Ministère de l'Enseignement Supérieur de la Recherche et de l'Innovation

Université Assane Seck de Ziguinchor

UFR Sciences et Technologies

Département Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master

Mention : Informatique ; Spécialité : Génie Logiciel

Sujet :

Conception et développement sous Salesforce d'un module de gestion des dépenses et de la facturation d'un projet

Présentépar : M. Youssouph Niaouba DIEDHIOU

Soutenance le 07/07/2020

Sous la direction de : Dr Khadim DRAME

Sous la supervision du - Pr. Salomon SAMBOU

Membres du jury

Salomon SAMBOU	Professeur	Président	UASZ
Khadim DRAME	Maître de Conférence Titulaire	Encadreur	UASZ
Ibrahima DIOP	Maître de Conférence Titulaire	Rapporteur	UASZ
Gorgoumack SAMBE	Enseignant-Chercheur	Rapporteur	UASZ
Mamadou BOUSSO	Maître de Conférence Titulaire	Maître de Stage	TérangaCloud Solutions

Résumé

Dans le cadre de mon projet de fin d'étude à l'université Assane Seck de Ziguinchor et en vue de l'obtention du titre d'ingénieur en informatique, j'ai effectué un stage de six mois à Téranga Cloud Solutions SA.

Durant mon séjour dans ladite société, j'ai dans un premier temps suivi une formation sur la plateforme Salesforce.

Dans un deuxième temps concevoir et développer un module de gestion des dépenses et facturations de projet en suivant un cycle de vie.

Dans un troisième temps, j'étais amené à intégrer mon module dans l'application globale de Gestion de Projet.

Ce module suit la méthode agile, l'utilisation du formalisme uml pour la réalisation de l'ensemble des diagrammes et enfin pour la modélisation j'ai utilisé Schéma Builder de la plateforme Salesforce pour une base de données robuste.

Abstract

As part of my end of study project at Assane Seck University in Ziguinchor and with a view to obtaining the title of computer engineer, I did a sixmonth internship at Téranga Cloud Solutions SA.

During my stay in the said company, I first took training on the Salesforce platform.

Secondly, design and develop a project expense and billing management module following a life cycle.

Thirdly, I had to integrate my module into the global Project Management application.

This module follows the agile method, the use of the uml formalism for the realization of all the diagrams and finally for the modeling I used Schema Builder of the Salesforce platform for a robust database.

Dédicaces

A mes chers parents plus particulièrement à mon défunt père (que le Bon DIEU l'accueille au paradis), pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études, A mes chères sœurs pour leurs encouragements permanents, et leur soutien moral,

A mes chers frères pour leur appui et leur encouragement,

A toute ma famille pour leur soutien tout au long de mon parcours universitaire,

Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fuit de votre soutien infaillible, Merci d'être toujours là pour moi.

Remerciements

En guise de reconnaissance, je tiens à témoigner mes sincères remerciements à toutes les personnes qui ont contribués de près ou de loin au bon déroulement de mon stage de fin d'étude et à l'élaboration de ce modeste travail.

Mes sincères gratitudes à mon encadreur Mr Khadim Dramé et tous mes professeurs pour la qualité de leurs enseignements, leurs conseils et leur intérêt incontestable qu'ils portent à tous les étudiants.

Je tiens à remercier l'ensemble du personnel de Téranga Cloud Solutions SA pour leur patience, leurs conseils pleins de sens et pour le suivi et l'intérêt qu'ils ont portaient à mes travaux.

Dans l'impossibilité de citer tous les noms, nos sincères remerciements vont à tous ceux et celles, qui de près ou de loin, ont permis par leurs conseils et leurs compétences la réalisation de ce mémoire

Enfin, je n'oserais oublier de remercier tout le corps professoral de l'université Assane Seck de Ziguinchor, pour le travail énorme qu'il effectue pour nous créer les conditions les plus favorables pour le déroulement de nos études.

Avant-Propos

Lors de ma formation en Génie Logiciel, il est d'usage d'effectuer un stage technique dans le but de mettre en œuvre les connaissances théoriques acquises et de développer l'aptitude de s'intégrer dans un environnement professionnel.

C'est dans cette optique, que j'ai été aimablement accueilli par Téranga Cloud Solutions SA pour effectuer mon stage. Le présent mémoire est donc le fruit de six mois de stage, effectué au sein de l'entreprise, ce stage est en fait un stage technique qui m'a permis d'intégrer des équipes professionnelles et de toucher à la culture, organisation, procédures, et de participer à l'élaboration, planification et réalisation de projets concrets au sein de l'établissement.

Sommaire :

I

X

Table des figures

Liste des tableaux

Sigles et Abréviations

Sigles	Significations
SA	Société Anonyme
SAAS	Software as a service
PAAS	Plateforme as a service

IAAS	Infrastrucure As A Service
VPN	Virtual Private Network
LAN	Local Area Network
SSL	Secure Sockes Layer
CRM	Customer Relationship Management
PME	Petites et Moyennes Entreprises
API	Application Programming Interface
DML	Data Manipulation Language
SOQL	Salesforce Object Query Language
SOSL	Salesforce Object Search Language
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JSON	JavaScript Object Notation
EEM	Entairtainement Expense management
ALM	Application Lifecycle Management
UML	Unified Modeling Language
IHM	Interface Homme Machine
SOC	Separation Of Concern
XML	Extensible Markup Language
SLDS	Salesforce Lightning Design System

Introduction

Le recours à un système de traitement automatique de l'information par les entreprises est devenu, depuis quelques années, une solution incontournable pour ces dernières. Automatiser les traitements peut aider à gagner en rapidité, fiabilité et traçabilité. L'informatisation d'une structure peut donc conduire, si elle est bien menée, à une meilleure productivité.

Ainsi dans le cadre de mon stage de fin d'études de Master 2 Informatique à l'UFR Sciences et Technologie de l'Université Assane Seck de Ziguinchor, Département Informatique, j'ai effectué un stage à TerangaCloud Solutions SA[1] (une entreprise spécialisée dans le cloud computing). Ce stage s'intègre dans un projet de recherche et d'application.

TerangaCloud Solutions SA est une nouvelle entreprise en relation avec des partenaires européens spécialisés dans le cloud computing et le développement d'applications sur Salesforce.

Le développement de l'application de gestion de projet s'inscrit dans une logique commerciale avec la volonté de l'entreprise de déployer l'application dans AppExchange (la boutique des applications Salesforce pour pouvoir le commercialiser à des milliers d'entreprises à travers le monde).

Le développement de l'application avec une nouvelle technologie, qui est Salesforce Lightning, nous a plongé dans un monde vaste, où il n y'a pas assez de forums de développeurs nous incitant à faire de la recherche, de la conception et à établir une architecture d'application évolutive et performante.

Mon stage s'inscrit dans la conception et le développement d'un module de gestion des dépenses et facturations de projet robuste et évolutif avec la mise en place d'une architecture personnalisée et l'utilisation de quelques patrons de conception pour rendre l'application performante et maintenable.

Dans une première partie, nous présenterons le Cloud. Nous présenterons ensuite la plateforme Salesforce et ses fonctionnalités en rapport avec le projet, le prototypage de l'application et la

[1] Société Anonyme

réalisation du projet. Dans la dernière partie, nous présenterons l'application et son déploiement dans l'organisation Salesforce dédiée à l'Université Assane Seck de Ziguinchor.

Chapitre 1: Présentation de la structure d'accueil

1.1 Présentation générale

Téranga Cloud Solutions est le premier partenaire intégrateur certifié de Salesforce, plateforme n°1 de solution de gestion de la relation client. Elle est installée à Thiès, ville qui est située à 70 km de Dakar capitale du Sénégal. Téranga Cloud Solutions est le leader dans le domaine de l'intégration CRM[2] en Afrique de l'ouest (www.Terangacloudsolutions.com).

L'entreprise propose des services de développement Salesforce, d'intégration et de formation en passant par l'audit de votre CRM, la migration de données, l'administration et le support AM mais aussi des solutions BI[3] afin d'aider les entreprises à identifier, séduire et fidéliser de plus en plus de client.

[2] Customer relationship management
[3] Bussiness Intelligence

Figure 1 Logo de L'entreprise TérangaCloud Solutions

1.2 Présentation des différents services de l'entreprise

L'entreprise propose différents services à savoir :

Service	Présentation
L'Audit CRM et réingénierie des processus :	Ce processus permet à une entreprise de gérer de façon efficace les données clients en permettant une accessibilité simple et automatique et une meilleure gestion de la communication. À travers un examen complet du code, de l'architecture applicative et de la plateforme, l'audit CRM permet : D'identifier les freins de performances qui pénalisent l'application, et de préconiser les solutions adaptées aux difficultés rencontrées ; De remodeler votre application au niveau de son architecture et/ou de son code à moindre coût et dans des délais raisonnables.
Développements d'applications	Pour chaque projet une méthodologie rigoureuse est appliquée : Aide au choix de la solution ; Cadrage et définition du besoin ; Gestion de la conduite du changement ;

	Définition de l'architecture fonctionnelle ; Pilotage de projet et assistance méthodologique ; Solutions développées dans une « sandbox » ; Validation accélérée en mettant en relation les développeurs et les équipes opérationnelles autour d'un démonstrateur ; Déploiement fait par étapes sans interruption de services ; Maintenance prévue dès la phase de conception pour réduire les coûts ; Assurance d'un haut niveau d'adoption des utilisateurs grâce à la formation.
Mise en œuvre et déploiement Salesforce :	Téranga Cloud Solutions vous permet de personnaliser votre CRM et de l'intégrer avec votre SI à travers : La Modélisation des processus / workflow de l'entreprise ; L'Intégration par connecteurs, mash-ups avec les autres applications de votre SI (Google apps, Microsoft Outlook, box…) ; L'Echange de données par services Web Apex, callouts Apex, API[4] (SOAP[5], REST, …), messages ; Export vers l'outil de production comptable (journal des ventes JDV, journal de situation JDS, …) Sage comptabilité générale/analytique.

[4] Application programming interface
[5] Simple object Access Protocol

Formations	L'entreprise propose aussi des formations qui permettent d'acquérir des compétences dans le domaine Salesforce. Elle est surtout connue au niveau international pour ses solutions en gestion de la relation client. Les sociétés peuvent notamment profiter des dernières innovations mobiles, cloud et issues des réseaux sociaux, pour mieux gérer leurs ventes, leur service client et leur marketing. Téranga Cloud Solutions, vous propose ci-dessous les formations Salesforce les plus pertinentes de l'éditeur : Formation utilisateur commercial ou marketing ; Formation manager ; Formation administrateur ; Formation développeur. Elle propose aussi d'autres services comme la mise en place d'une solution CRM Salesforce® performante et adaptée à votre métier ce qui permet D'optimiser vos processus métiers et les temps de réponse de vos applications ; D'analyser et de mettre en place des solutions d'intégration de données ; De mettre en place des fonctionnalités complémentaires ; De refondre votre application pour l'adapter à vos besoins ; De définir les règles de sécurité (rôles et profils) de votre organisation ;

	D’implémenter des workflows pour automatiser vos processus, assigner des tâches, envoyer des relances par email ; De Mettre en place de formulaires web pour faire le lien entre votre site Internet et votre CRM ; De mettre en place des rapports et tableaux de bord Salesforce.

