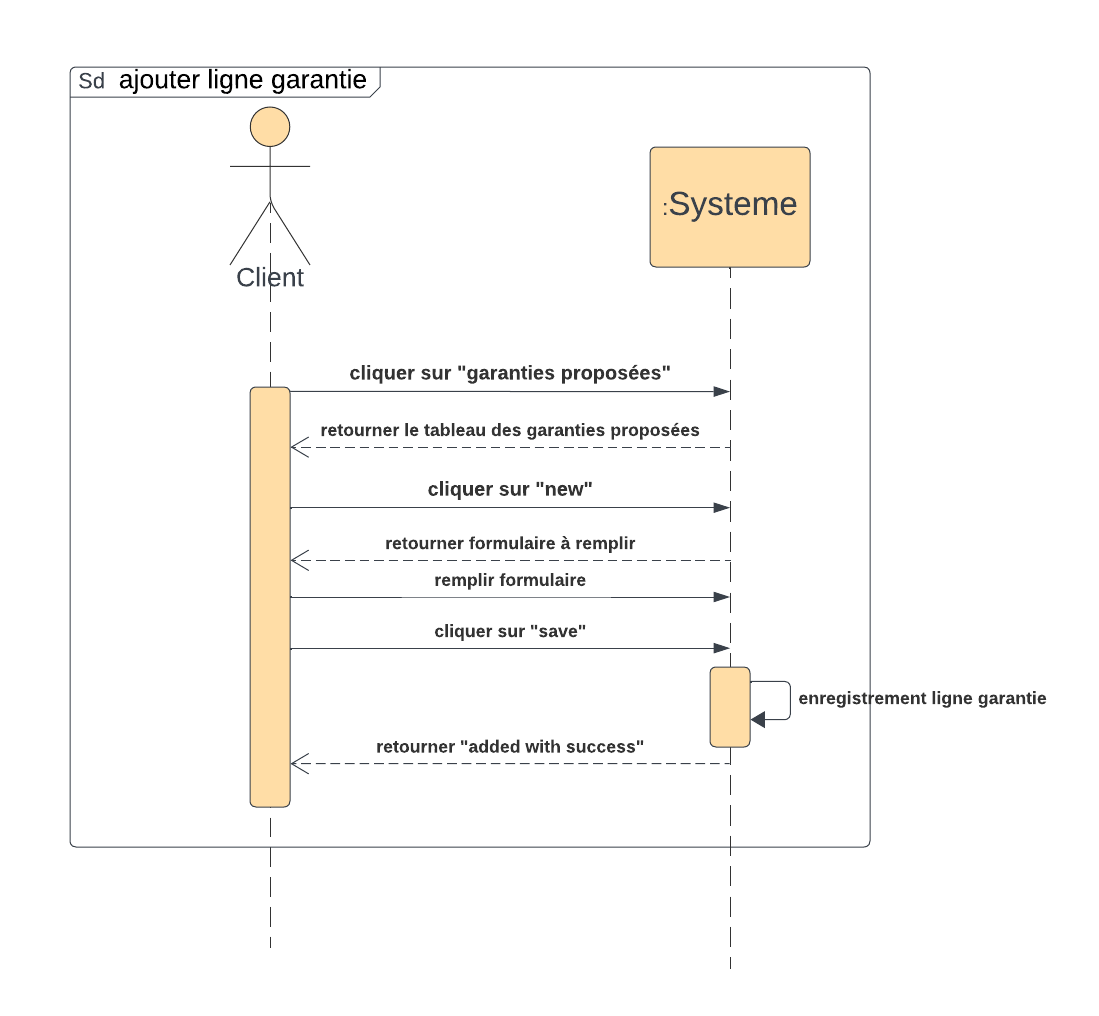


Figure 4 Diagrammes de séquence « authentification »

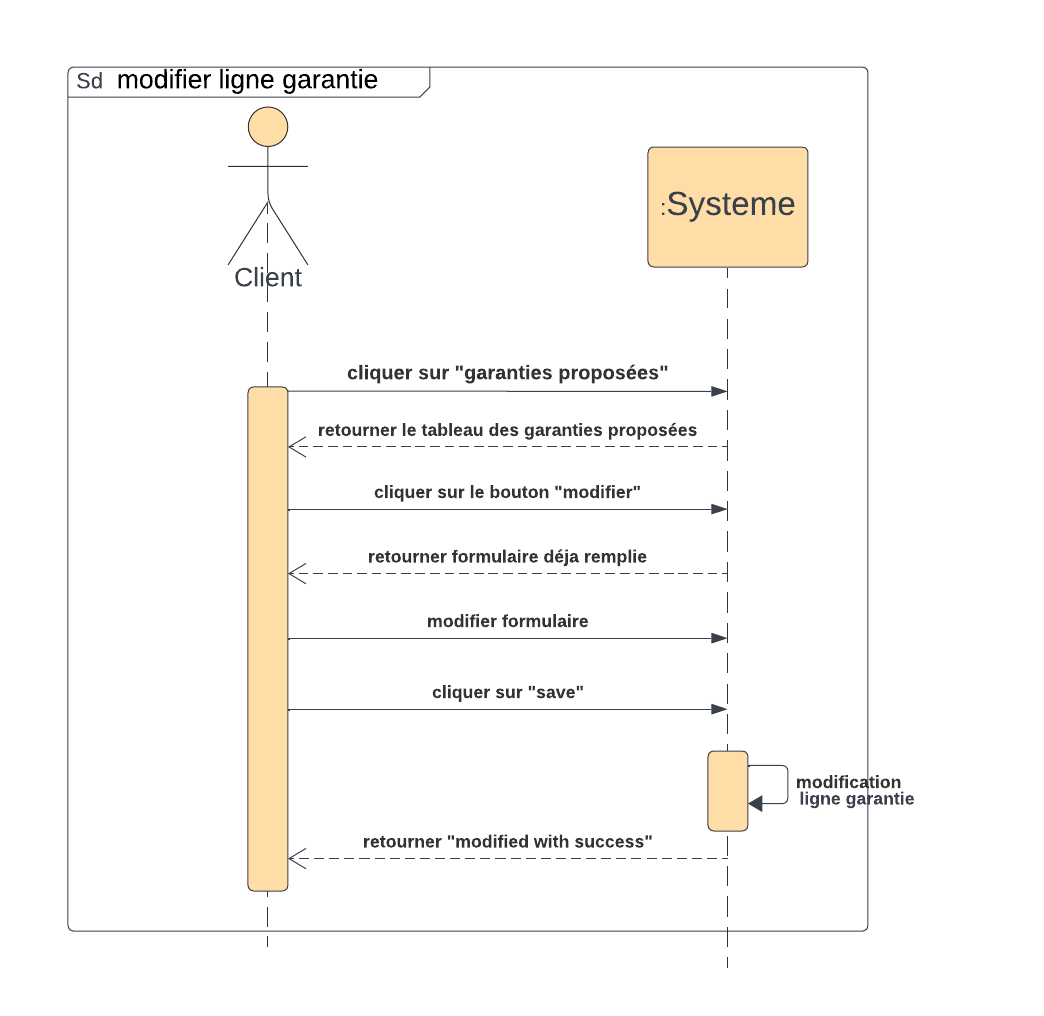
#### Diagramme de séquence système « gérer ligne garantie »



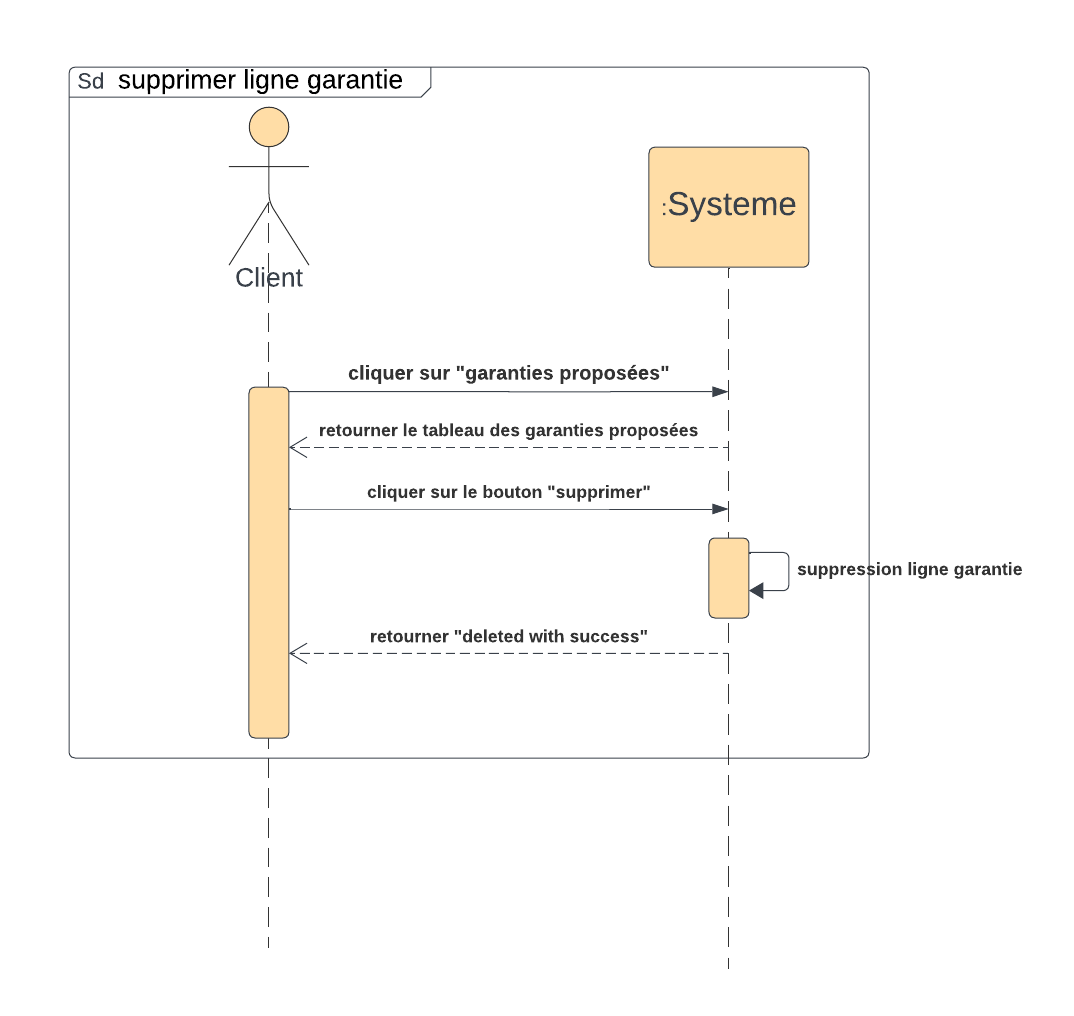
#### Diagramme de séquence système « ajouter ligne garantie »



#### Diagramme de séquence système « modifier ligne garantie »



#### Diagramme de séquence système « supprimer ligne garantie »

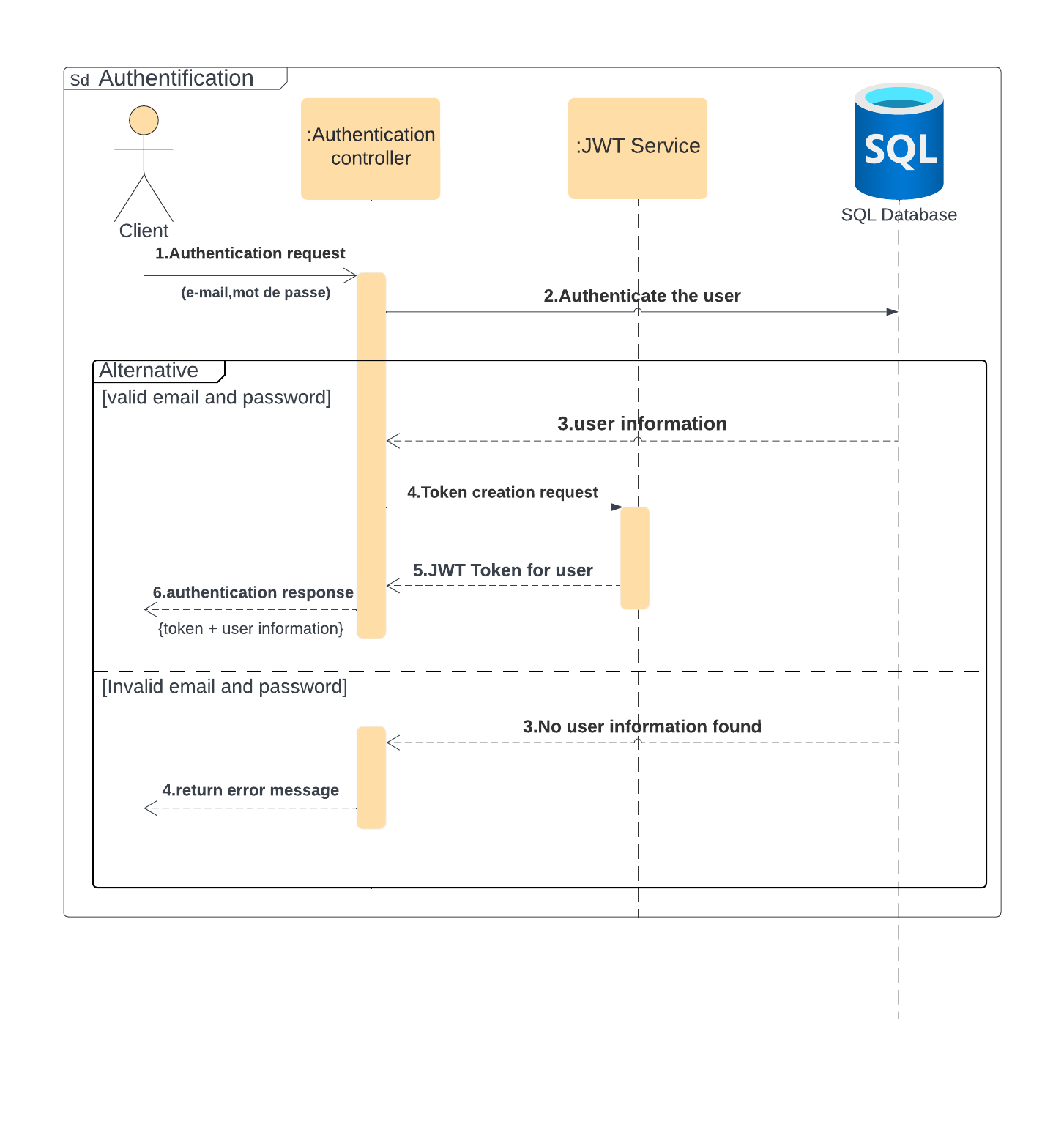


### Diagrammes de séquence objet

Les diagrammes de séquences modélisent l’aspect dynamique du système, ils permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs.