1.3 Organigramme :

L’entreprise Téranga cloud solutions comporte plusieurs services structurés de la façon suivante :

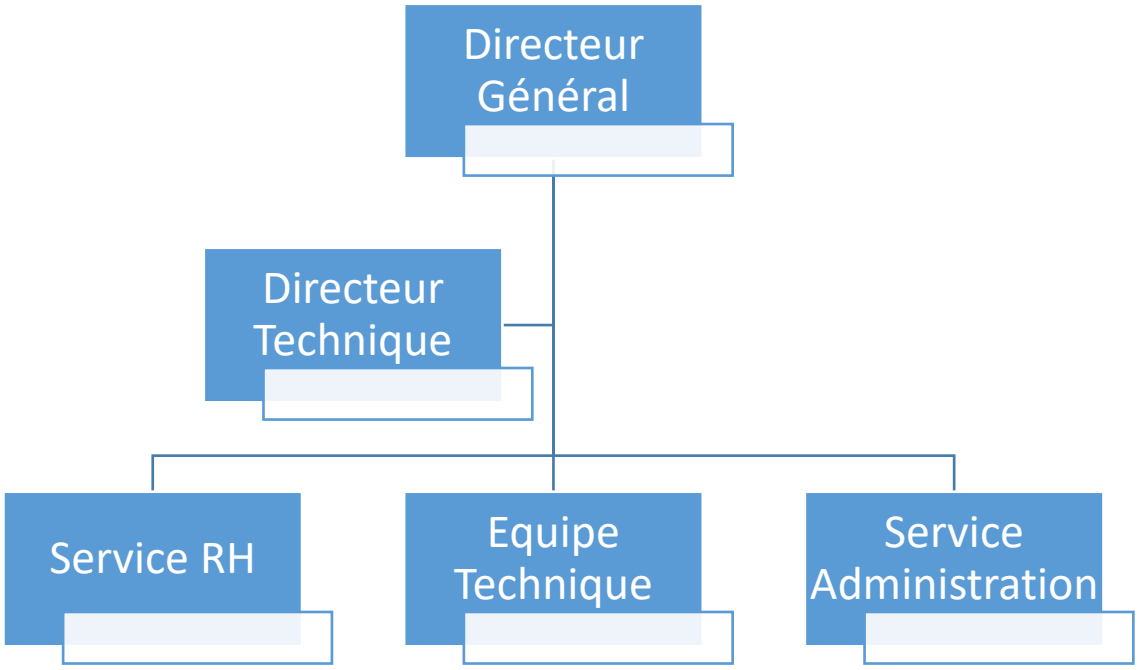


Figure 2 Organigramme de l'entreprise

Chapitre 2: Le Cloud

2.1 **Définition** :

Le cloud (ou cloud computing) est une technologie qui permet d'utiliser un ensemble ressources distant accessibles via Internet. Ces ressources sont stockées dans des serveurs distants, elles sont accessibles à tout moment et en tout lieu via internet.

Les différents intervenants disposent à cet effet de gigantesques champs de serveurs de stockages appelés Datacenter. Le Cloud est aussi souvent appelé Cloud Computing ou Nuage Informatique.

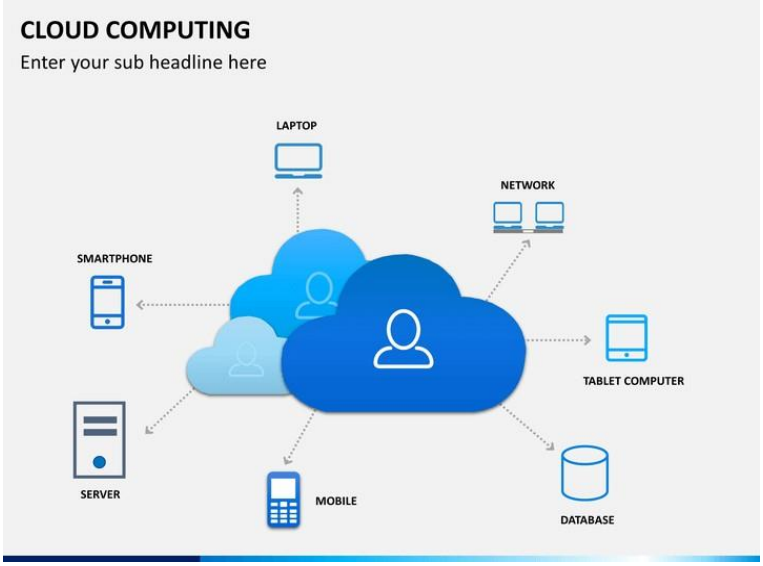


Figure 1 Le Cloud

2.2 Modèles de service de cloud:

Il existe différents types de service de cloud : SAAS[6] ,PAAS[7] ,IAAS[8].

[6] Software as a service : https://www.1min30.com/dictionnaire-du-web/saas-software-as-a-service
[7] Plateforme as a service : https://www.lemagit.fr/definition/Platform-as-a-Service-PaaS
[8] Infrastructure as a service : https://www.ionos.fr/digitalguide/serveur/know-how/iaas-infrastructure-as-aservice/

Figure 2 Modèles de Service de Cloud

✓ Le Saas :

Le Software as a Service ou Logiciel en tant que Service en Français, est un modèle de distribution de logiciel au sein duquel un fournisseur tiers héberge les applications et les rend disponibles pour ses clients par l'intermédiaire d'internet.

Grâce à un logiciel SaaS, les entreprises n'ont plus besoin d'installer et de lancer des applications sur leurs propres ordinateurs ou sur leurs Data Centers. Le coût d'acquisition de matériel est ainsi éliminé, au même titre que les coûts d'approvisionnement et de maintenance, de licence de logiciel, d'installation et de support. On compte également plusieurs autres avantages.

Au lieu d'investir dans un logiciel à installer, et dans un équipement permettant de le prendre en charge, les utilisateurs souscrivent à une offre SaaS. En général, l'offre se présente sous la forme d'un abonnement mensuel dont le tarif est proportionnel à l'utilisation. Grâce à cette flexibilité, les entreprises peuvent organiser leur budget avec plus de précision et de facilité. De plus, il est possible de résilier l'abonnement à tout moment pour couper court aux dépenses.

Figure 3 Le Software as a Service

✓ Le PAAS :

Une Plateforme en tant que Service est un service Cloud Computing permettant aux entreprises d'externaliser l'hébergement des outils logiciels et matériels de développement d'applications. Découvrez les différents avantages, inconvénients et tarifs de ces services.

La Plateforme en tant que Service (PaaS) est un modèle de Cloud Computing, au même titre que les SaaS, les PaaS et les IaaS. Un fournisseur de services Cloud propose des outils hardware et logiciels en tant que service via internet, permettant à l'utilisateur de développer des applications. Le hardware et le software sont hébergés sur l'infrastructure du fournisseur.

Ainsi, les utilisateurs n'ont pas besoin d'installer leur propre hardware et leurs logiciels en interne pour développer ou lancer de nouvelles applications.

Parmi les principales fonctionnalités proposées par les fournisseurs de PaaS, on dénombre le système d'exploitation, l'environnement de programmation, le système de gestion de base de données, le logiciel de serveur, le support, le stockage, l'accès réseau, les outils de design et de développement, et l'hébergement. Bien entendu, les fournisseurs se distinguent en proposant des fonctionnalités supplémentaires plus spécifiques. Il est préférable d'étudier toutes les offres au cas par cas.

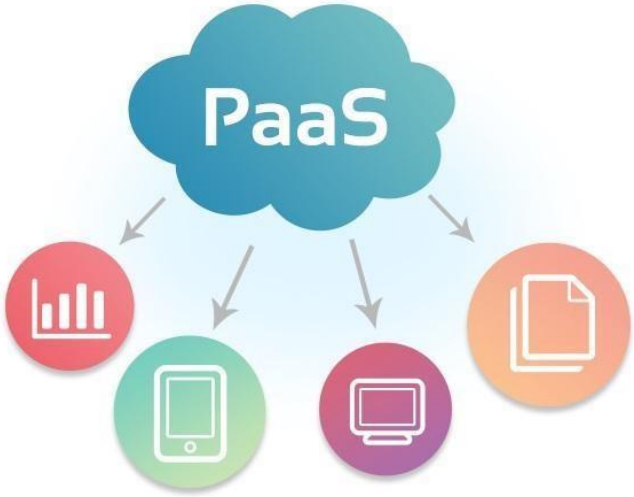


Figure 4 Le Plateforme As A Service

✓ Iaas Infrastructure as a service :

L'Infrastructure en tant que Service est l'une des principales composantes du Cloud Computing. Elle permet de bénéficier de ressources informatiques virtualisées. Découvrez la définition complète de ce type de service Cloud, ainsi que ses avantages et ses inconvénients pour les entreprises.

L'Infrastructure en tant que Service est une forme de Cloud Computing offrant des ressources informatiques au sein d'un environnement virtualisé (le Cloud) par le biais d'internet ou d'une autre connexion. L'IaaS est l'une des quatre principales catégories de services Cloud Computing, au même titre que le Software as a Service , le Desktop as a Service , et le plateform as a service .

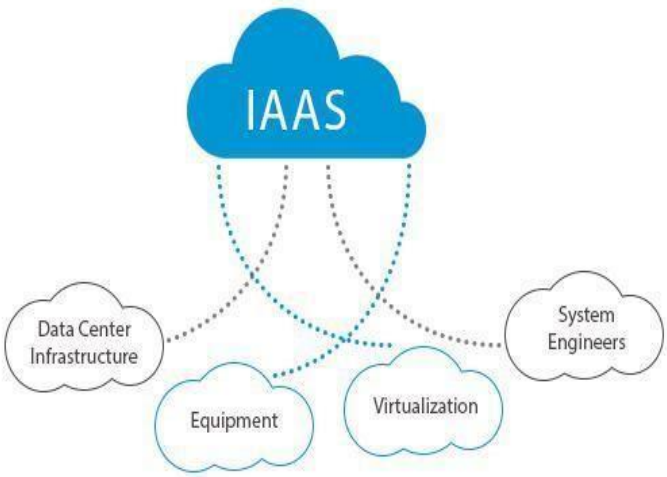


Figure 5 L'Infrastructure As A Service

2.3 Types de cloud

Il existe différents types de cloud computing. Mais, le type de services cloud n'est pas le seul critère à prendre en considération lorsque l'on souhaite passer au cloud computing. Il existe différentes implémentations qui offrent des niveaux de sécurité et de personnalisation variables.

- **Le cloud public :** les ressources informatiques de l'entreprise sont stockées sur un serveur mutualisé, autrement dit partagé entre plusieurs clients, et accessibles par Internet. Ces serveurs sont partitionnés de manière à interdire les fuites de données.
- **Le cloud privé :** comme son nom l'indique, il est dédié à un seul utilisateur. L'avantage du cloud privé est son important niveau de sécurité, renforcé par une connexion VPN[9]. Le cloud privé est administré par l'entreprise elle-même ou un prestataire de services.
- **Le cloud hybride :** l'entreprise utilise à la fois le cloud privé et le cloud public pour mettre en œuvre certaines activités. Par exemple, le cloud public est utilisé par les collaborateurs pour les tâches opérationnelles, tandis que le cloud privé sert à héberger le site web ecommerce de l'entreprise ou ses données financières, pour réduire le risque de piratage.
- **Le cloud communautaire :** plus rarement utilisé, il consiste à partager un espace donné entre plusieurs entreprises ayant les mêmes exigences en matière de sécurité et de confidentialité. Il s'apparente donc à un cloud privé partagé.

2.4 Les avantages et les inconvénients du cloud:

2.4.1 Les avantages

Le cloud computing est généralement associé à une multitude d'avantages qui créent l'unanimité parmi les professionnels de l'entreprise. Notez toutefois que ces avantages demeurent théoriques, étant donné la nature même du concept.

- La possibilité de déployer et de rendre disponibles des applications majeures et des environnements de travail de manière immédiate. La mise à jour des applications est systématique, et le fournisseur décharge son client de toute responsabilité de maintenance. Une simplicité imbattable donc, qui vous épargne en plus les développements coûteux
- Les données peuvent être partagées, puisque tout utilisateur du cloud computing peut aisément rendre disponibles ses données à un ou plusieurs autres utilisateurs du. Il est donc possible de créer une plateforme virtuelle collaborative en un temps record.

[9] Virtual private network : https://www.expressvpn.com/fr/what-is-vpn

- Un calcul particulièrement puissant, ce qui constitue probablement l'argument de choc en faveur de ce type de solution. Il faut effectivement garder à l'esprit que les structures limitées à ce niveau (puissance de calcul) peuvent ici se permettre une délocalisation de leurs traitements, et bénéficier ainsi de toutes les ressources et performances mises à leur disposition par le serveur du cloud computing. Bien qu'il ne concerne qu'un nombre assez réduit d'entreprises, cet avantage demeure l'un des plus importants du Cloud Computing.
- Un accès libre et ouvert au client, qui peut établir sa connexion de n'importe où et avoir accès à ses données immédiatement, sans passer par la mise en place d'un VPN dans l'entreprise.
- Un suivi constant du développement de votre espace cloud computing. Vous êtes généralement informé, en temps réel, de l'évolution de votre plateforme de cloud computing, puisque l'installation d'un logiciel n'est pas nécessaire et que l'accès est effectué via un simple navigateur web
- Une liberté totale, puisque vous n'êtes lié à votre fournisseur par aucun engagement à long terme. Les services du cloud computing sont soit facturés à la demande ou par abonnement mensuel. Vous demeurez donc libre de mettre un terme à ce service à tout moment, si vous jugez n'en avoir plus besoin, ou si vous désirez simplement changer de fournisseur.
- Coût : du fait que le même service est proposé à de nombreux utilisateurs, son coût en est nettement amoindri.

2.4.2 **Les inconvénients du Cloud Computing**:

La question controversée de la confidentialité et de la sécurité des vos données demeure la limite majeure de cette solution. L'hébergement de vos données se fait en effet en dehors de l'entreprise, dans un service de base mis à votre disposition par votre fournisseur. Le risque de voir vos données finir en situation de vol ou de mauvaise utilisation demeure donc une possibilité.

Notre rôle, à ce niveau, consiste à veiller à ce que votre fournisseur propose une sécurité suffisamment exhaustive et à ce qu'il mette à notre disposition une politique de confidentialité englobant toutes vos données.

En fonction du fournisseur choisi, notre marge de manœuvre peut être limitée par la nature de l'offre proposée. Par exemple, si nous souhaitons accéder à certaines fonctionnalités de notre choix, notre fournisseur pourrait être dans l'impossibilité de les proposer.

2.5 **Le développement d'applications dans le cloud**:

Le développement d'application dans le cloud computing se pose comme un défi pour les développeurs, car il nécessite la maîtrise de notion.

En effet, les compétences et connaissances en développement ont assurément changé avec l'émergence des applications Cloud. Ainsi nous allons essayer de mettre en exergue la différence entre le développement d'application dans le cloud computing et dans le classique.

- **Notion de gestion de la défaillance :**

En effet dans le Cloud cette notion est considérée comme faisant partie de l'application, c'est à dire une fonction inévitable, et non comme un événement qui interrompt l'ensemble des fonctions de cette application. Il est important pour un développeur de savoir que l'approche qui consiste à réparer un serveur unique lorsqu'une défaillance apparaît est obsolète dans le monde Cloud car les serveurs sont multiples : en cas de défaillance de l'un d'entre eux, il suffit de le suspendre. ▪ **Le temps de latence :**

La latence est toujours supérieure sur Internet que sur un LAN[10]. Certaines transactions peuvent ainsi être très rapides en local, mais plus lentes dans le Cloud. Cela est dû aux volumes de requêtes effectuées sur le réseau. Les développeurs Cloud doivent tenir compte de cette latence et savoir la gérer.

- **Les filtres**

En effet les applications traditionnelles sur site reposent souvent sur des pare-feu dont l'objectif est de filtrer les accès. Même si ces accès peuvent aussi être limités dans le Cloud, cela n'est souvent pas souhaitable.

Dans le Cloud, l'idée est en effet de pouvoir accéder aux applications depuis n'importe où. Le bureau classique n'est plus là où se trouve l'activité. Avec le Cloud Computing, vous pouvez protéger vos applications par d'autres moyens que le pare-feu; par exemple via des procédés comme la fédération d'identités et les connexions SSL[11].

[10] Local area network : https://test-et-avis.com/quest-ce-quun-local-area-network-lan/

[11] Secure Socket Layer : https://kinsta.com/fr/base-de-connaissances/comment-fonctionne-ssl/

Chapitre 3: Salesforce

1.4 Historique et présentation de l'entreprise Salesforce:

Salesforce est un éditeur de logiciels, basé à San Francisco aux États-Unis. Il distribue des logiciels de gestion basés sur Internet et héberge des applications d'entreprises. L'entreprise est surtout connue au niveau international pour ses solutions en gestion de la relation client.

En 1999, Salesforce est né d'une vision : celle de réinventer la gestion de la relation client (CRM)[12]. Depuis lors, son utilisation innovante du Cloud computing a révolutionné la mise en place et l'utilisation des logiciels d'entreprise, transformant fondamentalement ce secteur.

Tous leurs produits sont 100% Cloud, et ce, depuis leur premier outil CRM. Tout est donc en ligne : pas de logiciel, pas de matériel. Ainsi, vous n'avez aucun coût de configuration, aucune maintenance et vos collaborateurs peuvent travailler depuis tous les terminaux avec une simple connexion Internet (smartphone, tablette, ordinateur portable…).

En outre, avec 3 montées en version annuelles, leur outil convient aussi bien aux PME[9] qu'aux grandes entreprises. Cette approche révolutionnaire est l'un des facteurs de la réussite de Sales Cloud, l'outil CRM n° 1.

Mais Salesforce ne se limite pas à un seul produit. C'est une plateforme complètement intégrée permettant de gérer toutes interactions clients et prospects, conçu pour aider à prospérer et à réussir.

Noms	Année	Montant
Datorama	Juillet 2018	-
Rebel Mail	Octobre 2018	-
Griddable.io	Janvier 2019	-
MapAnything	Avril 2019	-
Tableau	Juin 2019	15.3 milliards de dollars

Tableau 1: Chiffres d'affaire

[12] Customer relationship management: https://blog.hubspot.fr/customer-relationship-management
[9] Petites et moyennes entreprises :

1.5 Présentation du CRM:

Salesforce est un logiciel de gestion client entièrement basé sur le cloud. Un CRM est une application utilisée pour automatiser les ventes et les fonctions de marketing en utilisant un logiciel appelé CRM Software.

Exemple de quelques CRM : SAP CRM, Sage CRM

✓ Architecture de Salesforce

Salesforce est représentée sous forme de couches superposées.

Les couches partent de la couche application jusqu'à la couche plateforme.

Figure 6 Architecture de Salesforce

La force de Salesforce repose sur les aspects suivants :

➢ Architecture mutualisée :

Salesforce offre un arrangement central des administrations à chacun de ses clients dans le cloud mutualisé. Quelle que soit la portée de votre entreprise, vous bénéficiez d'un pouvoir d'enregistrement, d'un stockage d'informations et de fonctionnalités centrales similaires. Alors qu'une application habituelle pour un seul habitant nécessite un arrangement engagé d'actifs pour satisfaire les besoins d'une seule association, une application multi-locataires peut répondre aux besoins de nombreux occupants (organisations ou divisions au sein d'une organisation, etc.) en utilisant les actifs matériels et le personnel censés superviser uniquement un cas de programmation solitaire.

Single-Tenant vs Multi-Tenant Single-Tenant

:

Une seule application destinée à servir plusieurs clients business et celle-ci est installée sur une seule plage d'instance pour chaque client. **Multi-Tenant :**

Une seule instance d'application va servir plusieurs clients. Chaque client est appelé locataire (tenant en Anglais). Elle s'oppose à une architecture Single-Tenant où chaque organisation cliente à sa propre instance d'installation logicielle ou matérielle.

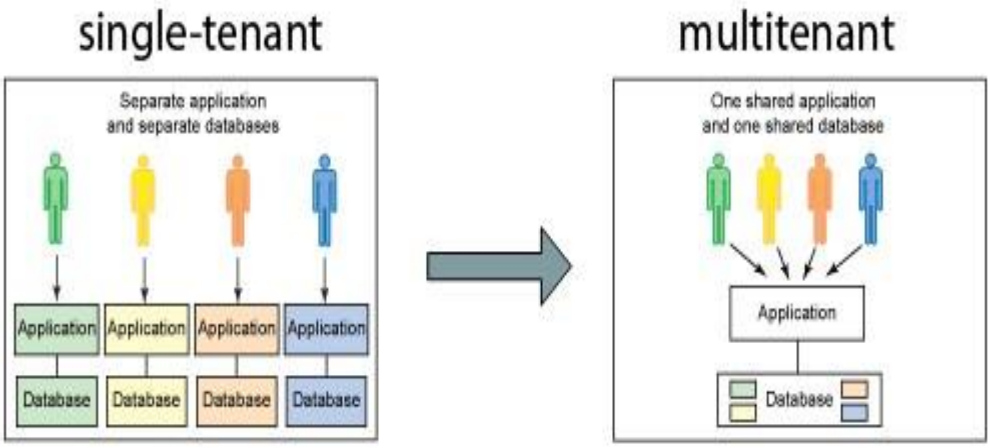


Figure 7 Single tenant vs Multitenant

- Métadonnées

Dans les architectures pilotées par les métadonnées, la multi localisation est viable lorsqu'elle peut prendre en charge des applications fiables, adaptables, évolutives, sécurisées et rapides. Il est difficile de rendre une application à ordre statique exécutable qui puisse répondre à ces critères et à d'autres difficultés de la mutualisation. De façon caractéristique, une application multi-locataire doit être de nature dynamique ou polymorphe pour satisfaire les désirs individuels des différents habitants et de leurs clients. Ainsi, les plans d'application multi-locataires ont été développés pour utiliser un moteur d'exécution qui produit des parties d'application à partir de métadonnées - informations sur l'application elle-même.