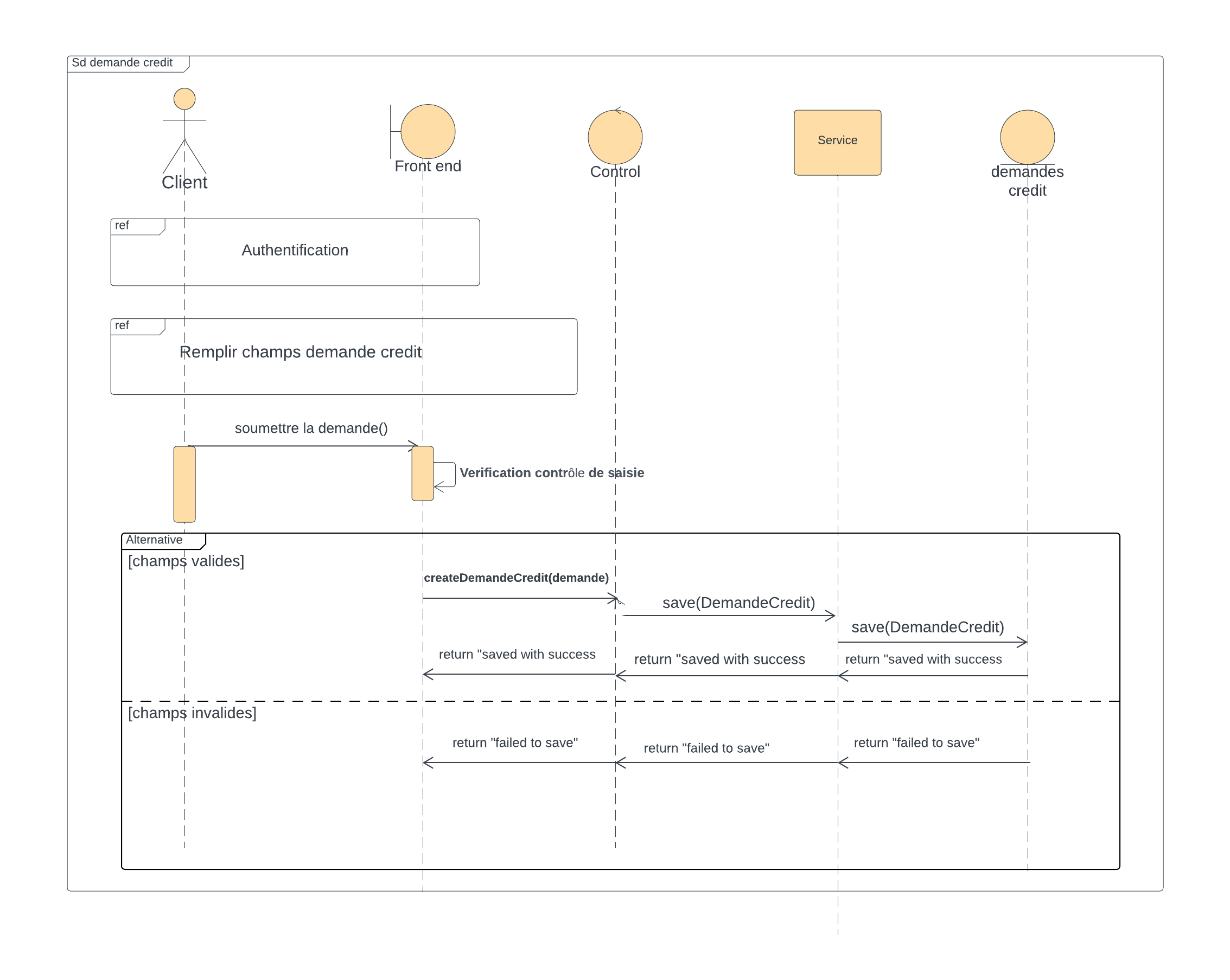
#### Diagramme de séquence objet « Authentification »

|  |
| --- |
| Description detaillé |
| L’utilisateur envoie une requête d’authentificationau serveur en incluant les informations d’identifications(e.mail et mot de passe) pour l’authentification. La requête est reçue par AuthenticationController qui envoie la demande d’identification à un service, et si l’objet retourné est non nul et que le service a validé l’authentification, il demande à un second service (JWT service) de créer notre Token à partir de l’objet authentification, le JWT service retourne ensuite le Token crée comme contenu de la réponse ainsi que les détails de l’utilisateur, y compris ses rôles. |



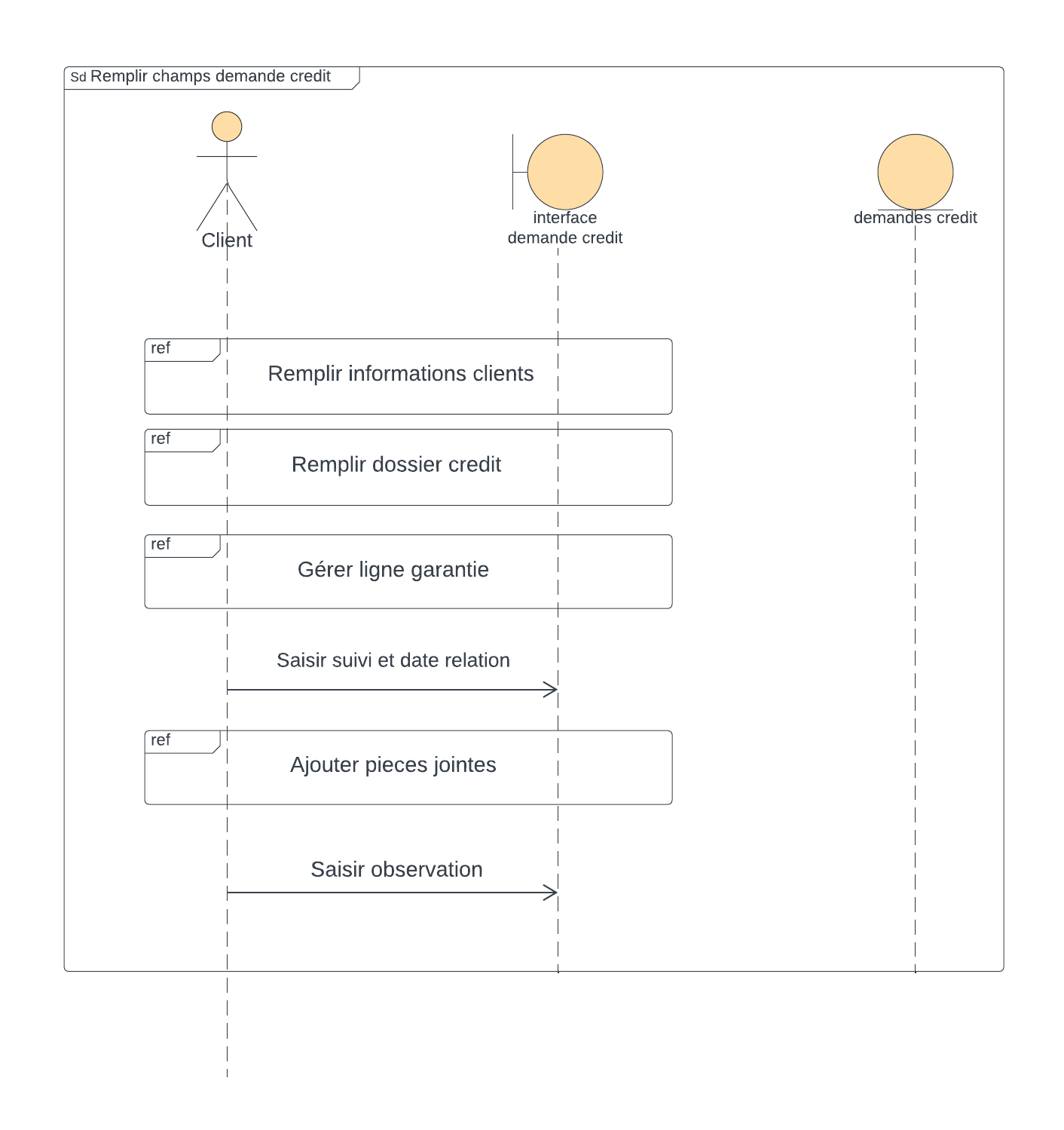
#### Diagramme de séquence objet « Demander crédit »

Le diagramme de séquence de la "demande de crédit" illustre le processus complet de soumission d'une demande de crédit par un client, depuis son authentification jusqu'à l'enregistrement de la demande en passant par différentes étapes :



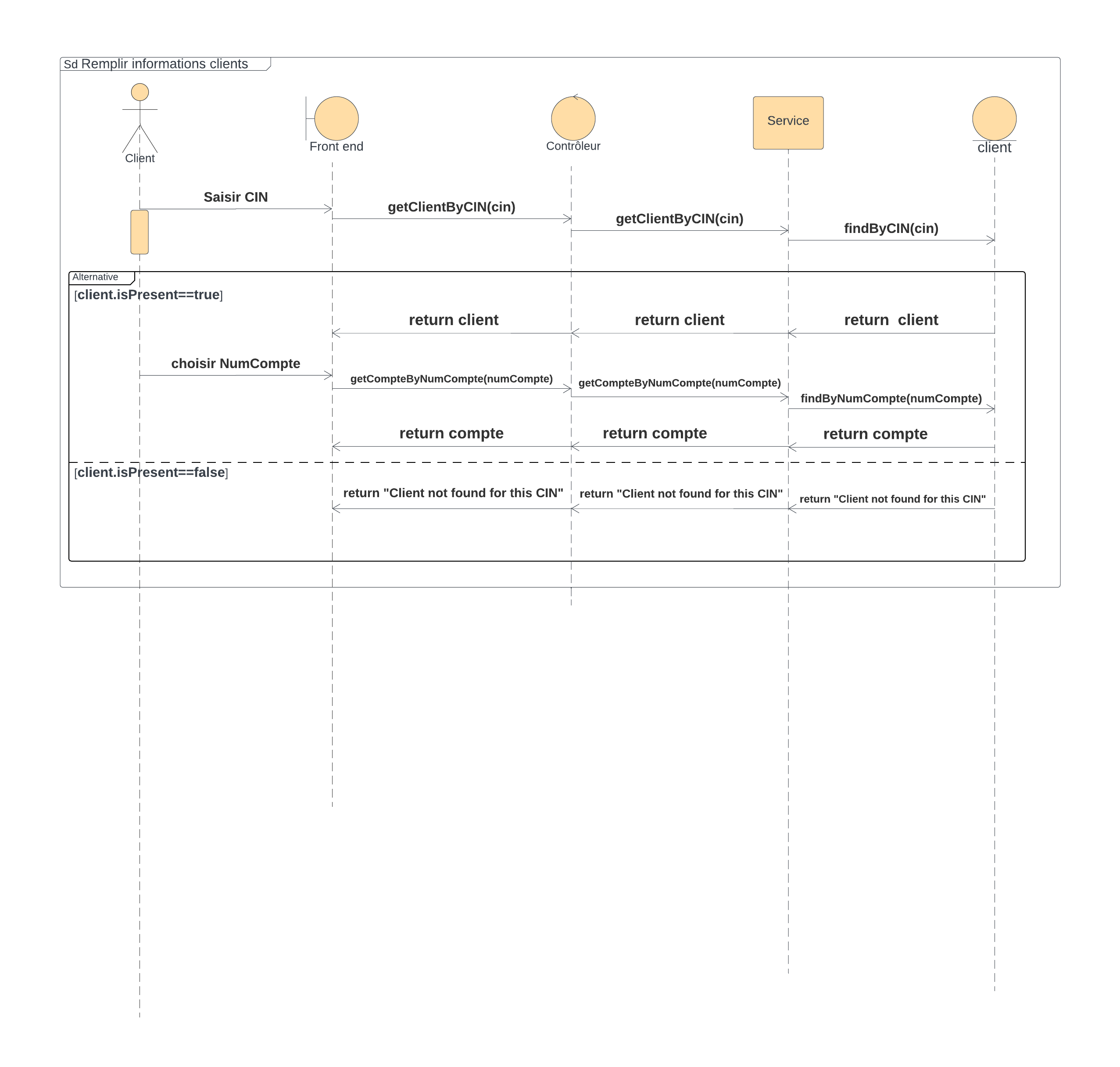
#### Diagramme de séquence objet « Remplir champs demande crédit »

Le processus de demande de crédit implique que le client, via la vue (frontend), renseigne les champs de la demande, fournisse des détails personnels et financiers, propose des garanties éventuelles, joigne des pièces justificatives, et saisisse des observations pour évaluation, avant de soumettre la demande à l'institution financière.