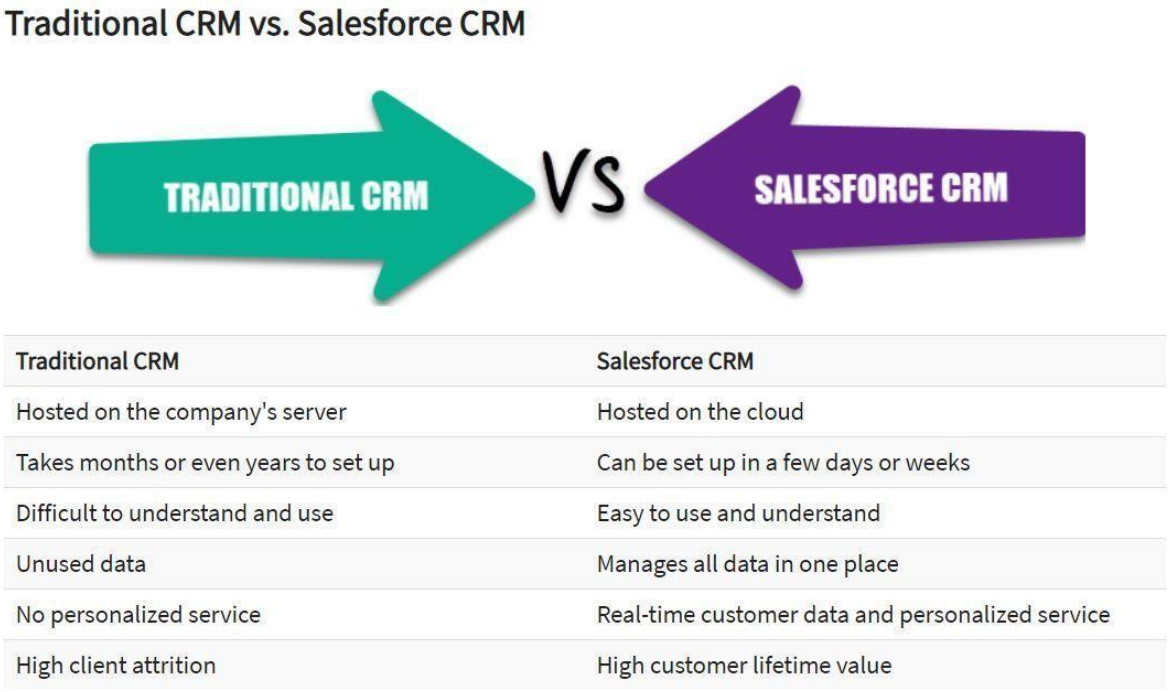
- API

A un niveau très basique, les API[13] permettent à divers bits de programmation de s'interfacer entre eux et de négocier des données. Si par hasard cela vous semble un peu théorique, jetez un coup d'œil au PC sur lequel vous travaillez en ce moment même. Vous pouvez sans doute découvrir une progression de connexions de différentes formes et tailles qui aident divers types d'associations. Celles-ci ressemblent à la forme d'équipement des API. Vous n'avez pas besoin de savoir comment

[13] Application programming interface : https://www.redhat.com/fr/topics/api/what-are-applicationprogramming-interfaces

le port USB[14] fonctionne. Vous devez simplement comprendre que lorsque vous connectez votre téléphone à un port USB, il transmet des données à votre PC.

Les API sont comparables. Sans en connaître les éléments subtils, vous pouvez interfacer vos applications avec différentes applications ou cadres de programmation. L'innovation fondamentale porte sur les spécificités du passage des données à travers le framework.



Traditional CRM	Salesforce CRM
Hosted on the company's server	Hosted on the cloud
Takes months or even years to set up	Can be set up in a few days or weeks
Difficult to understand and use	Easy to use and understand
Unused data	Manages all data in one place
No personalized service	Real-time customer data and personalized service
High client attrition	High customer lifetime value

Figure 8 Salesforce vs CRM Traditionnel

1.6 Les différents services de Salesforce :

La plateforme Salesforce offre un ensemble de services dont les plus connus sont : Sales Cloud. Services Cloud, AppExchange, Chatter, Marketing cloud, Analytics Cloud… ●

Sales Cloud :

Sales Cloud est une application dédiée à la productivité d'une force de vente en ligne accessible à travers une connexion Internet par mobile ou ordinateur. Elle permet de consulter à tout moment le profil de clients ou de prospects, de gérer des campagnes marketing et de rassembler toutes les informations essentielles pour optimiser la force de vente d'une entreprise.

- Service Cloud :

[14] Universal Serial Bus

Services Cloud est une plate-forme permettant d'optimiser la gestion de la relation client d'une entreprise. Elle offre divers services comme la centralisation d'informations clients pertinentes et l'historique provenant de centre d'appels ou encore la possibilité pour chaque utilisateur de participer aux conversations sur leur entreprise au sein des réseaux sociaux. Services Cloud fournit aussi des outils d'analyse, des services d'e-mail et de tchat pour permettre aux sociétés de communiquer en temps réel avec leurs clients et ainsi optimiser leur service client.

- Force.com et AppExchange :

Force.com est une plate-forme permettant de créer des applications d'entreprise hébergées sur le web et ne nécessitant aucun logiciel ni matériel. Ainsi toute entreprise peut créer son site web et ses propres applications pour diverses fonctions comme les ressources humaines, la gestion de projet, la logistique… Avec le lancement de la Salesforce1 Platform, les fonctionnalités de Force.com se sont enrichies : les applications métiers développées par les entreprises sur la plate-forme peuvent désormais être déployées sur tout type de terminal mobile. AppExchange est une place de marché (App Store pour entreprise) qui permet d'acheter l'une des 2 800 solutions d'éditeurs de logiciels partenaires ou de vendre des logiciels à plus de 100 000 clients salesforce.com.

- Heroku

Heroku est un service de cloud computing de type plate-forme en tant que service. Il permet le déploiement très rapide d'applications dans le cloud, avec une gestion souple du scaling horizontal au travers d'un modèle de gestion des processus emprunté à Unix et adapté au Web. Il supporte différents langages comme Python, Ruby, PHP, Java. Facebook le propose par défaut aux développeurs qui veulent concevoir des applications sur le réseau social. Heroku est également intégré à Force.com et permet ainsi le transfert de données entre ces deux plates-formes. ● Chatter et Community Cloud

Créé en juin 2010, Chatter est une plate-forme collaborative en temps-réel. Elle permet à tous les collaborateurs d'une même entreprise de mettre à jour leur profil et ainsi échanger et partager tout type d'informations sur leurs activités professionnelles du quotidien. Ce service de réseau social d'entreprise fait partie de la plateforme Salesforce1. Son point fort, selon Salesforce, est qu'il intègre des process métiers avec lesquels les employés peuvent collaborer, comme la validation d'une note de frais, d'une commande… Depuis 2014, Community Cloud[59] permet de créer des plateformes communautaires personnalisées en ligne et d'optimiser la collaboration au sein des entreprises. Accessibles en mobilité, reliés aux réseaux sociaux et directement connectés aux

processus métiers, elles permettent de connecter directement les clients, partenaires et collaborateurs.

- Marketing cloud :

Lancée en 2012, Marketing Cloud est une plateforme marketing permettant d'établir des relations personnalisées avec les clients, à travers tous les canaux, et en mobilité : e-mails, web, réseaux sociaux, objets connectés.

Agrégeant les fonctionnalités de Radian 6 (écoute et captation d'informations), Buddy media (publication de contenus), Exact Target (marketing multicanaux et parcours client) et social.com (publicité), la plateforme permet d'organiser et de suivre les campagnes sur tous les canaux. Elle permet également d'extraire et d'exploiter l'ensemble des données recueillies lors des interactions avec les clients, pour leur proposer du contenu d'autant plus ciblé, personnalisé et en adéquation avec leurs besoins.

- Analytics Cloud

Analytics Cloud est la 1re plateforme cloud analytique, reposant sur la technologie Wave[15]. Elle est accessible depuis ordinateurs, smartphones et tablettes. La plateforme permet aux entreprises d'analyser leurs données, de partager les analyses entre collaborateurs et d'en tirer les informations nécessaires, afin d'identifier des opportunités commerciales.

1.7 Salesforce et le développement d'applications:

Salesforce donne la possibilité de développer des applications basées entièrement dans le cloud. Il existe deux façons de les développer dans la plateforme : en faisant du point and click ou en utilisant force.com.

Avec le point and click Salesforce nous permet de développer des applications de manière déclarative. Ses différents outils ne nécessitent pas de connaissances approfondies sur les principes de développement.

Exemple : Le générateur d'application Lightning et le process builder.

Pour le développement avec du code, Salesforce met à notre disposition un ensemble d'outils pour encourager une communauté de développeurs et de clients autour de ses solutions.

❖ La base de données force.com

La base de données fait partie des fondations de l'infrastructure Force.com. Elle nous permet de :

[15] https://doc.eedomus.com/view/La_technologie_radio_Z-Wave

Créer des objets : tels que des tables relationnelles et d'utiliser les métadonnées pour décrire ces objets et leur utilisation.

Créer des relations entre des objets de données : elles seront automatiquement implémentées dans les applications de Force.com, comme des « lookups » (parents et des listes d'objets enfants reliés), et des « master-details » (relations plusieurs à plusieurs dont la résultante est une table, dont la clé est celles des tables participantes à la relation)

Assurer l'intégrité des données : vous pouvez spécifier des règles de validation et utiliser des formules pour dériver logiquement de nouvelles valeurs. Auditer automatiquement les changements de votre base.

- ❖ La console du développeur Dev console

Le developer Console est un environnement de développement intégré avec une collection d'outils que nous pouvons utiliser pour créer, déboguer et tester des applications dans notre organisation Salesforce.

Pour ouvrir la console du développeur, cliquez sur nom d'utilisateur | Developer Console ou naviguez jusqu'à Setup| Develop | Tools et cliquez sur le lien de la console du développeur. La console du développeur s'ouvre dans une fenêtre distincte. L'interface utilisateur est organisée dans les sections suivantes :

Une barre de menu.

Un espace de travail avec un onglet pour chaque élément ouvert.

Un panneau avec les onglets Logs, Problems et Progress pour afficher les données d'exécution en temps réel.