#### Diagramme de séquence objet « Remplir informations client »

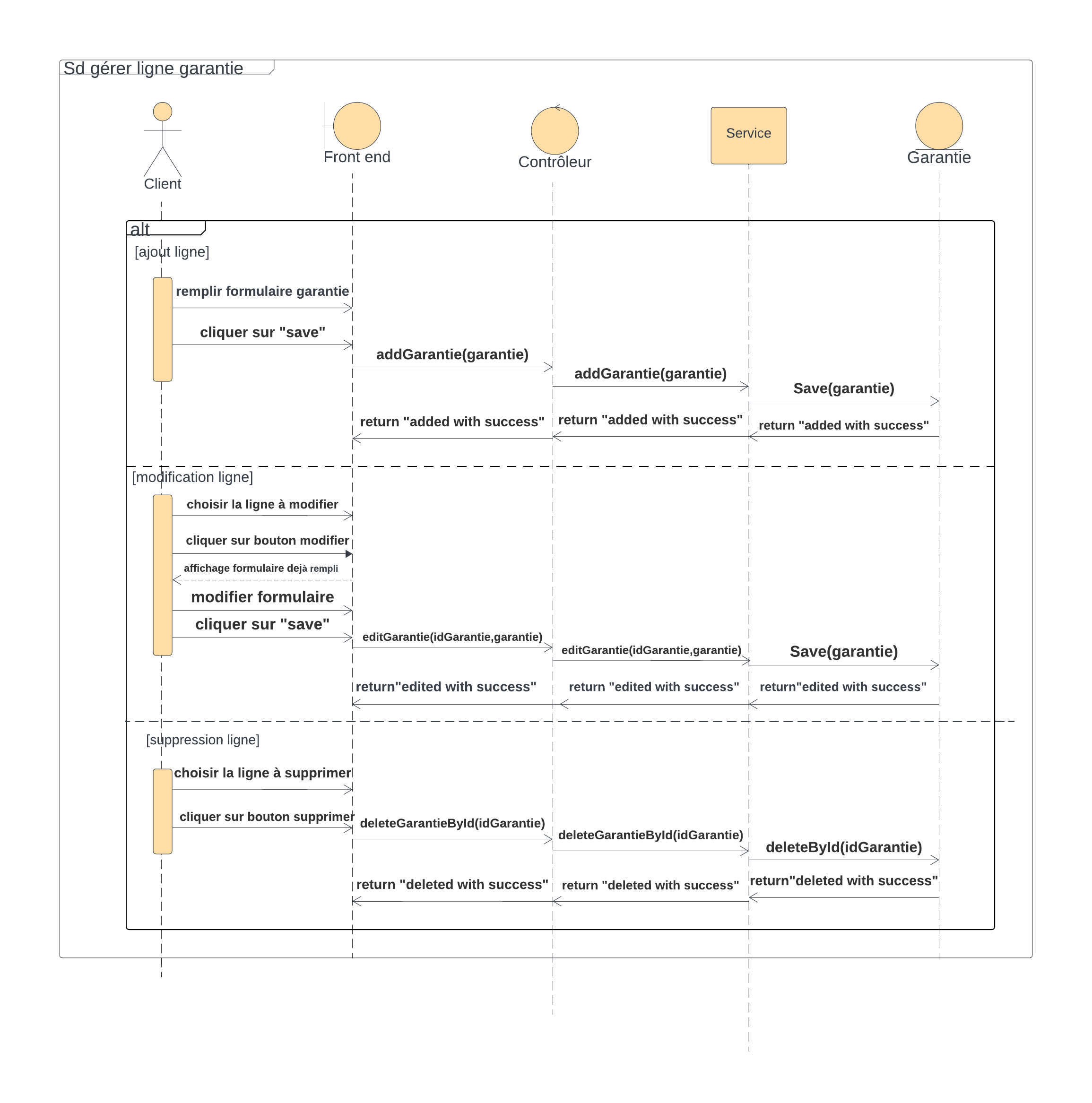
L'utilisateur interagit avec la partie Front end (Angular) en saisissant le CIN. Cette partie envoie une requête HTTP GET au contrôleur Spring Boot avec le CIN saisi. Le contrôleur traite la requête et appelle une méthode du service pour récupérer les données associées au CIN. Le service interagit avec la couche d'accès aux données (Repository) pour récupérer les données stockées dans la base de données. Une fois que les données sont récupérées, le service les retourne au contrôleur. Le contrôleur utilise les données récupérées pour mettre à jour l'interface utilisateur, affichant ainsi les données du client correspondantes.



#### Diagramme de séquence objet « gérer ligne garantie »

Le client remplit le formulaire de garantie via l'interface utilisateur, fournissant les détails requis. Le contrôleur reçoit les données de la vue et les transmet au service de garantie pour traitement. Le service de gestion des garanties traite les requêtes CRUD en fonction des actions du client, assurant la cohérence des données et communique avec le repository pour persister les informations dans la base de données. Le repository agit comme une passerelle entre le service et la base de données, effectuant les opérations de sauvegarde, récupération, mise à jour et suppression des garanties. La base de données stocke les informations sur les garanties proposées par les clients, offrant une persistance et une accessibilité aux données.

Après chaque opération (ajout, édition, suppression) effectuée sur une garantie, la base de données envoie un message de confirmation au service, indiquant le succès de l'opération. Ce message est ensuite propagé par le service vers le contrôleur, et éventuellement jusqu'à l'interface utilisateur, afin d'informer l'utilisateur du résultat de son action.



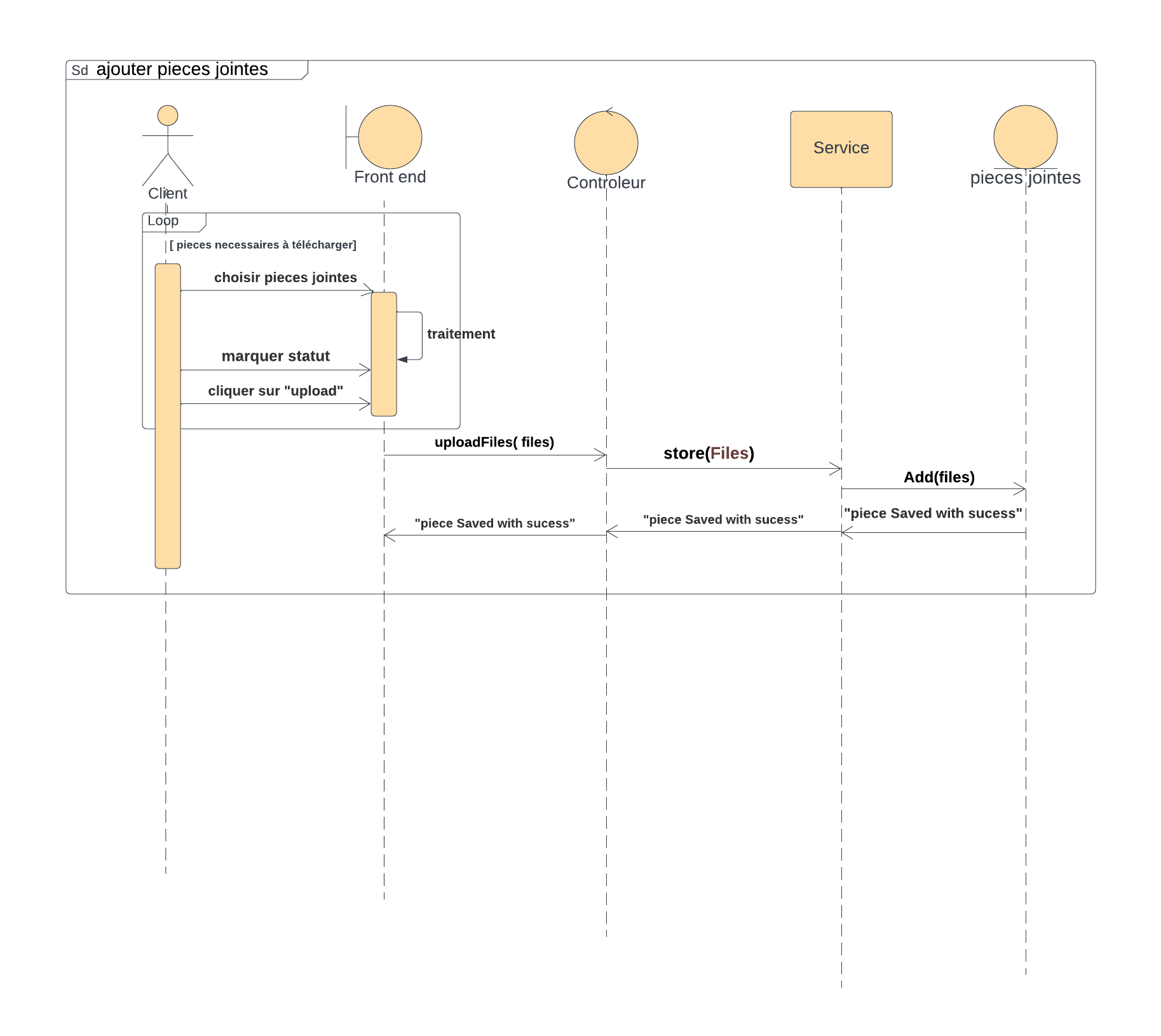
#### Diagramme de séquence objet « ajouter pièces jointes »

Le client sélectionne les fichiers à téléverser via l'interface utilisateur et marque le statut du pièce jointe récemment choisie. Le contrôleur intercepte la demande d'envoi de fichiers et la transmet au service approprié pour le traitement. Le service d'upload de fichiers gère le traitement des fichiers à téléverser, assure la validation des formats et des tailles, et orchestre l'interaction avec le système de stockage ou la base de données.

Les fichiers sont stockés dans la base de données ce qui nécessite une interaction étroite avec le repository. Le repository agit comme une interface entre le service et la base de données, fournissant des méthodes pour sauvegarder les fichiers téléversés.

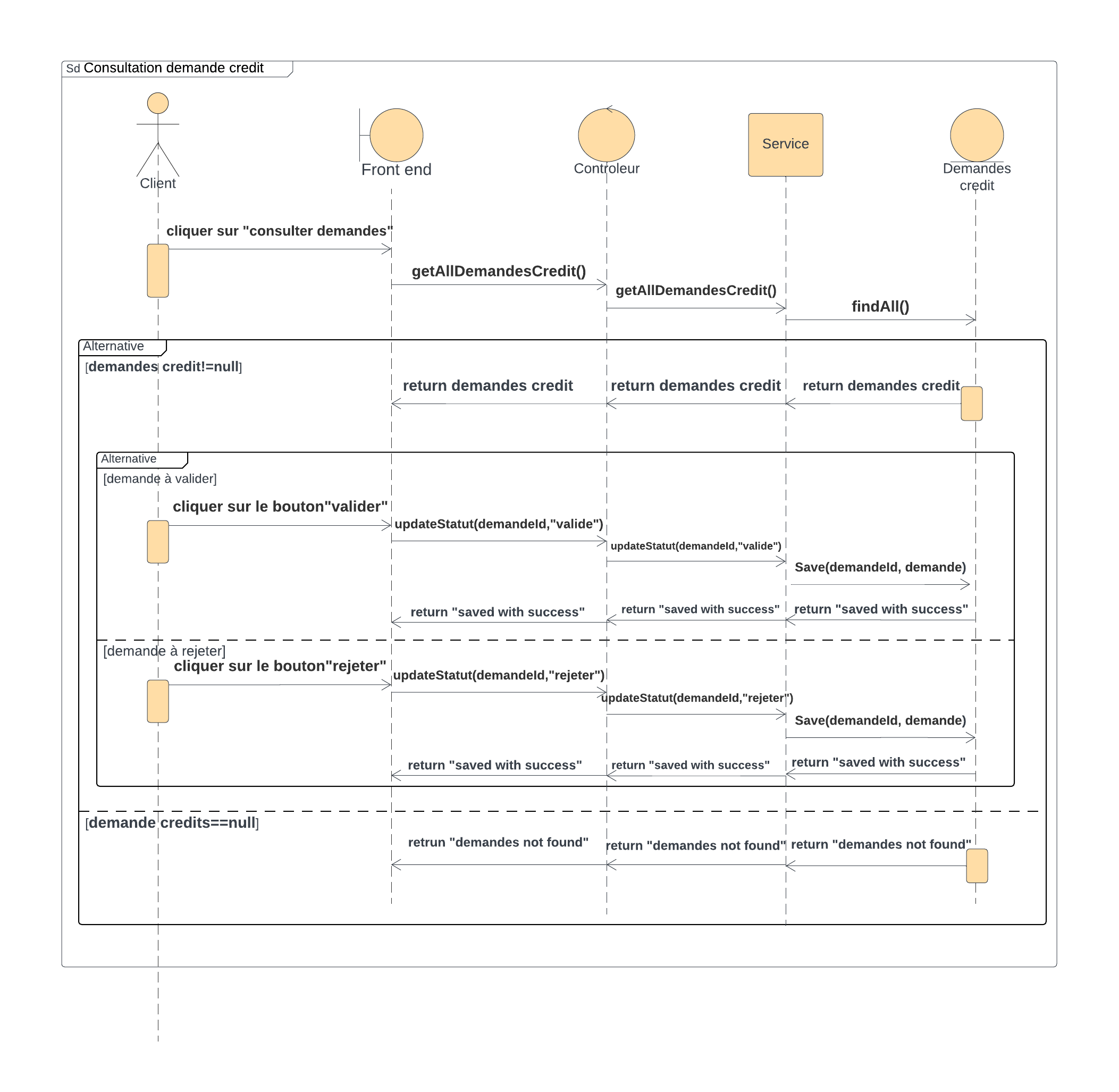
Une fois que les fichiers sont téléversés avec succès, le système renvoie un message de confirmation au service, indiquant que l'opération a été effectuée avec succès.