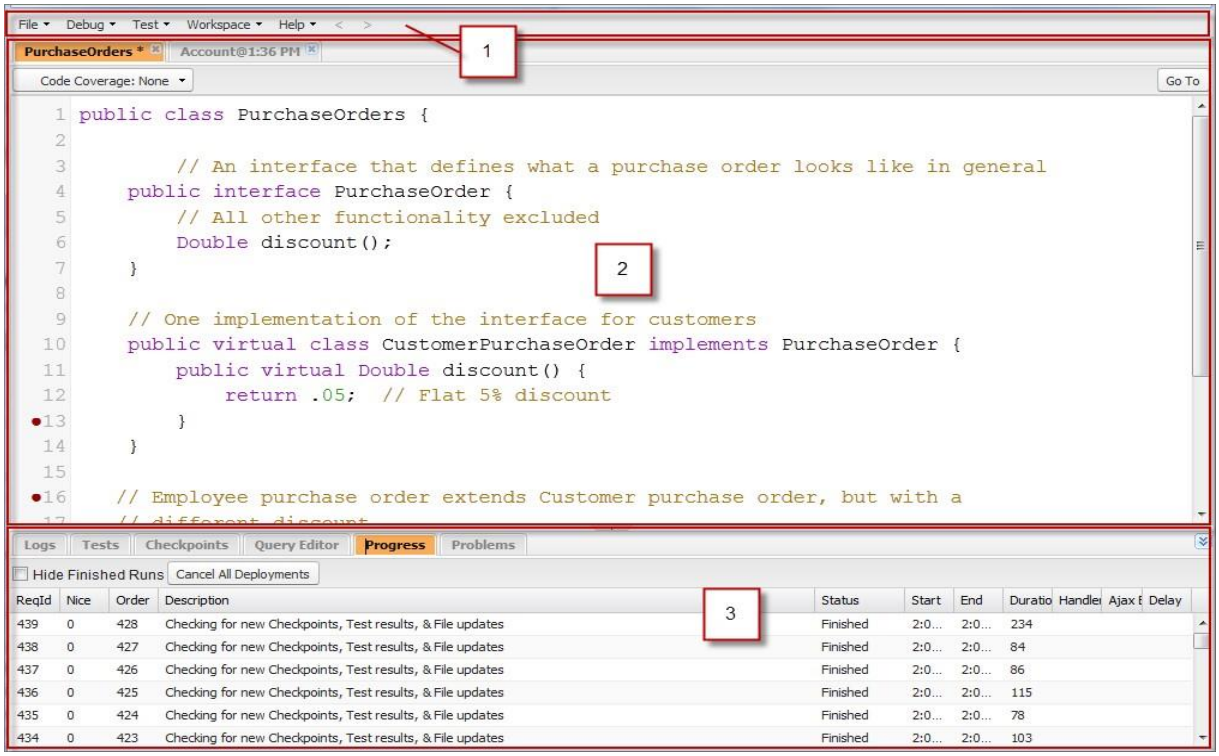


Figure 9 Développeur Console

❖ Le langage Apex

Apex est un langage de programmation orienté objet, fortement typé, qui permet aux développeurs d'exécuter des instructions de contrôle de flux et de transactions sur le serveur de la plateforme Force.com, en conjonction avec des appels à l'APIForce.com. Avec une syntaxe semblable à Java et qui se comporte comme les procédures stockées dans une base de données, le langage Apex permet aux développeurs d'ajouter une logique applicative à la plupart des événements système, y compris aux clics de bouton, aux mises à jour d'enregistrements associés et aux pages Visualforce. Le code Apex peut être initialisé par des demandes émanant de services Web et de déclencheurs d'objets.

En tant que langage, Apex est : **Intégré**

Apex fournit une prise en charge intégrée pour des idiomes communs de la plateforme Force.com qui comprennent :

Les appels DML[16] tels que INSERT, UPDATE et DELETE qui incluent le traitement DML Exception intégré.

Des requêtes SOQL[17] Salesforce et SOSL[18] Salesforce qui renvoient des listes d'enregistrements SObject

[16] Data Manipulation Language : http://www.gaudry.be/sql-syntaxe-dml-intro.html
[17] Salesforce Object Query Language: https://www.tutorialspoint.com/apex/apex_soql.htm
[18] Salesforce Object Search Language : https://www.tutorialspoint.com/apex/apex_sosl.htm

Des boucles qui permettent de traiter en masse plusieurs enregistrements même temps
Une syntaxe de verrouillage qui empêche les conflits de mise à jour d'enregistrements.
Des appels API Force.com publics personnalisés qui peuvent être créés à partir de méthodes Apex stockées
Des avertissements et des erreurs émis lorsqu'un utilisateur tente de modifier ou de supprimer un objet personnalisé ou un champ qui est référencé par un code Apex.

- **Facilité d'utilisation**

Le langage Apex est basé sur des idiomes Java familiers, tels qu'une syntaxe de variables et d'expressions, une syntaxe de déclaration de bloc et conditionnelle, une syntaxe de boucle, une notation d'objet et de tableau, etc. Lorsque le langage Apex introduit de nouveaux éléments, il utilise une syntaxe et une sémantique faciles à comprendre et favorise une utilisation efficace de la plate-forme Force.com. Par conséquent, le langage Apex produit un code à la fois succinct et facile à écrire.

- **Orienté vers les données**

Le langage Apex est conçu pour relier entre elles plusieurs requêtes et déclarations DML dans une seule unité de travail sur le serveur de la plate-forme Force.com, de la même façon que les développeurs utilisent des procédures stockées dans une base de données pour relier entre elles plusieurs déclarations de transaction sur un serveur de base de données. Notez que comme d'autres procédures stockées dans une base de données, le langage Apex n'essaie pas de fournir un support général pour le rendu des éléments dans l'interface utilisateur.

- **Rigoureux**

L'Apex est un langage fortement typé qui utilise des références directes à des objets de schéma, tels que des noms d'objet et de champ. Il échoue rapidement lors de la compilation si des références ne sont pas valides, et stocke toutes les dépendances de champ, d'objet et de classe personnalisés dans des métadonnées afin de les protéger contre la suppression lorsqu'elles sont demandées par un code Apex actif.

- **Hébergé**

Le langage Apex est entièrement interprété, exécuté et contrôlé par la plate-forme Force.com.

- **Mutualisé**

Comme le reste de la plate-forme Force.com, le langage Apex est exécuté dans un environnement mutualisé (multitenant). Par conséquent, le moteur d'exécution du langage Apex est conçu pour

empêcher qu'un emballement de code ne monopolise des ressources partagées. Tout code qui viole ces limites échoue avec des messages d'erreur faciles à comprendre.

- **Automatiquement mis à niveau**

Il n'est jamais nécessaire de réécrire le code Apex lorsque d'autres parties de la plateforme Force.com sont mises à niveau. Le code étant compilé et stocké sous forme de métadonnées sur la plate-forme, il est toujours automatiquement mis à niveau avec le reste du système.

- **Facile à tester**

Le langage Apex fournit une prise en charge intégrée de la création et de l'exécution de tests unitaires, qui comprennent des résultats de tests indiquant la quantité de code couverte et les parties de votre code qui peuvent être améliorées. Salesforce.com garantit la fiabilité du fonctionnement du code Apex, en exécutant des tests unitaires stockés dans des métadonnées avant toute mise à niveau de la plate-forme.

- **Versions multiples**

Vous pouvez enregistrer votre code Apex dans plusieurs versions API Force.com. Vous pouvez ainsi préserver des comportements.

Le langage Apex est inclus dans les versions Unlimited Edition, Developer Edition, Enterprise Edition et Database.com.

❖ **Le Framework Lightning**

Présentation du Framework

Lightning comprend le Lightning Component Framework et quelques outils passionnants pour les développeurs. Lightning facilite la création d'applications réactives pour n'importe quel appareil.

A la différence des autres, Framework Lightning dispose de deux contrôleurs : un contrôleur coté client et un contrôleur coté serveur.

Comparé à la programmation classique de Salesforce avec les pages visualforce, Lightning a des avantages incluant un ensemble de composants exclusifs, une architecture événementielle et un cadre optimisé pour les performances.

✓ **Performance**

Utilise une architecture de serveurs qui s'appuie sur JavaScript du côté client pour gérer les métadonnées et les données d'application des composants graphiques. Le client n'appelle le serveur que lorsqu'il est absolument nécessaire ; par exemple pour obtenir plus de métadonnées ou de données. Le serveur envoie uniquement les données nécessaires à l'utilisateur pour maximiser l'efficacité. Le Framework utilise JSON[19] pour échanger des données entre le serveur et le client.

[19] Javascript object Notation

Il utilise intelligemment notre serveur, notre navigateur, nos périphériques et notre réseau afin que nous puissions nous concentrer sur la logique et les interactions de nos applications.

- ✓ **Architecture événementielle**

Utilise une architecture événementielle pour un meilleur découplage entre les composants. Tout composant peut s'abonner à un événement d'application, ou à un événement de composant qu'il peut voir.

- ✓ **Développement plus rapide**

Permet aux équipes de travailler plus rapidement avec des composants prêts à fonctionner parfaitement avec les ordinateurs de bureau et les appareils mobiles. La création d'une application avec des composants facilite la conception parallèle, ce qui améliore l'efficacité globale du développement.

Les composants sont encapsulés et leurs contenus restent privés, alors que leur forme publique est visible pour les utilisateurs du composant. Cette séparation forte confère aux auteurs de composants la liberté de modifier les détails internes de mise en œuvre et d'isoler les consommateurs du composant de ces changements.

- ✓ **Compatible avec le périphérique et tous les navigateurs**

Les applications utilisent un design réactif et offrent une expérience utilisateur agréable. Le cadre Lightning Component prend en charge les dernières technologies de navigation telles que HTML[20]5, CSS[21]3 et les événements tactiles.