Le service transmet ce message de confirmation au contrôleur, qui peut ensuite le renvoyer à l'interface utilisateur pour informer l'utilisateur que les fichiers ont été téléversés avec succès.



#### Diagramme de séquence objet « Consultation demandes crédit »

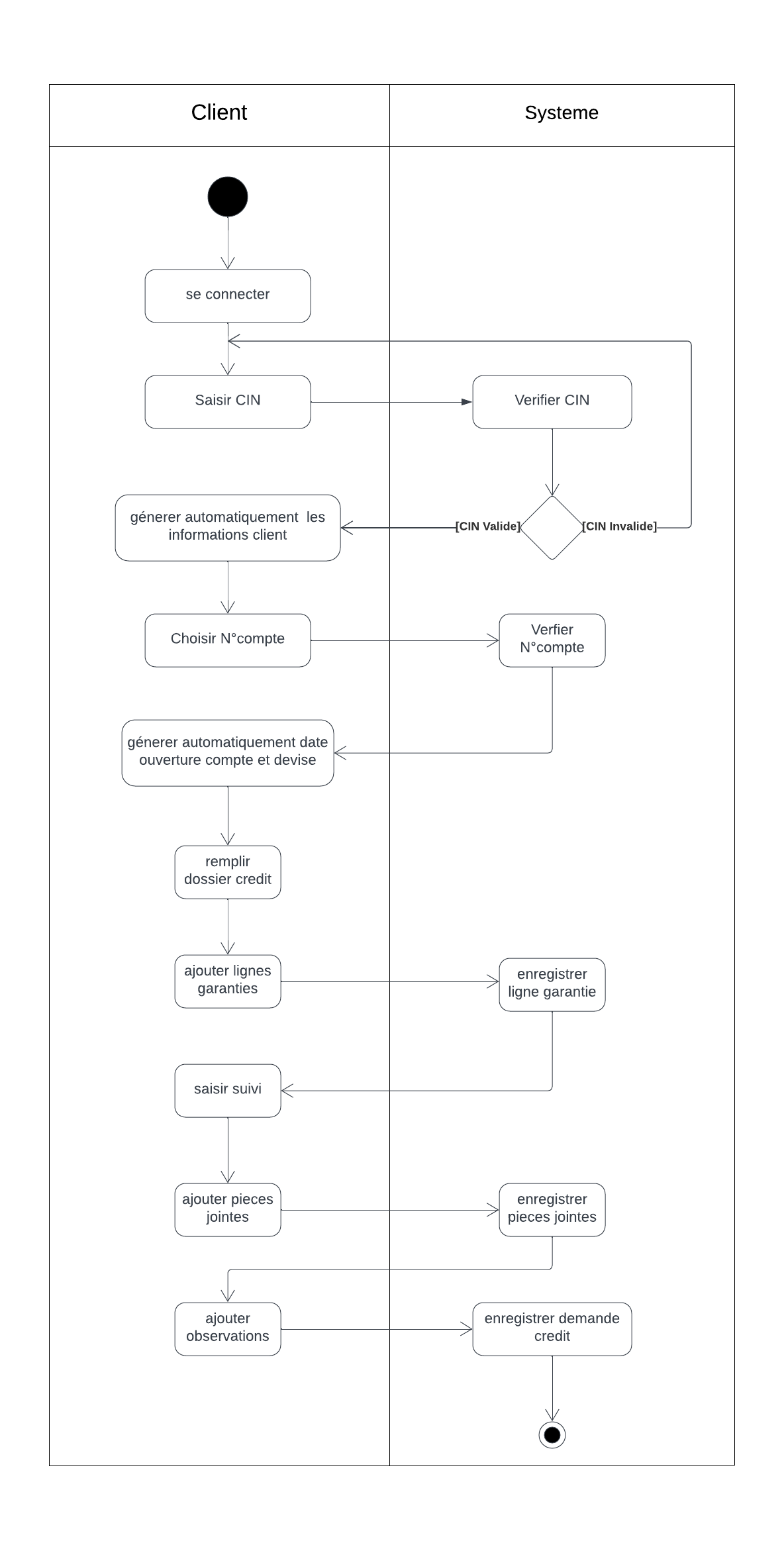
Dans le processus de consultation des demandes de crédit, le contrôleur reçoit la demande d'affichage des demandes de crédit depuis l'interface utilisateur. Il utilise les méthodes appropriées du service de consultation pour récupérer les données nécessaires. Le service, à son tour, fait appel au repository associé pour accéder aux données stockées dans la base de données. Le repository interagit directement avec la base de données pour exécuter les requêtes de lecture et récupérer les demandes de crédit enregistrées. Une fois les données récupérées, elles sont transmises du repository au service, puis au contrôleur, qui les renvoie à l'interface utilisateur pour affichage. Lorsque l’administrateur demande la modification du statut d'une demande de crédit, le contrôleur réagit en appelant les méthodes pertinentes du service. Le service utilise le repository pour accéder aux données dans la base de données. Le repository met à jour le statut de la demande dans la base de données, puis retourne la réponse au service, qui la transmet ensuite au contrôleur. Enfin, le contrôleur renvoie une confirmation à l'utilisateur, assurant ainsi le traitement complet et précis de la modification du statut de la demande de crédit.



### Diagramme d’activité

Les diagrammes d'activité fournissent une représentation visuelle des processus, facilitant la communication et la documentation des flux de travail. Ils permettent de modéliser clairement les étapes, interactions et décisions, favorisant une conception efficace des processus métier. Dans le contexte de la demande de crédit, le choix du diagramme d'activité clarifie la compréhension pour les utilisateurs et les parties prenantes.

Le diagramme d’activité ci-dessous présente le fonctionnement de la demande crédit soumise par le client :



# Chapitre 3 : Modélisation structurelle de l’application

## Diagramme de classes

### Descriptif du diagramme

La classe « client » représente tout utilisateur de l’application « e-crédit » (un utilisateur, un client, un chargé de banque). Elle possède un identifiant (idClient) comme clé primaire et d’autres attributs déclarées « private », ainsi que d’autres méthodes déclarées « public ».

La classe « compte » représente le compte du client. Le compte est identifié par idCompte comme clé primaire et présente des attributs déclarés « private » et d’autres méthodes déclarées « public ».

Le client peut avoir zéro ou plusieurs comptes mais le compte n’appartient qu’à un seul client.

La classe « demande crédit » nous renseigne sur les composants principaux de la demande de crédit soumise par un client.

La demande crédit est soumise par un seul compte du client mais un compte peut soumettre plusieurs demandes de crédit ou ne soumet aucune demande.

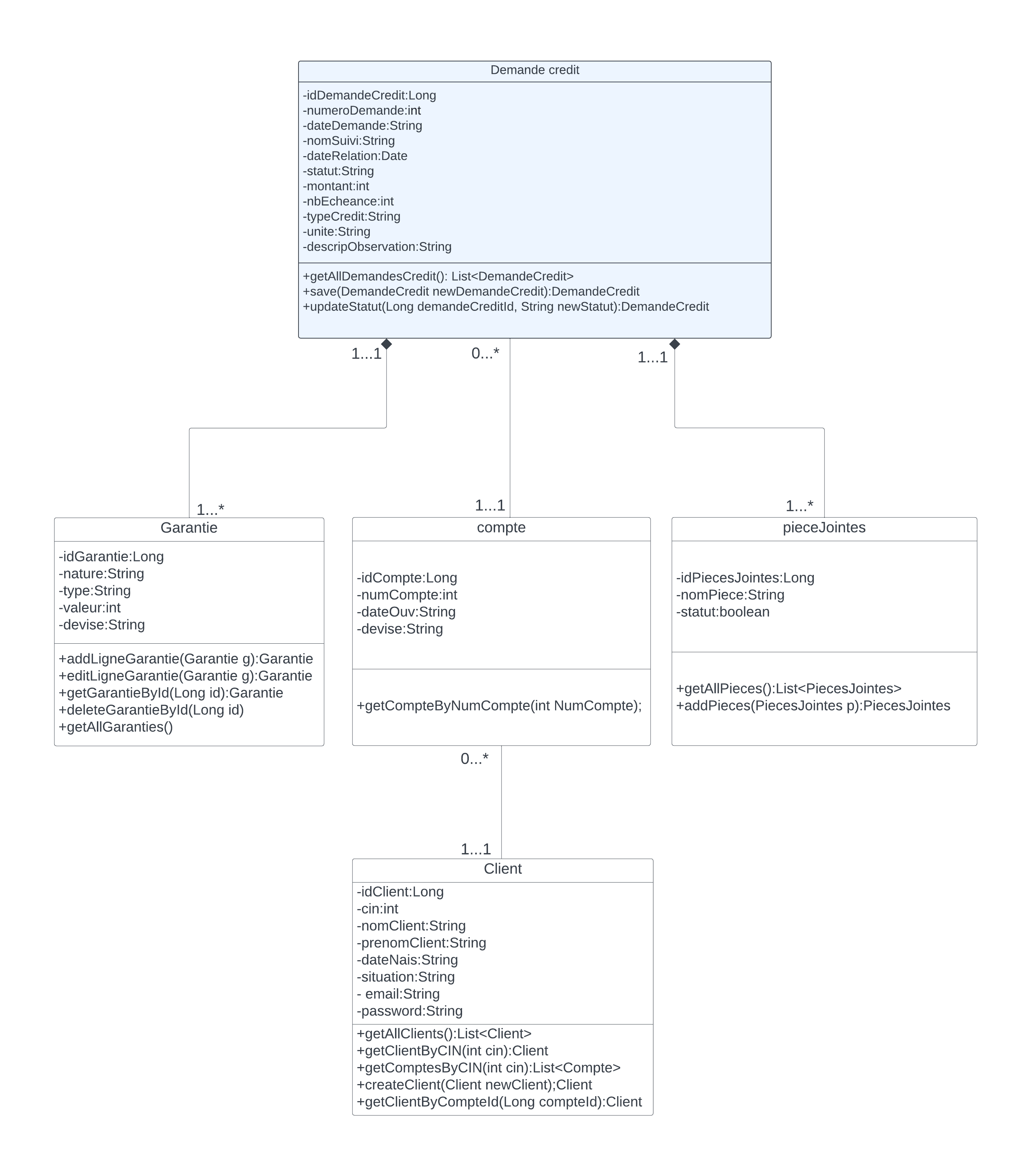
La classe « garantie » représente la garantie proposée par le client. Elle est identifiée par idGarantie comme clé primaire et présente des attributs déclarés « private » et d’autres méthodes déclarées « public ».

La demande crédit possède un ou plusieurs garanties alors que la ligne de garantie appartient qu’à une seule demande.

La classe « pièce jointe » représente la pièce jointe nécessaire à importer par le client. Elle est identifiée par idPièce comme clé primaire et présente des attributs déclarés « private » et d’autres méthodes déclarées « public ».

La demande crédit possède un ou plusieurs pièces alors que la pièce jointe appartient qu’à une seule demande.

### Représentation du diagramme



## Modélisation de la base de donnée

### Modèle relationnel

Demande crédit (**idDemandeCredit**, numeroDemande, dateDemande, nomSuivi, dateRelation, statut, montant, nbEcheance, typeCredit, unite, descripObservation, #**idCompte**)

Garantie (**idGarantie,** nature**,** type**,** valeur, devise, #**idDemandeCredit**)

Pièce Jointe (**idPiecesJointes,** nom Pièce, statut, #**idDemandeCredit**)

Compte (**IdCompte,** numCompte, dateOuv, devise, #**idClient**)

Client (**idClient,** cin**,** nomClient, prenomClient, dateNais, situation, email, password)

# Chapitre 4 : Réalisation

Dans ce chapitre, nous façonnons l'application "e-crédit" en dévoilant son écosystème de développement et ses interfaces graphiques principales. Ces éléments convergent pour donner forme à notre vision, transformant les concepts en une expérience utilisateur concrète et immersive.