Architecture du framework

[20] Hypertext markup language
[21] Cascading Style Sheets

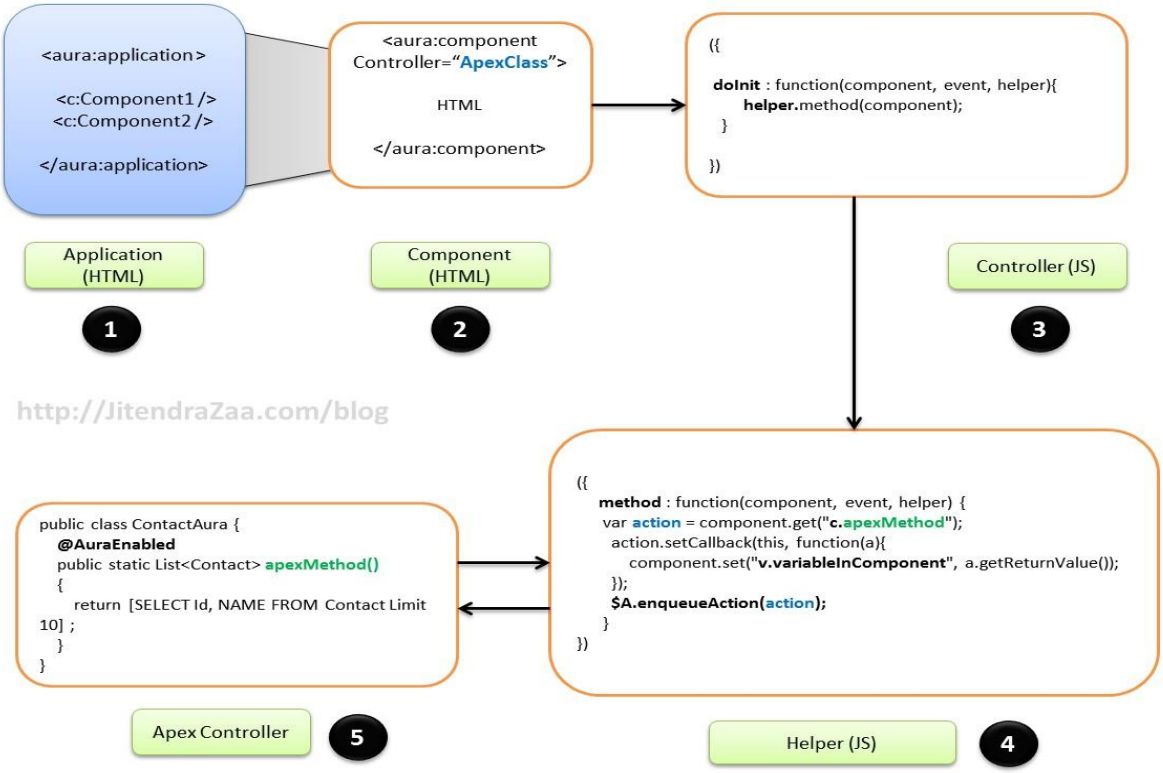


Figure 10 Architecture du Framework

Aura Application

Une application est un composant spécial de haut niveau dont le balisage se trouve dans une ressource avec l'extension .app.

Le balisage ressemble à du HTML et peut contenir des composants ainsi qu'un ensemble de balises HTML prises en charge. La ressource .app est un point d'entrée autonome pour l'application et nous permet de définir la présentation globale de l'application, les feuilles de style et les fonctionnalités JavaScript globales. Il commence par la balise <aura : application> de niveau supérieur, qui contient des attributs optionnels du système. Ces attributs système indiquent au Framework comment configurer l'application.

Aura Component

Les composants sont les unités fonctionnelles d'Aura, qui encapsulent des sections modulaires et réutilisables de l'interface utilisateur. Ils peuvent contenir d'autres composants ou un balisage HTML. Les parties publiques d'un composant sont ses attributs et leurs événements. Aura fourni des composants exclusifs dans les espaces de noms de l'aura et de l'interface utilisateur.

Le Contrôleur JS

Le contrôleur Client représente un fichier JavaScript directement associé au composant. Ce contrôleur comporte l'ensemble des fonctions JavaScript utilisées dans le composant aura. Ces méthodes s'exécutent à la suite d'un déclencheur et accèdent aux méthodes du contrôleur serveur.

Le Helper

Le Helper représente une extension du contrôleur client pour proposer des fonctions partagées.

Le contrôleur serveur

Le contrôleur serveur représente l'ensemble des classes écrites dans le langage apex.

Ce contrôleur représente la couche DAO ayant accès à la base de données et l'ensemble du processus métier du backend.

❖ **Lightning et les évènements**

Le framework utilise la programmation événementielle. Des gestionnaires sont écrits pour répondre aux événements de l'interface lorsqu'ils se produisent. Les événements peuvent ou non être déclenchés par l'interaction de l'utilisateur.

Dans le cadre de Lightning Component, les événements sont déclenchés à partir d'actions de contrôleur JavaScript. Les événements peuvent contenir des attributs qui peuvent être définis avant que l'événement ne soit déclenché et qu'ils soient lus lorsque l'événement est traité.

Les événements sont déclarés par le tag aura:event d'événement dans une ressource dont l'extension est .evt, et ils peuvent être de l'un des deux types: composant ou application. **Événements de type composant**

Un événement de type composant est déclenché à partir d'une instance d'un composant. Il peut être traité par le composant qui a déclenché l'événement ou par un composant qui reçoit l'événement dans la hiérarchie de confinement.

Événements de type application

Les événements de type application suivent un modèle traditionnel de publication-abonnement. Il est déclenché à partir de l'instance d'un composant. Tous les composants qui fournissent un déclencheur pour l'événement sont notifiés.

❖ **Lightning et la programmation orientée objet**

Un composant Lightning peut désormais étendre un autre composant Lightning. Cette fonctionnalité exploite le concept d'héritage de la programmation orientée objet et l'applique au développement de la couche de présentation.

Un composant qui étend un autre composant reçoit les attributs et les événements enregistrés du super composant dont il hérite.

Ces attributs système sont désormais disponibles dans les balises <aura:component> et <aura:application>. ○ **Extensible**

Cet attribut est défini sur true si le composant ou l'application peut être étendu mais sa valeur par défaut est false.

○ **Extends**

On utilise cet attribut pour définir l'application ou le composant à étendre. ○

Abstract

On le défini sur true si le composant ou l'application est abstrait. Sa valeur par défaut est false. L'infrastructure de composants Lightning prend en charge le concept de composants abstraits qui ont une mise en œuvre partielle, mais laissent la mise en œuvre restante à des sous-composants concrets. Pour utiliser un composant abstrait, on doit l'étendre et effectuer la mise en œuvre restante. Un composant abstrait ne peut pas être directement utilisé dans le balisage.

Chapitre 4: Le projet

4.1 **Présentation du projet** :

Project Management est une grande application qui va gérer un projet de sa création à la gestion des dépenses et des finances. Elle comporte plusieurs branches comme la gestion des risques, la gestion des reportings des ressources etc. Notre partie travail s'intéresse à la gestion des finances et des dépenses.

Dans le cadre de l'amélioration de la gestion de projet, la gestion des finances et des dépenses se présente comme un aspect fondamental. Les dépenses constituent l'une des principales causes de l'échec d'un projet quelconque. Il est bien pratique de bien gérer les dépenses d'un projet pour ne pas dépasser le budget ce qui peut nuire à celui-ci.

4.2 **Etude de l'existant** :

Il existe déjà des applications de gestion des dépenses et des finances qui ont été développées sous Salesforce.

Dans la gestion de projet il est souvent difficile de suivre les dépenses de divertissement à l'échelle de l'entreprise et le rendement du capital investi à partir de ces dépenses. Il est également extrêmement difficile d'avoir une visibilité sur les employés qui utilisent ces billets malgré les clients ou prospects qui les ont utilisés.

Entertainment Expense management a été développé sous Salesforce. IL fournit une visibilité pour comprendre quels clients ou prospects ont tiré profit des récompenses de divertissement. Cette information peut aider à identifier les clients ou prospects qui ont mérité la récompense ou qui ont fourni suffisamment de valeur au client pour la justifier. Les approbations internes garantissent également que les employés utilisent ces récompenses d'une manière qui profite le plus à l'entreprise.

Salesforce.com EEM[22] ; disponible sur AppExchange est facile à utiliser et peut être personnalisé pour correspondre à vos besoins spécifiques dans la gestion de tout type de récompenses que votre entreprise peut avoir à distribuer à ses principaux clients et prospects.

[22] Entairtainement Expense management :
https://appexchangejp.salesforce.com/listingDetail?listingId=a0N300000016YJGEA2

Figure 11 Salesforce Labs

4.3 **Méthodologie adoptée** :

Dans le développement classique d'applications il existe différentes méthodologies de développement de logiciels qui nous permettent de gérer le développement d'un projet en différentes phases séquentielles jusqu'à la livraison du produit final.

Exemple de méthodes traditionnelles méthodes en cascade ou modèle en V.

Il existe aussi d'autres méthodes récentes comme les méthodes agiles.

Salesforce quant à lui propose comme processus de développement Application Life cycle Management. C'est ce processus qui a été utilisé tout le long de notre projet.

ALM[23] définit le processus de gestion du développement d'une application de la conception à la version finale. ALM crée également un cadre permettant de corriger les différents bugs d'application et d'améliorer les fonctionnalités au fil du temps. Nous distinguons six étapes au niveau du processus ALM :

- ✓ Planification de la version

On commence tout d'abord à planifier le projet ce qui consiste à collecter des exigences et de les analyser. On doit aussi déterminer les divers environnements de développement et de tests dont l'équipe a besoins au fur et à mesure que le projet avance dans le cycle ALM. ✓ Développement

On doit réaliser le travail en respectant les spécifications de conception, et ce, dans un environnement contenant une copie des métadonnées de l'organisation de production, mais dépourvu des données de production. ✓ Test

On teste les modifications apportées pour vérifier qu'elles fonctionnent comme prévu avant de les intégrer dans l'environnement de production.

- ✓ Compilation de la version

[23] Application lifecycle management : https://www.lebigdata.fr/alm-application-lifecycle-management

On regroupe tous les actifs qu'on a créés ou modifiés au cours de la phase de développement en un seul artefact, un ensemble logique de personnalisation qu'on a déployé en production. ✓ Test de la version

On teste dans un environnement de transfert qui produit autant que possible la production ce qu'on a réellement déployé pour plus de sécurité. ✓ Publication

Lorsque les tests sont terminés et qu'on atteint les critères de qualité, on peut déployer le projet en production.

Figure 12 ALM

4.4 **Organisation du projet** :

Notre module de gestion des risques appartient au grand projet : Project Management

Il a été développé au sein de l'entreprise par plusieurs personnes :

Acteurs	Module à Developper
Dr Mamadou Bousso	Responsable technique Chargé du suivi du projet
Ingénieur Moussa Fofana	Project Manager chargé de la définition des tâches
Abdoulaye Diallo	Gestion des risques
Moustapha Diouf	Reporting
Dieynaba Diallo	Gestion du planning et des collaboration
Stephane Ouédraogo	Gestion des activités et du timesheet
Youssouph Niaouba Diedhiou	Gestion de la Finance et des dépenses

Tableau 2: Les membres du projet

4.5 Analyse des besoins et spécification

4.5.1 Analyse des besoins

○ Les besoins attendus de l'application

L'application envisagée doit satisfaire les besoins fonctionnels qui seront exécutés par le système et les besoins non fonctionnels qui perfectionnent la qualité logicielle du système.

Les besoins fonctionnels :

Les besoins fonctionnels ou besoin métiers représentent les actions que le système doit exécuter, il ne devient opérationnel que s'il les satisfait. Cette application doit couvrir principalement les besoins fonctionnels suivants :

Gestion d'un projet

Gestion des dépenses d'un projet

Gestion des facturations d'un projet

Gestion des tâches d'un projet

Gestion de la planification et de l'ordonnancement d'un projet

Gestion de la collaboration d'un projet

Gestion des risques d’un projet

Gestion du contrôle d’un projet
Gestion de portefeuille d’un projet

Gestion d’équipe d’un projet Les

besoins non fonctionnels :

Ce sont des exigences qui ne concernent pas spécifiquement le comportement du système mais plutôt identifient des contraintes internes et externes du système. Les principaux besoins non fonctionnels de notre application se résument dans les points suivants : Le code doit être clair pour permettre des futures évolutions ou améliorations ;

L’ergonomie : l’application offre une interface conviviale et facile à utiliser ;

La sécurité : l’application doit respecter la confidentialité des données ;

Garantir l’intégrité et la cohérence des données à chaque mise à jour et à chaque insertion.