## Environnement de développement

### Environnement matérielle

Tableau 5 environnement matérielle

|  |  |
| --- | --- |
| Description matérielle | |
| Marque | HP |
| Processeur | 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 2.70 GHz |
| Disque dur | 475 GO |
| Mémoire vive | 16,0 Go |
| Carte graphique | Intel(R) Iris(R) Xe Graphics |

### Environnement logiciel

Tableau 6 environnement logiciel

|  |  |
| --- | --- |
| Description logicielle | |
| Système d’exploitation | Windows 11 |
| Serveur base de donnée | MySQL |
| Outils de modélisation | Lucidchart |
| Outils de développement | Visual Studio code, Spring Tool Suite 4 |
| Langages | Java, HTML, CSS, typeScript |
| Technologies | Angular 17, Spring boot 2, Spring Security |
| Bibliothèque | PrimeNG |
| Outils de Test | Postman |

### Présentation de l’environnement logiciel

#### MySQL

##### Définition

MySQL est un [système de gestion de base de données (ou SGBD)](https://www.ionos.fr/digitalguide/hebergement/aspects-techniques/systeme-de-gestion-de-base-de-donnees-sgbd/) relationnelles basé sur SQL. Ce **logiciel open source** est aujourd’hui géré par Oracle, l’entreprise qui a également développé le langage de programmation Java. MySQL stocke, gère et affiche des données dans des tableaux en fonctionnant comme un **système client-serveur**. La figure 27 illustre le logo de MySQL :



Figure 30 LOGO MySQL

#### Angular

##### Définition

Angular est un Framework de développement front-end open-source, basé sur TypeScript, et développé par Google. Il est utilisé pour créer des applications web dynamiques et réactives. Angular est conçu pour faciliter la création d'applications monopage (SPA - Single Page Applications) et offre de nombreuses fonctionnalités pour le développement d'applications web modernes. La figure 28 présente le logo de Angular :



Figure 31 logo Angular

##### Utilisation des fonctionnalités clés d’Angular

Lors du développement de l'application « e-crédit », nous exploitons plusieurs fonctionnalités clés offertes par le Framework. Parmi ces fonctionnalités, nous pouvons citer :

* Architecture basée sur les composants : Nous adoptons l'approche basée sur des composants pour structurer l'interface utilisateur de l'application. Chaque composant Angular représente une unité autonome et réutilisable de l'interface utilisateur, ce qui facilite la gestion et la maintenance du code.
* Binding des donnés bidirectionnel : L'utilisation du système de binding de données bidirectionnel d'Angular nous permet de synchroniser automatiquement les données entre la vue et le modèle. Cela nous facilite la mise à jour dynamique de l'interface utilisateur en réponse aux changements de données.
* Système de routage : Le système de routage d'Angular est intégré pour gérer la navigation entre les différentes vues et composants de l'application. Cela permet de créer une expérience utilisateur fluide et intuitive, avec des transitions transparentes entre les différentes parties de l'application.
* Utilisation des formulaires réactifs : Les formulaires réactifs d'Angular, basés sur les observables, sont employés pour la gestion dynamique des formulaires dans l'application. Cette approche nous permet de créer des formulaires réactifs et interactifs, offrant une meilleure expérience utilisateur.

1. **Avantages d’angular**

Des avantages qu’Angular nous offre comme des développeurs contribuent à faciliter le processus de développement et à optimiser l’efficacité du travail réalisé de notre application :

* Productivité accrue : Grâce à des fonctionnalités telles que le binding de données, Angular nous aide considérablement à améliorer notre productivité en tant que développeurs. La capacité à lier dynamiquement les données entre la vue et le modèle simplifie la gestion des données et réduit le temps de développement nécessaire pour créer des fonctionnalités interactives.
* Modularité : Angular encourage fortement la modularité en permettant aux développeurs de diviser leur application en modules réutilisables. Cette approche facilite la gestion de projets complexes en permettant une séparation claire des fonctionnalités et en favorisant la réutilisation du code.
* Performances : Angular est conçu pour offrir des performances élevées, notamment grâce à des techniques telles que le lazy loading comme dans notre cas. Ces fonctionnalités ont permis d'optimiser les performances de l'application et d'offrir une expérience utilisateur fluide et réactive.
* Communauté active : Angular bénéficie d'une large communauté de développeurs et d'une documentation exhaustive, ce qui facilite l'apprentissage et le développement d'applications. La possibilité d'accéder à des ressources de qualité, ainsi que de partager des connaissances et des bonnes pratiques avec d'autres membres de la communauté, a été un atout précieux pour résoudre les problèmes et relever les défis rencontrés lors du développement.

#### Spring boot

##### Définition

Spring boot est un framework open-source moderne et puissant pour le développement d’applications java. Il est conçu pour simplifier le processus de création, de configuration et

de déploiement d’applications java, en mettant l’accent sur la facilité d’utilisation, la productivité et la performance.



Figure 32 logo spring boot

1. **Avantages**

* Facilité de démarrage : Spring boot offre un système de configuration automatique qui permet de configurer rapidement et facilement les composants de l’application en fonction des dépendances présentes dans le projet.
* Embarquement de serveur : Spring Boot permet d'embarquer un serveur d'application (comme Tomcat ce qui est utilisé dans notre cas, Jetty ou Undertow) directement dans l'application, ce qui élimine la nécessité de déployer l'application sur un serveur externe.

1. **Architecture**

L'architecture d'une application Spring Boot suit généralement le modèle MVC (Modèle-Vue-Contrôleur), mais elle peut également inclure d'autres composants pour la gestion des données et la logique métier :

* Les entités : représentent les objets métier de l'application. Ce sont des classes Java qui correspondent généralement à des tables dans la base de données relationnelle.
* Les repositories : offrent une manière simple et efficace de travailler avec les bases de données en fournissant des méthodes prédéfinies pour effectuer des opérations CRUD sur les entités de base de données. Les repositories utilisent généralement des interfaces qui étendent JpaRepository ou CrudRepository pour bénéficier des fonctionnalités de base de Spring Data JPA.
* Les services : Le service layer contient la logique métier de l'application. Il encapsule les opérations métier complexes qui ne devraient pas être directement implémentées dans les contrôleurs.
* Les contrôleurs : Les contrôleurs sont des composants responsables de la gestion des requêtes HTTP et des réponses. Ils reçoivent les requêtes HTTP des clients, invoquent les méthodes appropriées dans le service layer et renvoient les réponses aux clients.
* Les vues : Dans le modèle MVC, les vues représentent la présentation des données à l'utilisateur. Cependant, dans une architecture Spring Boot typique, les vues sont souvent des templates HTML ou des frameworks front-end comme Angular qui est notre cas, React, ou Vue.js.
* Configuration : Spring Boot utilise souvent des fichiers de configuration, tels que ‘application.properties’, pour définir les paramètres de configuration de l'application.

La figure 32 offre un aperçu détaillé de la dynamique d'interaction entre les classes et les interfaces composant le cœur du back-end de Spring Boot, révélant ainsi son architecture interne:

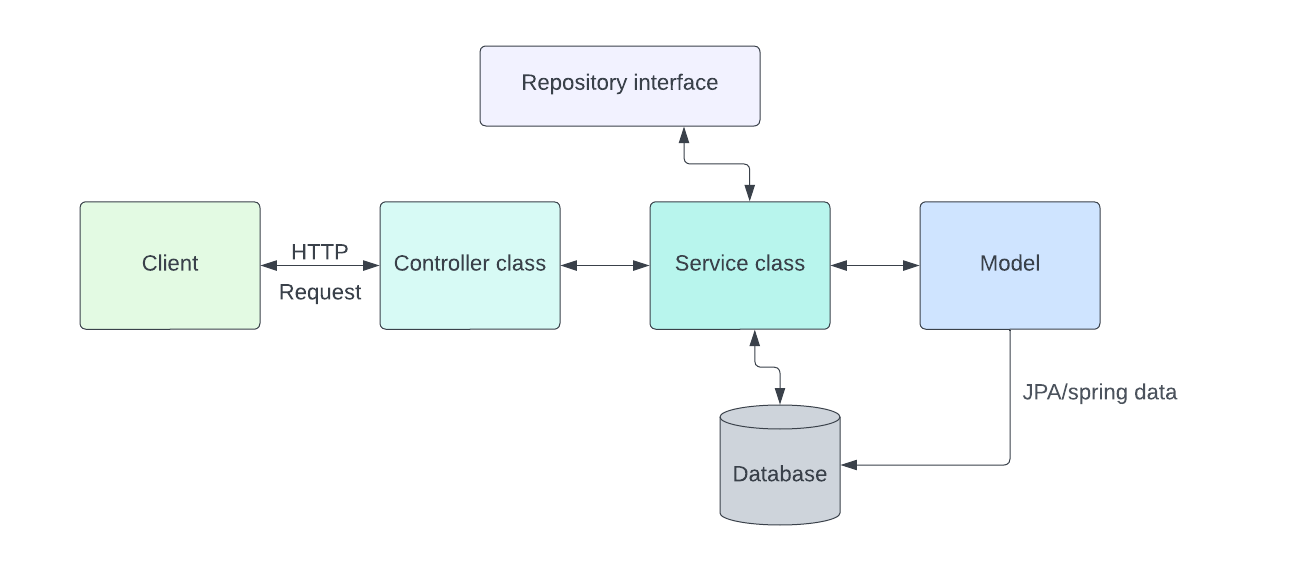


Figure 34 architecture interne de Spring boot

La figure 33 illustre l’interaction entre la partie front-end et la partie back-end :

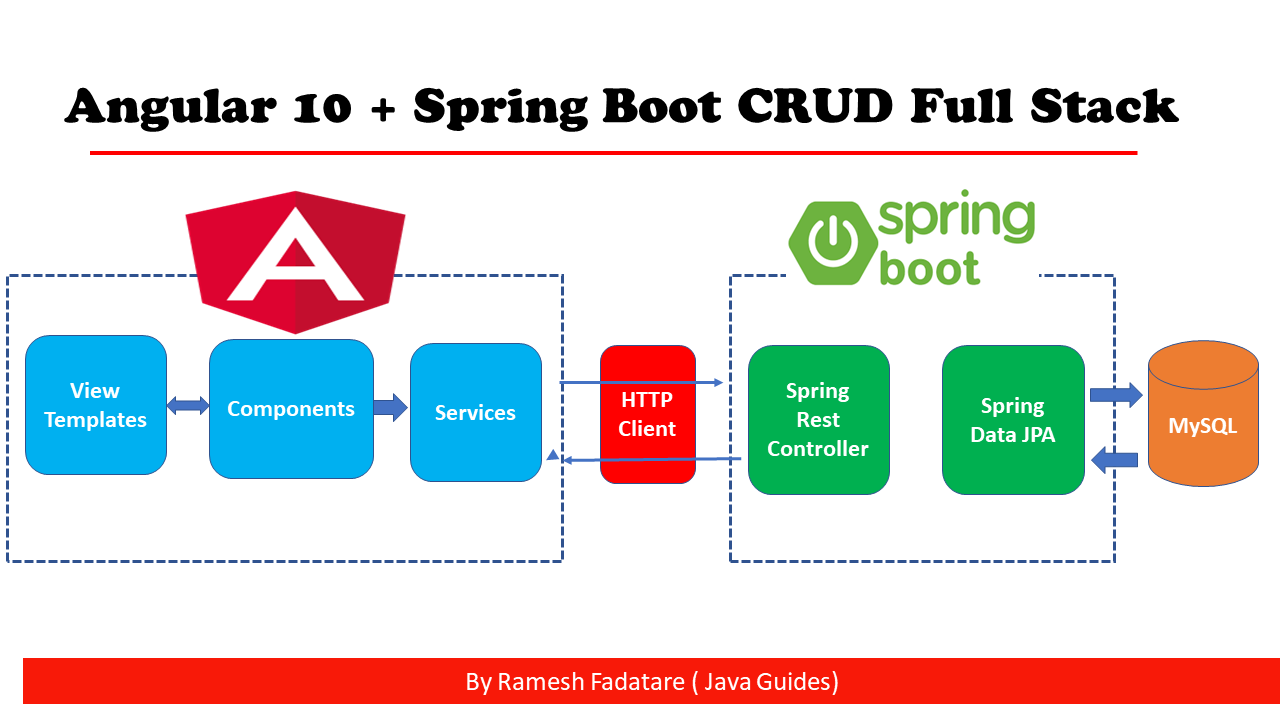


Figure 35 interaction entre Angular et Spring boot

#### Spring security

1. **Definition**

Spring Security est un Framework puissant et largement utilisé dans le développement d’applications java pour la sécurisation des applications web et des services Rest. Il fournit des fonctionnalités robustes pour gérer l’authentification, l’autorisation, la protection contre les attaques….

* Au niveau de l’authentification : Spring Security fournit des mecanismes d’authentification flexibles, notamment l’authentification par formulaire, l’authentification basé sur les jetons (Tokens) tels que Json web Token qui est utilisé dans notre cas. Il prend en charge l’authentification via diverses sources de données telles que les bases de données…..
* Au niveau de l’autorisation : Spring Security permet de définir des règles d’autorisation au niveau des méthodes, des URL et des objets métiers.

La figure 34 présente le logo de Spring Security :



Figure 36 logo spring security

**C’est quoi le Json web Token ?**

Un JSON Web Token (jwt) est un access token (jeton d’accès) qui permet un échange sécurisé de donnée entre deux parties. Il contient **toutes les informations importantes** sur une entité, ce qui rend la consultation d’une base de données superflue et la session n’a pas besoin d’être stockée sur le serveur (stateless session).

**À quoi ressemble la structure d’un JSON Web Token ?**

Un JWT signé se compose de trois parties codées en base64 et séparées par un point :

* L’en-tête (header) : est en général composé de deux parties et fournit des informations essentielles sur le token. Il contient le type de token et l’algorithme de signature et/ou de chiffrement utilisé
* La charge utile (Payload) : La charge utile du JSON Web Token est la partie qui contient les informations qui doivent être transmises à l’application. C’est là que sont définis certains standards qui déterminent quelles données doivent être transmises.
* La signature : La signature d’un JSON Web Token est créée grâce au codage base64 de l’en-tête et de la charge utile et la méthode de signature/cryptage spécifiée. La structure est définie par la JSON Web Signature (JWS), une norme standardisée selon le [RFC 7515](https://tools.ietf.org/html/rfc7515). Pour que la signature fonctionne, il est **nécessaire d’utiliser une clé secrète** connue uniquement de l’application source. Cette signature vérifie d’une part que le message ne sera pas modifié pendant le transfert. D’autre part, dans le cas d’un jeton signé avec une clé privée, il authentifie également l’expéditeur du JWT.

La figure ci-dessous illustre le décodage du Token de notre application « e-crédit » :

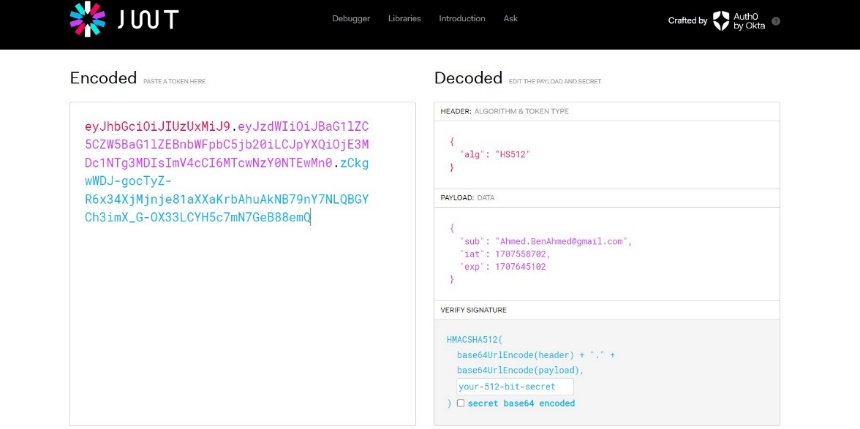


Figure 37 Token décodé de "e-crédit"

## Création du JWT à partir d’une Authentication ?

Lorsqu’un utilisateur se connecte à l’application sur une page de login, Spring Security utilise les informations renseignées pour créer un objet de type *Authentication.* Cet objet :

* Est stocké dans le ***Security Context*** jusqu’à la fin de la requête.
* Est utilisé pour générer un ***JWT***

Notre JWT est construit le nom de l’utilisateur comme sujet, la date d’émission et la date d’expiration. Le JWT est signé avec l’algorithme HS512 et une clé secrète, qui est récupéré à partir de la configuration de l’application.

Le **JWT** est alors renvoyé avec la réponse à la requête de connexion.

**Envoi du JWT avec chaque requête ?**

Une fois le ***JWT***généré et récupéré, il faut le transmettre à Spring Security avec chaque nouvelle requête, pour prouver l’identité de l’utilisateur. Typiquement, le ***JWT*** est adjoint aux requêtes dans un header, qui par convention est nommé “*Authorization*”. Toujours par convention, sa valeur contient la chaine “Bearer ” à laquelle est concaténé le ***JWT*** crypté :

https://miro.medium.com/v2/resize:fit:493/1*7RAsHadZN7NSSBL-xQiw9w.png

L’ensemble connexion + création de **JWT**peut donc être résumé dans le schéma suivant :

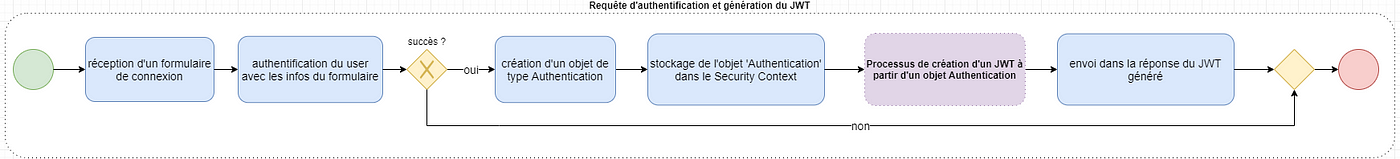
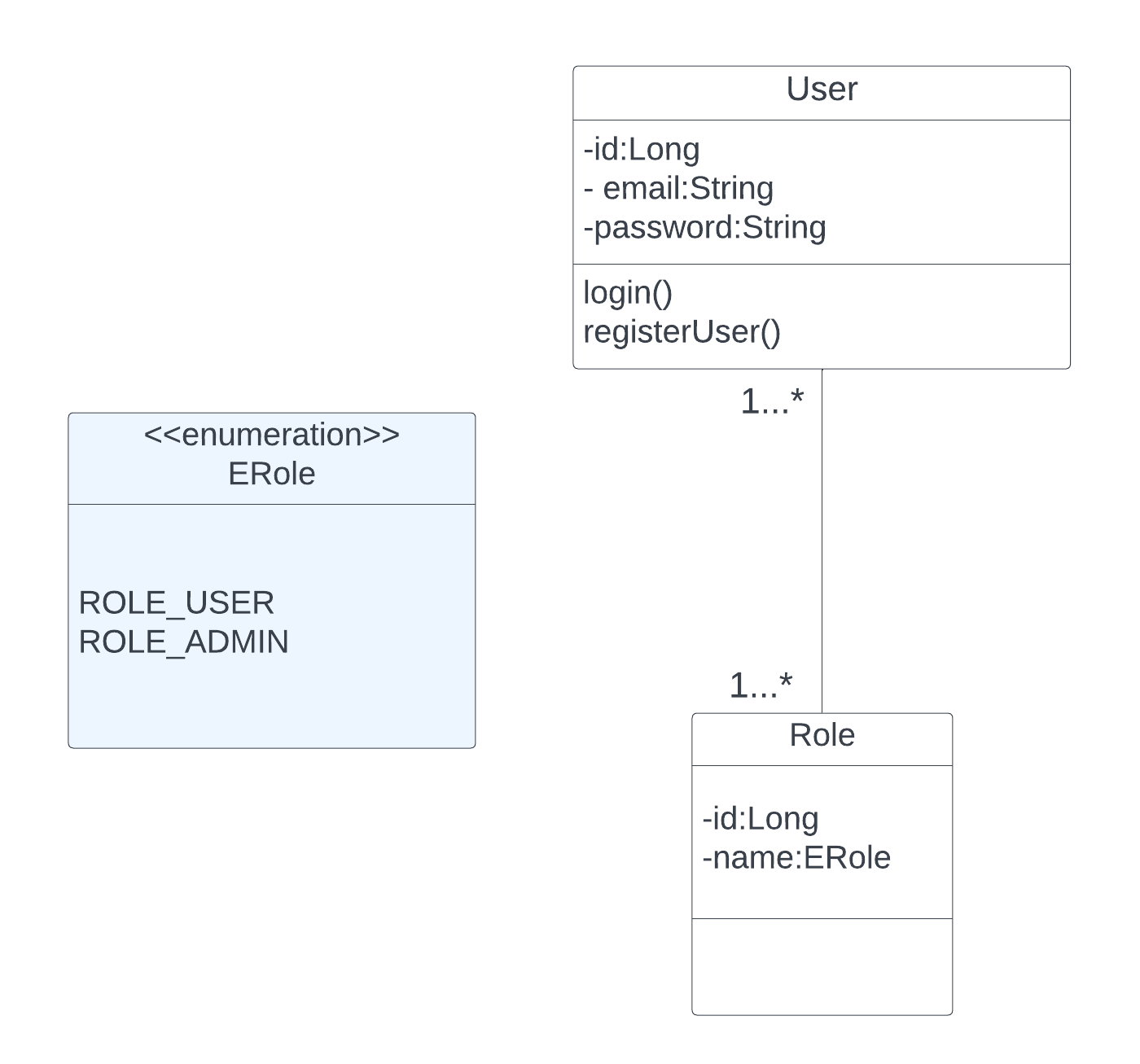


Figure 26 requête d'authentification et génération d'un jwt

1. **Diagramme de classes « gérer les rôles »**

Pour modéliser Spring Security dans une application, il est souvent nécessaire d'introduire une structure de gestion des utilisateurs et des rôles. Cela implique généralement la création de plusieurs classes, notamment une classe User et une classe Rôle.

Nous présentons le digramme de classe de l’authentification en utilisant spring security :



La **classe User** contient des informations telles que l’identifiant de l’utilisateur, le mot de passe et l'email. Ces informations sont essentielles pour l'authentification et l'identification des

utilisateurs dans le système. Le mot de passe est souvent stocké sous une forme sécurisée, comme un hash, pour des raisons de sécurité.

D'autre part, **la classe Rôle** est conçue pour définir les différents rôles attribués aux utilisateurs. Elle présente des attributs tel qu’un identifiant et un nom, où le nom est défini sous forme d'énumération pour assurer la cohérence et la facilité de gestion. Dans cet exemple, nous avons deux rôles principaux : "user" et "admin".

Spring Security s'intègre harmonieusement avec cette structure en permettant la configuration de l'authentification et de l'autorisation basées sur les rôles des utilisateurs.

#### Java

1. **Définition**

Java est un langage de programmation de haut niveau, orienté objet et multiplateforme, largement utilisé dans le développement logiciel. Il a été créé par Sun Microsystems (maintenant acquis par Oracle Corporation) et a été introduit pour la première fois en 1995. Depuis lors, Java est devenu l'un des langages de programmation les plus populaires au monde, largement utilisé dans une grande variété d'applications, des applications mobiles aux applications d'entreprise en passant par le développement web.

La figure ci-dessous présente le logo de java :



Figure 38 logo java

#### PrimeNG

1. **Définition**

PrimeNG est une bibliothèque de composants UI (Interface Utilisateur) pour Angular. Elle fournit une vaste gamme de composants prêts à l'emploi pour la construction d'interfaces utilisateur riches et interactives dans les applications Angular.Nous présentons le logo de Prime NG á partir de la figure 39 :

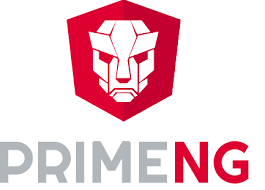


Figure 39 logo prime ng

1. **Fonctionnalités PrimeNG**

Prime NG est la bibliothèque que nous choisissons pour le développement de notre interface utilisateur dans notre application Angular en intégrant et exploitant les fonctionnalités clés de Prime NG :

* Composants riches : PrimeNG offre une variété de composants UI prêts à l'emploi, tels que des boutons, des tables de données, des listes déroulantes, des calendriers, des onglets, des menus, des arbres, des panneaux, des dialogues, et bien d'autres. Ces composants sont conçus pour être flexibles, réactifs et personnalisables afin de répondre aux besoins de conception des applications modernes.
* Thèmes et styles : PrimeNG offre également une variété de thèmes et de styles prédéfinis, ainsi que des outils de personnalisation, pour permettre aux développeurs de personnaliser l'apparence des composants selon les besoins de leur application. Cela permet d'assurer une cohérence visuelle et une expérience utilisateur homogène dans toute l'application.

PrimeNG est une bibliothèque de composants UI puissante et polyvalente pour Angular, offrant des fonctionnalités riches, une intégration facile, une documentation complète et une communauté active.

## Principales interfaces graphiques

### Interfaces graphiques

L'intégration du Template PrimeNG "Sakai" dans notre application est une décision stratégique pour améliorer notre interface. Motivée par sa modularité, sa facilité d'intégration et son chargement dynamique des ressources, cette approche accélère le développement, réduit

les coûts et optimise les performances. En exploitant ses fonctionnalités, nous livrons une UI attrayante, cohérente et réactive, respectant les délais du projet. Cette expérience souligne l'importance des Templates prêts à l'emploi pour optimiser nos processus de développement et offrir une expérience utilisateur exceptionnelle.

#### Interface d’authentification

L’utilisateur doit entrer son e-mail et son mot de passe correctement puis cliquer sur «Sign in », et selon son rôle, il est redirigé vers son interface correspondante à son rôle.