Résultats attendus

Que l’application se déroule convenablement, de conserver ces fonctionnalités dans l’application, et d’améliorer s’il est possible les performances du système ainsi que les bases des données existantes.

4.5.2 **Les diagrammes de cas d’utilisation**

Le diagramme des cas d'utilisation (Use Case Diagram) constitue la première étape de l’analyse UML en :

Modélisant les besoins des utilisateurs.

Identifiant les grandes fonctionnalités et les limites du système.

Représentant les interactions entre le système et ses utilisateurs.

Le diagramme des cas d’utilisation apporte une vision utilisateur et absolument pas une vision informatique. Il ne nécessite aucune connaissance informatique et l’idéal serait qu’il soit réalisé par le client.

Le diagramme des cas d’utilisation n’est pas un inventaire exhaustif de toutes les fonctions du système. Il ne liste que des fonctions générales essentielles et principales sans rentrer dans les détails.

Les différents éléments qui interviennent à la conception d’un diagramme des cas d’utilisation sont:

Acteur : c’est le rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d’utilisation.

Cas d'utilisation : c'est un ensemble de séquences d'actions réalisées par le système produisant un résultat observable intéressant pour un acteur particulier. Collection de scénarios reliés par un objectif utilisateur commun.

Association : elle est utilisée dans ce type de diagramme pour relier les acteurs et les cas d'utilisation par une relation qui signifie simplement " participe à ".

Inclusion : on le retrouve sous le nom de include dans la conception. C'est le cas d'utilisation de base qui incorpore explicitement un autre, de façon obligatoire, à un endroit spécifié dans ses enchaînements.

Extension : on le retrouve sous le nom de extends dans la conception. C'est le cas d'utilisation de base qui incorpore implicitement un autre, de façon optionnelle, à un endroit spécifié indirectement dans celui qui procède à l'extension.

Généralisation : C'est les cas d'utilisation descendants qui héritent de la description de leur parent commun. Chacun d'entre eux peut néanmoins comprendre des relations spécifiques supplémentaires avec d'autres acteurs ou cas d'utilisation.

Identification des acteurs

Un acteur représente un rôle joué par une personne qui interagit avec le système.

Par définition, les acteurs sont à l'extérieur du système. Les acteurs se recrutent parmi les utilisateurs du système et aussi parmi les responsables de sa configuration et de sa maintenance.

D'où, les acteurs potentiels qui risquent d'interagir avec l'application sont :

Developer (Développeur) : C'est l'élément central de notre système. Il est chargé d'effectuer toutes les tâches clientes. Ces tâches clientes ne sont rien d'autre que la relation entre l'utilisateur et les cas d'utilisation définis dans le schéma 6 ci-dessous.

Project Manager (Chef de projet) : Il est chargé de créer des entreprises, des emplois et d'avoir à sa disposition tous les profils des utilisateurs.

Financial (Directeur Financier) : Il est chargé d'effectuer tous les tâches métiers avec l'ajout, la modification, la récupération et la suppression de fonctionnalités.

Les différents cas d'utilisation

L'étude de cas d'utilisation a pour objectif de déterminer ce que chaque utilisateur attend du système. La détermination du besoin est basée sur la représentation de l'interaction entre l'acteur et le système.

Ainsi le schéma 6 ci-dessous définit aussi les cas d'utilisation préliminaires du système de gestion des dépenses. Ces cas d'utilisations sont :

Authentification : permet d'identifier chaque utilisateur, et de lui donner l'accès aux fonctionnalités qui lui sont destinées.

Create Expense : permet aux utilisateurs qui ont accès à cette fonctionnalité de créer une nouvelle dépense.

Update Expense : permet aux utilisateurs qui ont accès à cette fonctionnalité de modifier une dépense.

Delete Expense : permet aux utilisateurs qui ont accès à cette fonctionnalité de supprimer une dépense.

Submit expense request for approbation : Chaque dépense doit être envoyée au chef de projet (Project Manager) pour une validation. Ainsi ce cas d'utilisation permet aux utilisateurs qui ont accès à cette fonctionnalité de soumettre une nouvelle dépense pour une approbation.

Validate Expense : permet à l'utilisateur (Chef de projet) qui a accès à cette fonctionnalité de valider une nouvelle dépense.

List expense for a project during a period of time : permet à l'utilisateur (Directeur Financier) qui a accès à cette fonctionnalité de valider une nouvelle dépense.

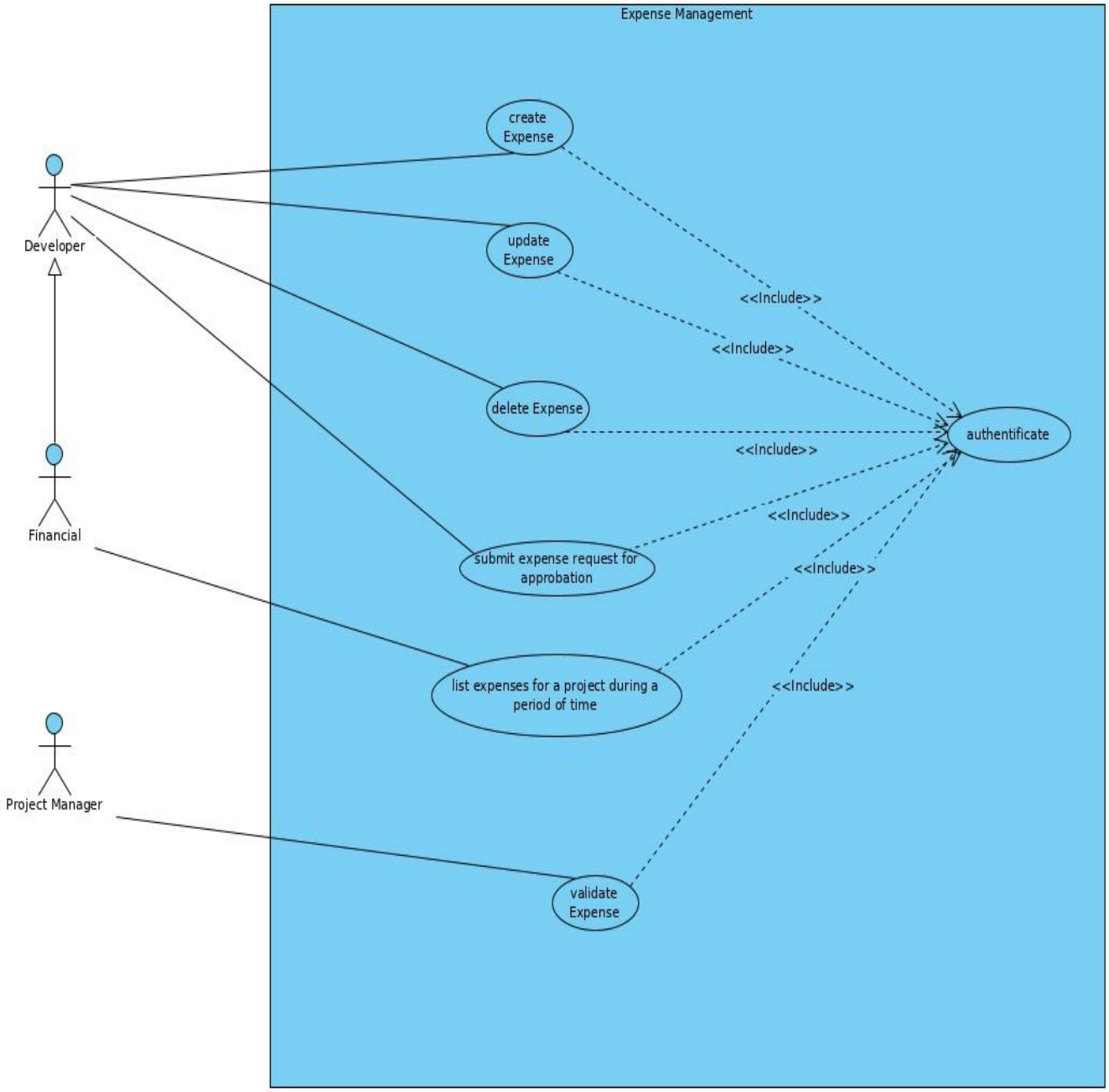


Figure 13 Uses Cases

Description des cas d'utilisation

Nous allons faire une description de quelques cas d'utilisation en utilisant les diagrammes de séquence. A cet effet nous allons essayer de définir c'est un diagramme de séquence. Pour ce faire nous allons essayer de répondre à cette question :

C'est quoi un diagramme de séquence ?

A l'instar des diagrammes de cas d'utilisation qui modélisent à QUOI sert le système, en organisant les interactions possibles avec les acteurs. Et les diagrammes de classes qui permettent de spécifier la structure et les liens entre les objets dont le système est composé : il spécifie QUI sera à l'œuvre dans le système pour réaliser les fonctionnalités décrites par les diagrammes de cas d'utilisation. Les diagrammes de séquences permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs :

Les objets au cœur d'un système interagissent en s'échangent des messages.

Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine)

Ainsi pour mener à bien notre travail nous allons commencer par la description de l'authentification qui est primordiale pour tout système sécurisé qui est représentée par le schéma 1 (AuthentificationUseCase).