La figure 15 présente l’allure de l’interface de l’authentification :

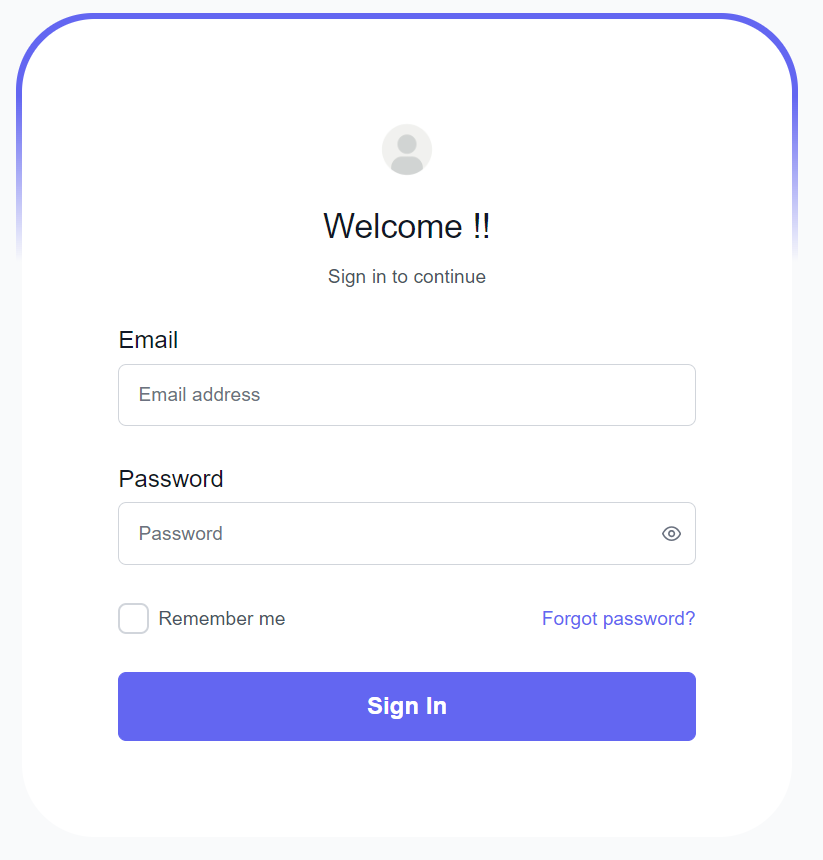
****

Figure 17 interface d'authentification

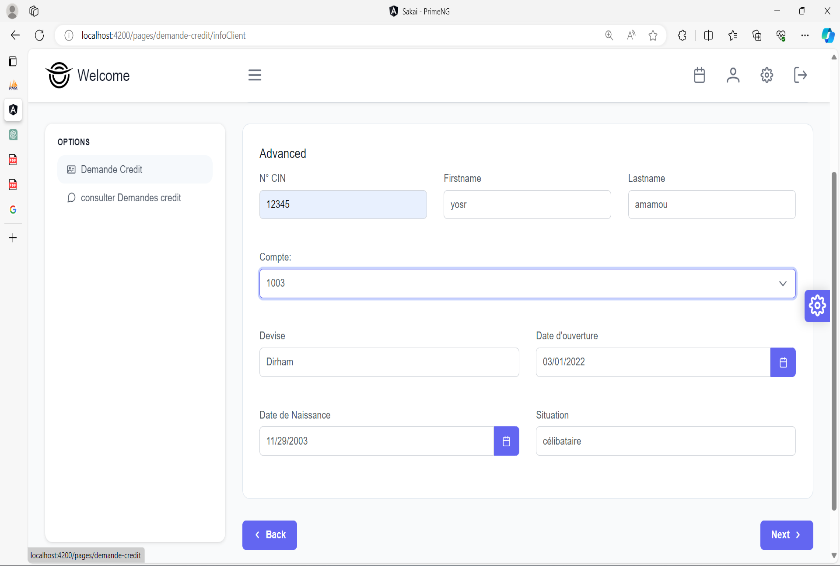
#### Interface client

* Formulaire demande crédit

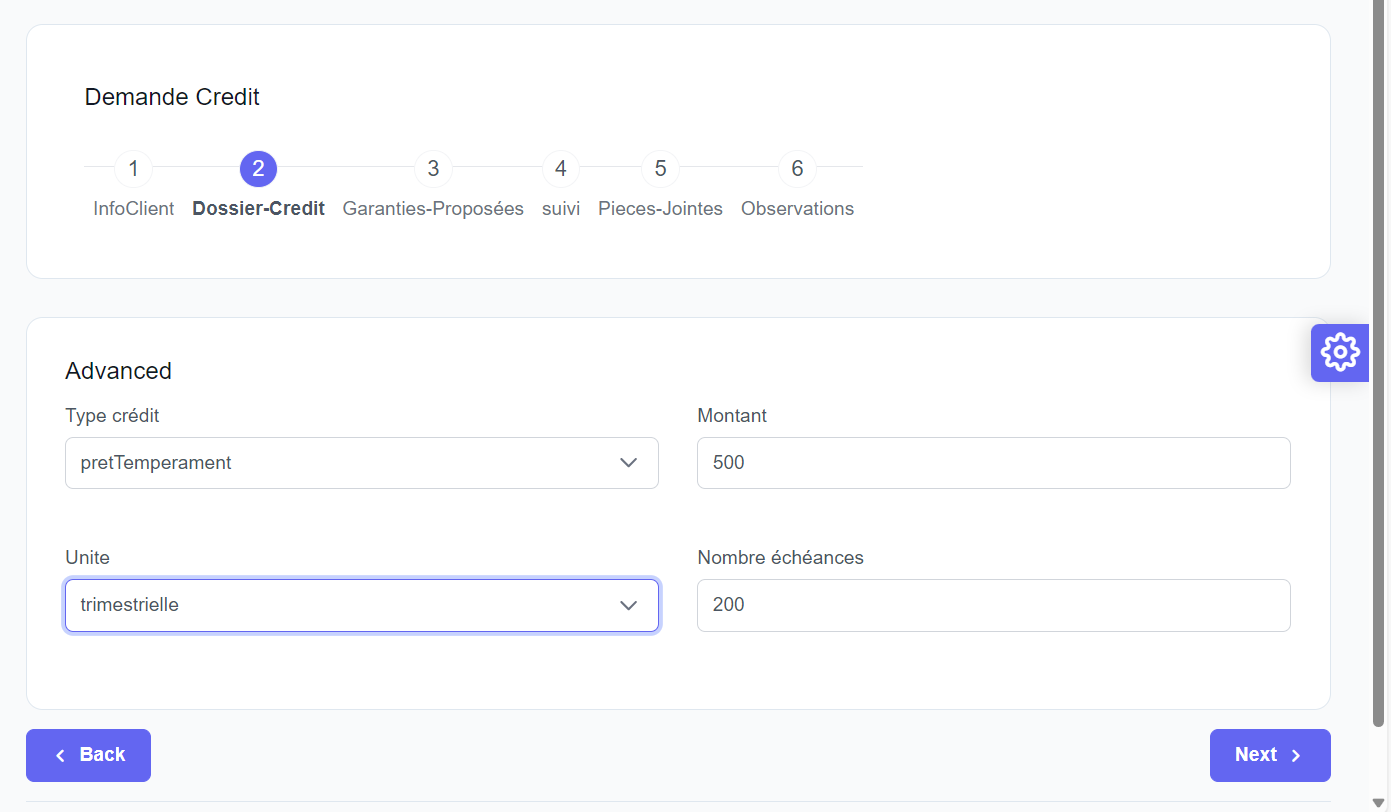
Notre formulaire de demande crédit comprend plusieurs étapes présentés ci-dessous :

Le client doit saisir son CIN correctement. Son nom, prénom, sa date de naissance, ses numéros de compte et sa situation seront générés automatiquement.

Le bouton « Next » permet au client de passer à la page suivante du formulaire, sauf si tous les champs sont valides.

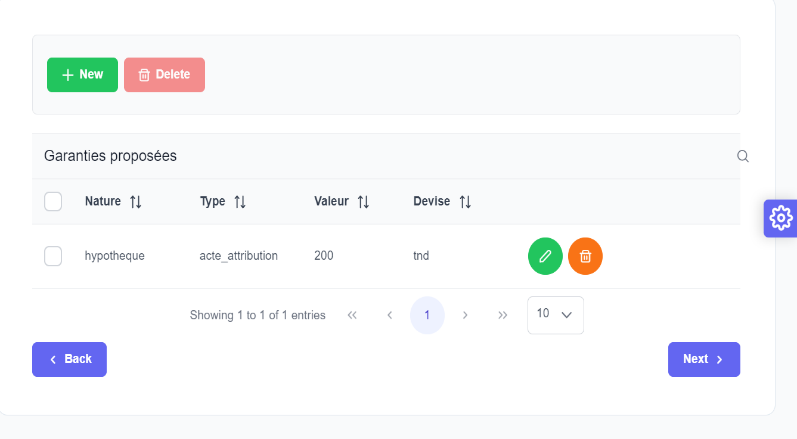
****

Le client est invité à sélectionner le type de crédit souhaité, à entrer le montant requis et à choisir l'unité de paiement correspondante. En conséquence, le nombre d'échéances sera automatiquement calculé, simplifiant ainsi le processus de planification financière pour le client :

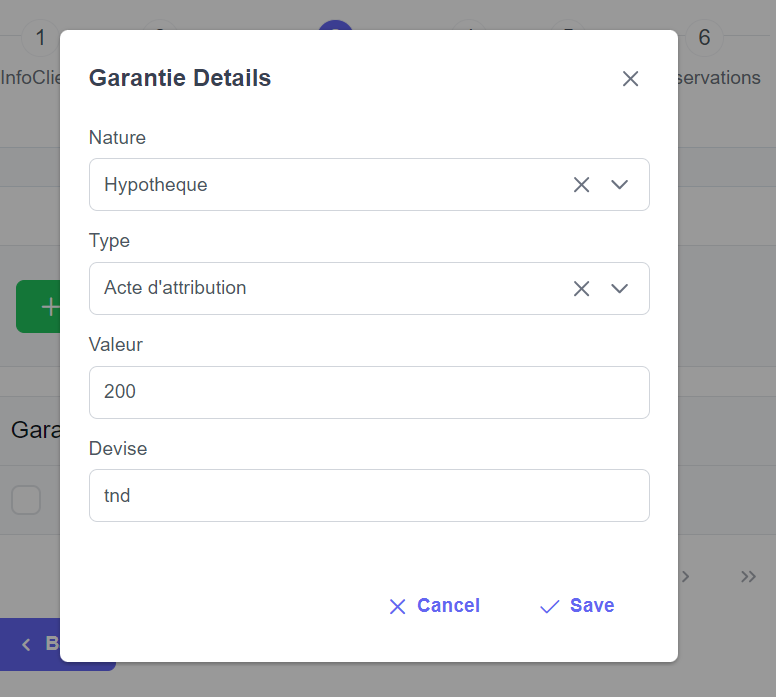
****

Le client doit proposer des lignes de garanties afin d’offrir une sécurité supplémentaire tant pour l'emprunteur que pour le prêteur.

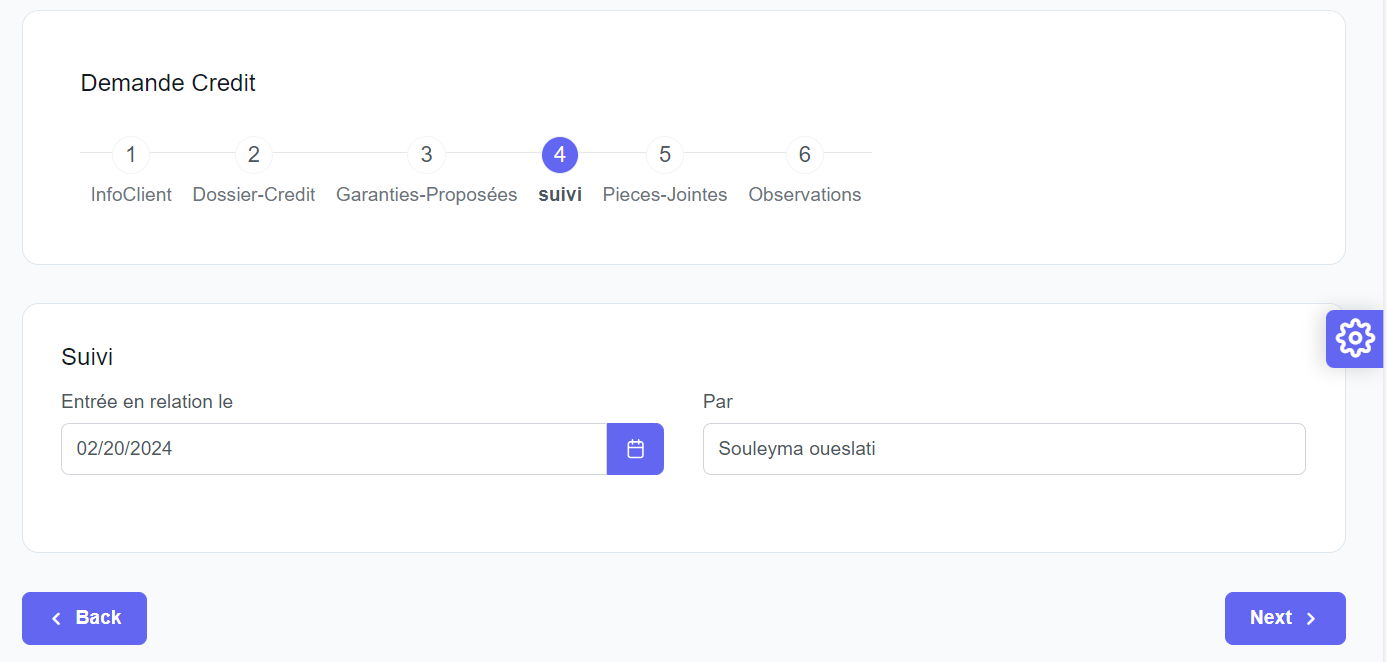
Le client a la possibilité d’ajouter, supprimer ou modifier ses lignes de garanties.

****

Le client doit remplir tous les champs correctement afin que sa garantie soit enregistrées.

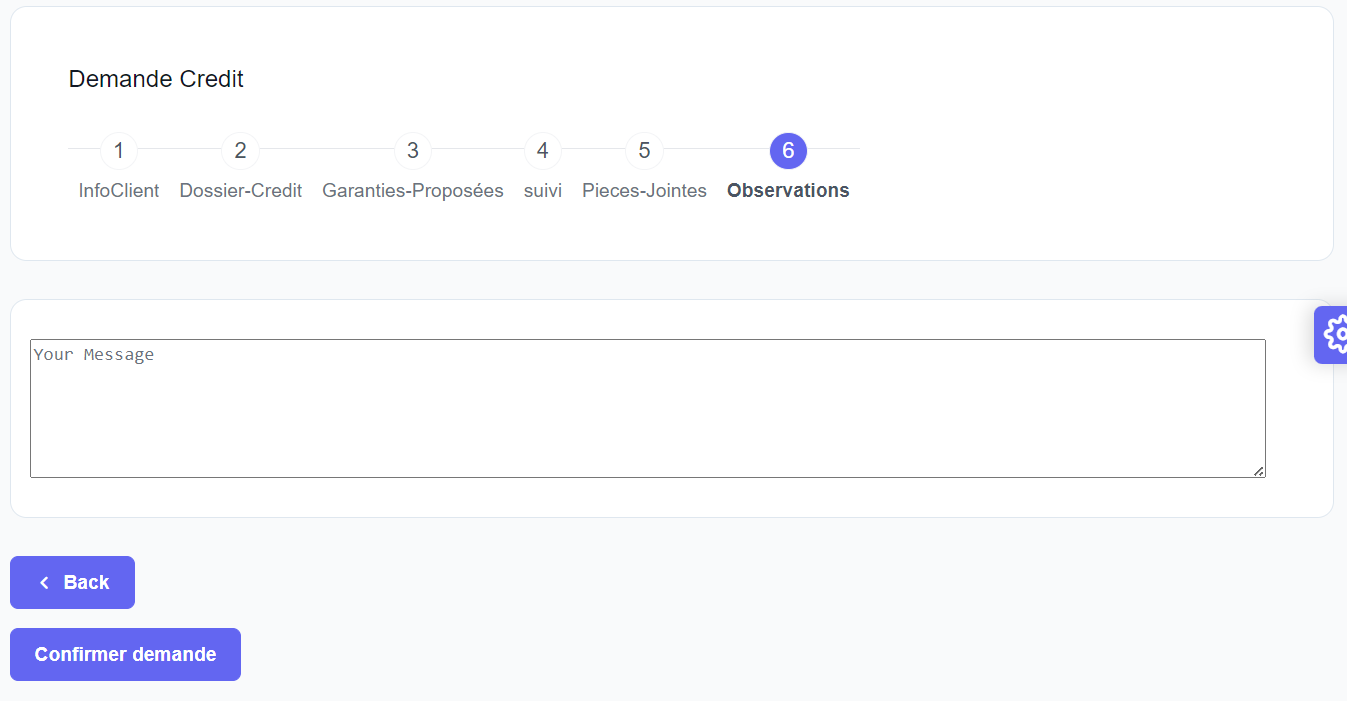
****

Le client doit introduire le nom du suivi et la date de relation entrée avec afin d’assurer un suivi efficace et une gestion appropriée de la relation client :

****

Selon le type de crédit choisi, des pièces jointes seront fournies au client pour qu'il puisse les importer :

Le client peut présenter ses observations afin de contribuer à l'amélioration continue des services ou des produits, de partager ses suggestions, d'exprimer son niveau de satisfaction.

****

* Consultation demandes crédit personnelles du client

Le client a la possibilité de consulter ses propres demandes et de suivre en temps réel l’état de ses demandes soumises :

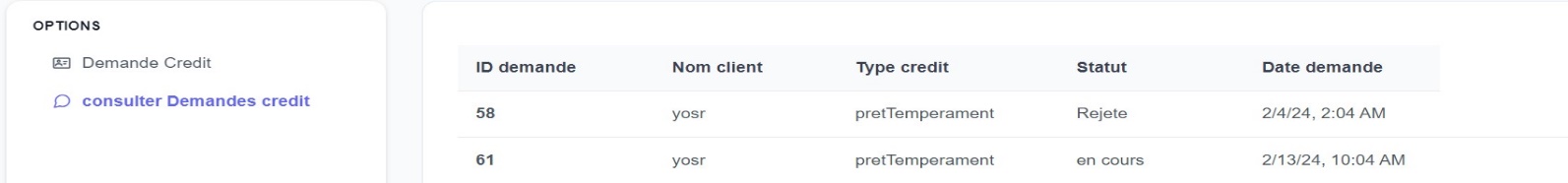


Figure 25 consultation demande crédit (rôle user)

#### Interface administrateur (le chargé de la banque)

 Dashboard

C’est la première interface que l’administrateur est redirigé vers afin de lui fournir une vue d'ensemble claire et facile à comprendre des données importantes pour les clients et leurs demandes crédit :

  Formulaire demande crédit

Le chargé de la banque peut passer une ou plusieurs demandes de crédit à partir de son compte d’administrateur sachant que c’est le même formulaire que le formulaire de l’interface client :

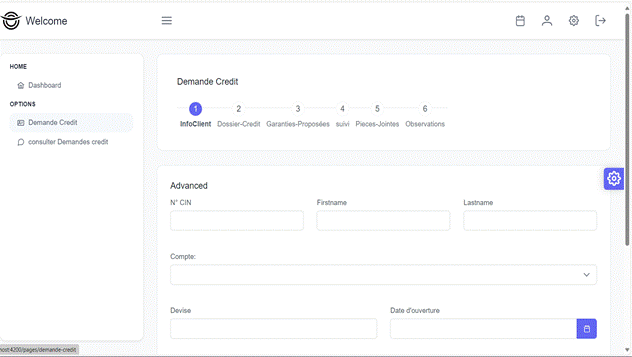


Figure 26 formulaire demande crédit (rôle admin)

  Consultation demande crédit

Le charge de la banque a la possibilité de consulter toutes les demandes soumises par les clients et de prendre l’une des actions suivantes : valider la demande ou rejeter la demande.

L’interface ci-dessous présente l’allure du tableau des demandes soumises :



Figure 28 interface consultation des demandes (rôle admin)

##### Test d’API

Nous réalisons un test d'API pour l'ajout d'une demande de crédit en utilisant Postman. Lors de ce test, nous identifions un point crucial : en l'absence d'authentification, une erreur "401 Unauthorized" est retournée. Cette erreur met en évidence l'importance de l'authentification pour sécuriser l'accès aux ressources de notre système, soulignant ainsi notre engagement envers la protection des données sensibles.

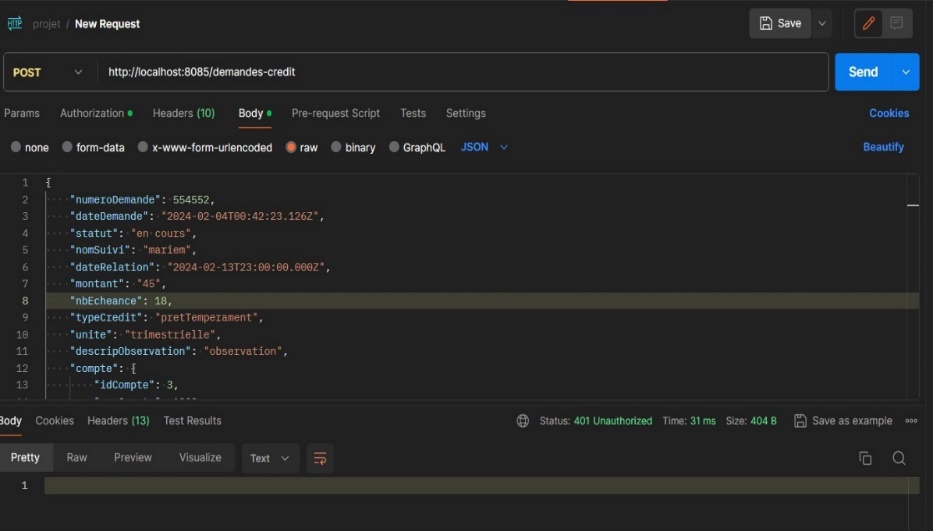


Figure 29 TEST API

Le chapitre de réalisation dévoile le développement complet de l'application "e-crédit", mettant en lumière son environnement matériel et logiciel, ainsi que ses interfaces utilisateur clés. Des tests d'API soulignent l'importance de l'authentification pour la sécurité. Le chapitre suivant explorera les concepts théoriques, complétant cette phase de développement.

# 

# Conclusion générale

Ce stage de perfectionnement à la Générale Tunisienne de l'Informatique (GTI) a été une expérience enrichissante et formatrice qui a profondément enrichi notre compréhension du monde professionnel de l'informatique et du secteur financier, notamment dans le domaine du développement d'applications.

Au cours de cette période, nous avons eu l'opportunité de mettre en pratique les connaissances acquises au cours de notre formation académique et d'explorer de nouveaux horizons professionnels. Plus précisément, nous avons participé au développement d'une application de demande de crédit destinée aux banques, ce qui nous a permis d'approfondir nos compétences en développement logiciel et en conception d'interfaces utilisateur.

Au sein du département des Technologies de l'Informatique (TI) de la GTI, ce stage nous a offert l'occasion d’appliquer nos compétences techniques pour concevoir et développer une solution logicielle robuste et adaptée aux besoins spécifiques des institutions bancaires.

Cette expérience pratique a été un tremplin essentiel pour notre future carrière professionnelle, mais surtout pour notre stage de Projet de Fin d'Études (PFE). Les connaissances et les compétences acquises lors de ce stage de perfectionnement ont jeté les bases solides nécessaires pour aborder avec confiance les défis et les opportunités qui nous attendent dans notre projet de fin d'études.

En collaborant avec des professionnels expérimentés du secteur, nous avons acquis une compréhension approfondie des exigences liés au développement d'applications pour les services financiers. Nous avons également été exposés aux meilleures pratiques et aux technologies de pointe utilisées dans le domaine du développement logiciel, ce qui a renforcé notre expertise et notre savoir-faire technique.

En conclusion, ce stage de perfectionnement a été une étape essentielle dans notre parcours professionnel, nous permettant de développer des compétences précieuses dans le domaine du développement d'applications pour le secteur financier. Nous sommes reconnaissants envers tous ceux qui ont contribué à rendre cette expérience aussi enrichissante et significative, et nous sommes impatients d'appliquer les connaissances et les compétences acquises dans nos futurs projets professionnels.

# Nétographie

|  |  |
| --- | --- |
| [1] |  |
| [2] |  |
| [3] |  |
| [4] |  |
| [5] |  |
| [6] |  |

**Annexe**

Diagramme de séquence système « remplir dossier crédit »

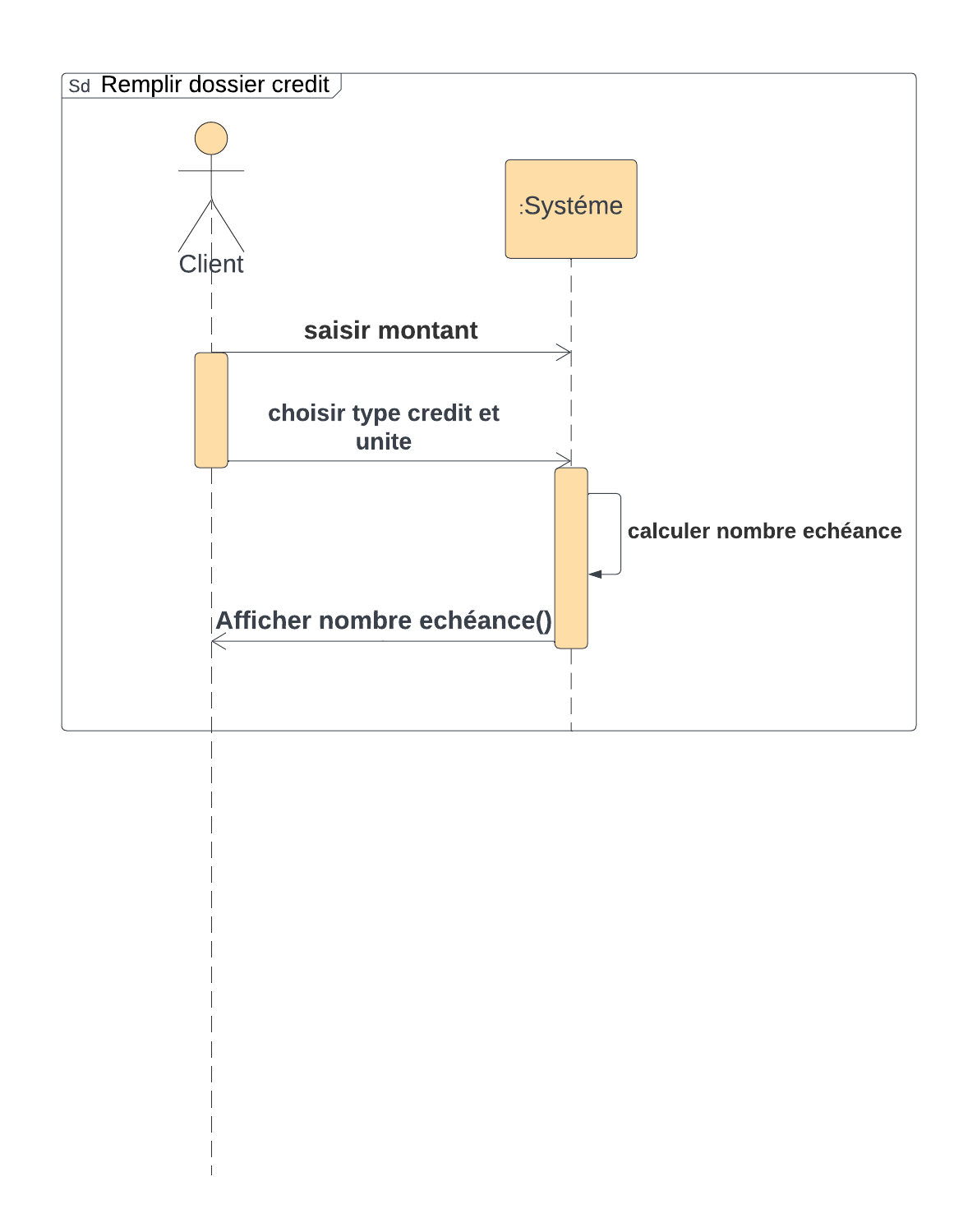


Figure 12 Diagramme de séquence « remplir dossier crédit »