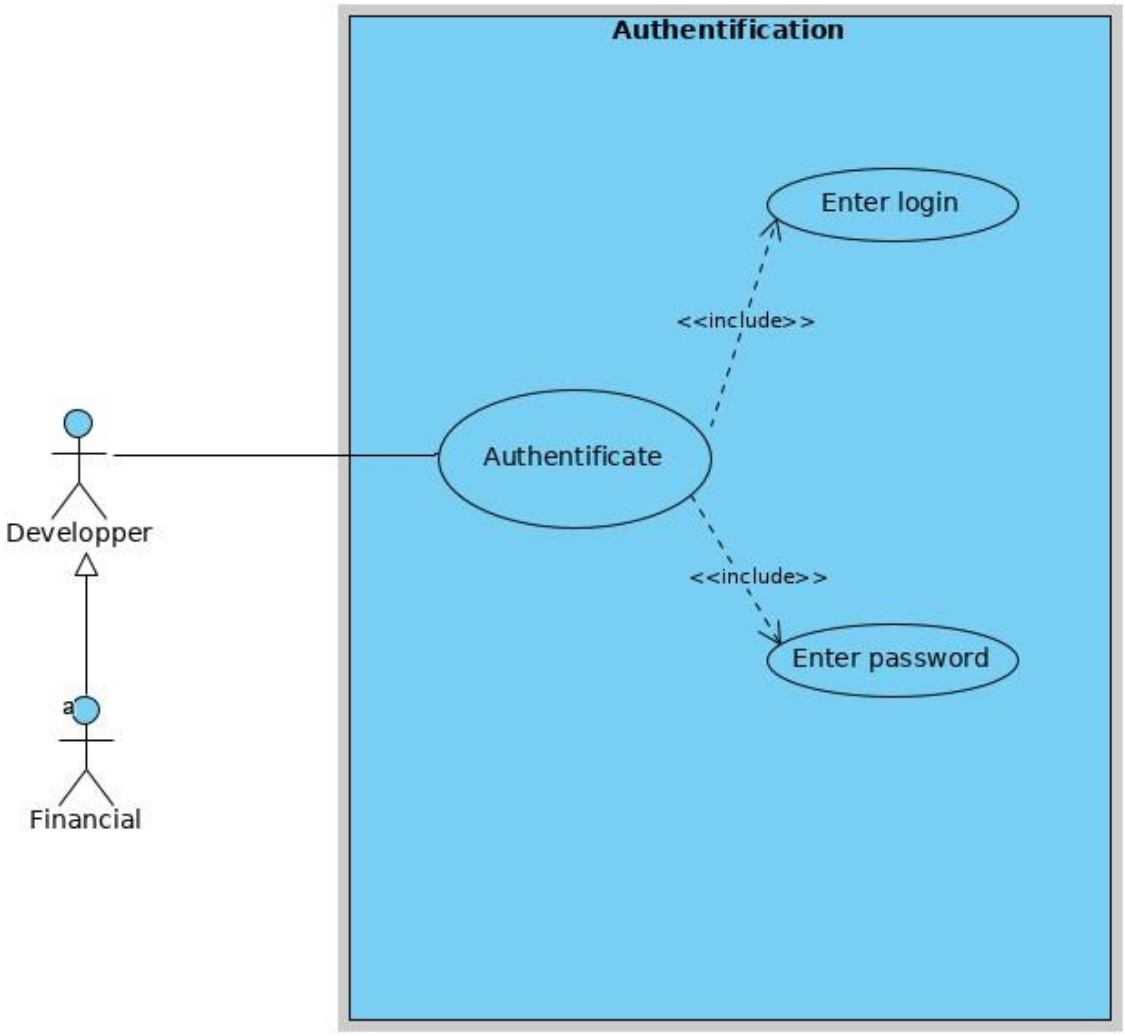


Figure 14 Authentification Use Case

Authentificate : Nous allons faire la description textuelle de ce cas d'utilisation qui permet de se connecter au système dans le but d'effectuer certaines tâches.

Description textuelle

Nom **:** Authentification

Acteur **:** Effectuer des opérations sécurisées (ajout, suppression, chargement, …)

Description **:** Chaque utilisateur doit s'authentifier pour effectuer certaines tâches dans le système

Date **:** 08/09/2019

Auteur **:** Youssouph Niaouba Diedhiou

Précondition **:** l'utilisateur doit être enregistré dans la base

Démarrage **:** un utilisateur souhaite effectuer une opération

Description des scénarios

Scénario nominal

1. Le système affiche le formulaire d'authentification
2. L'utilisateur entre son login et son mot de passe
3. L'utilisateur valide
4. Le système vérifie si les informations entrées sont correctes
5. Le système redirige l'utilisateur à la page de son profil

Scénario alternatif

1.a) l'utilisateur quitte l'application (étapes : 2.a), 3.a), 4.a), 5.a))

Il appui sur le bouton de fermeture de la page

Le système demande une confirmation

Il confirme

Scénario d'exception

4.b) Les informations saisies sont incorrectes

Le système affiche un message d'erreur

Le système demande à l'utilisateur de rectifier

Retour à l'étape 2

Fin

Étapes 1, 2, 3, 4, 5 sur décision de l'utilisateur

Post condition

L’utilisateur est connecté

Create Expense : Nous allons présenter ce cas d’utilisation qui est la création d’une nouvelle dépense d’un projet. **Description textuelle**

Nom : Chargement de la base des dépenses (ajouter dépenses)

Acteur **:** Développeur

Description : l’utilisateur ajoute des dépenses.

Date **:** 08/09/2019

Auteur **:** Youssouph Niaouba Diedhiou

Précondition **:** être connecté en tant développeur ou directeur financier

Démarrage **:** un projet doit être ajouté dans la base

Description des scénarios

Scénario nominal

Le système affiche le formulaire de saisi

L’utilisateur saisit les informations personnelles de chaque dépense

L’utilisateur valide la saisie

Le système vérifie si les informations entrées sont correctes Le

système demande une confirmation

L’utilisateur confirme

Le système alloue de l’espace

Scénario alternatif

1.a) Annulation de la saisie d’une dépense (étapes : 2.a), 3.a), 4.a))

L’utilisateur appui sur le bouton de fermeture ou sur le bouton annuler du formulaire

Le système demande une confirmation d’annulation

L’utilisateur confirme

Le système se ferme

5.a) L’utilisateur ne confirme pas la validation (étapes : 6.a), 7.a))

L’utilisateur appui sur le bouton annuler de la boîte de dialogue 6.b)

L’utilisateur quitte sans confirme (étapes : 7.b))

Scénario d’exception

4.b) Les informations saisies ne sont pas correctes

Le système envoie un message d'erreur Le système demande à l'utilisateur de rectifier

Retour à l'étape 2.

7.c) Espace insuffisant pour enregistrer la saisie

Fin

Étapes 1, 2, 3, 4, 5, 6 sur décision de l'utilisateur

Étape 7 si l'espace de stockage est insuffisant **Post condition**

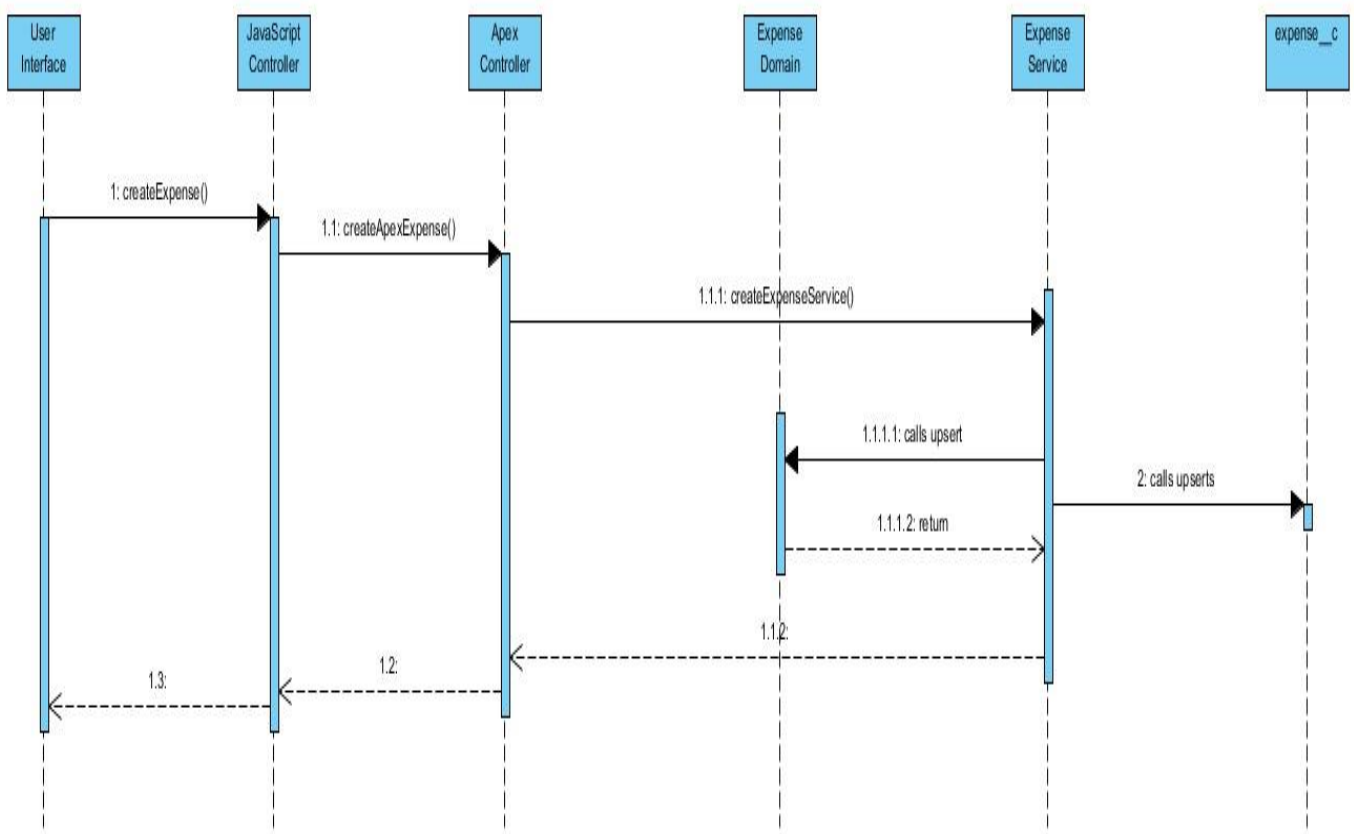


Figure 15 Création d'une dépense

Chaque dépense d'un projet est dans la base.

Nom : Modification d'une dépense

Acteur : Développeur

Description : le développeur modifie une dépense.

Date : 08/09/2019

Auteur : Youssouph Niaouba Diedhiou

Précondition : être administrateur

Démarrage : une dépense existante doit être modifiée

Description des scénarios

Scénario nominal

1. Le système affiche la liste des utilisateurs
2. Le développeur sélectionne une dépense à modifier
3. Il appui sur le bouton modifier de la liste
4. Le système affiche le formulaire avec les données concernant la dépense choisie
5. Le développeur saisit les modifications
6. Il valide la modification
7. Le système vérifie si la saisie est correcte
8. Le système demande une confirmation
9. Le développeur confirme la modification
10. Le système enregistre les modifications dans la base.

Scénario alternatif

1.a) l'administrateur souhaite annuler la modification d'utilisateur (étapes : 2.a), 3.a), 4.a), 5.a), 6.a), 7.a))

Le développeur appui sur le bouton de fermeture de la liste

Le système demande une confirmation de fermeture

Il confirme

La liste est fermée

2.b) le développeur annule la sélection

Il appui sur le bouton annuler de la liste

8.a) le développeur ne confirme pas la validation (étapes : 9.a), 10.a))

L'administrateur appui sur le bouton annuler de la boîte de dialogue

Scénario d'exception

3.b) L'utilisateur ne peut pas être modifié pour le moment (mandat non terminé)

Le système envoie un message d'erreur

Le système demande à l'administrateur de rectifier la sélection Retour

à l'étape 2.

7.b) Les informations saisie ne sont pas correctes

Le système envoie un message d'erreur

Le système demande à l'administrateur de rectifier la saisie

Retour à l'étape 2.

Fin

Étapes 1, 2, 3, 4, 5, 6, 7 sur décision de l'utilisateur

Étape 10 après allocation d'espace

Post condition

L'utilisateur est modifié

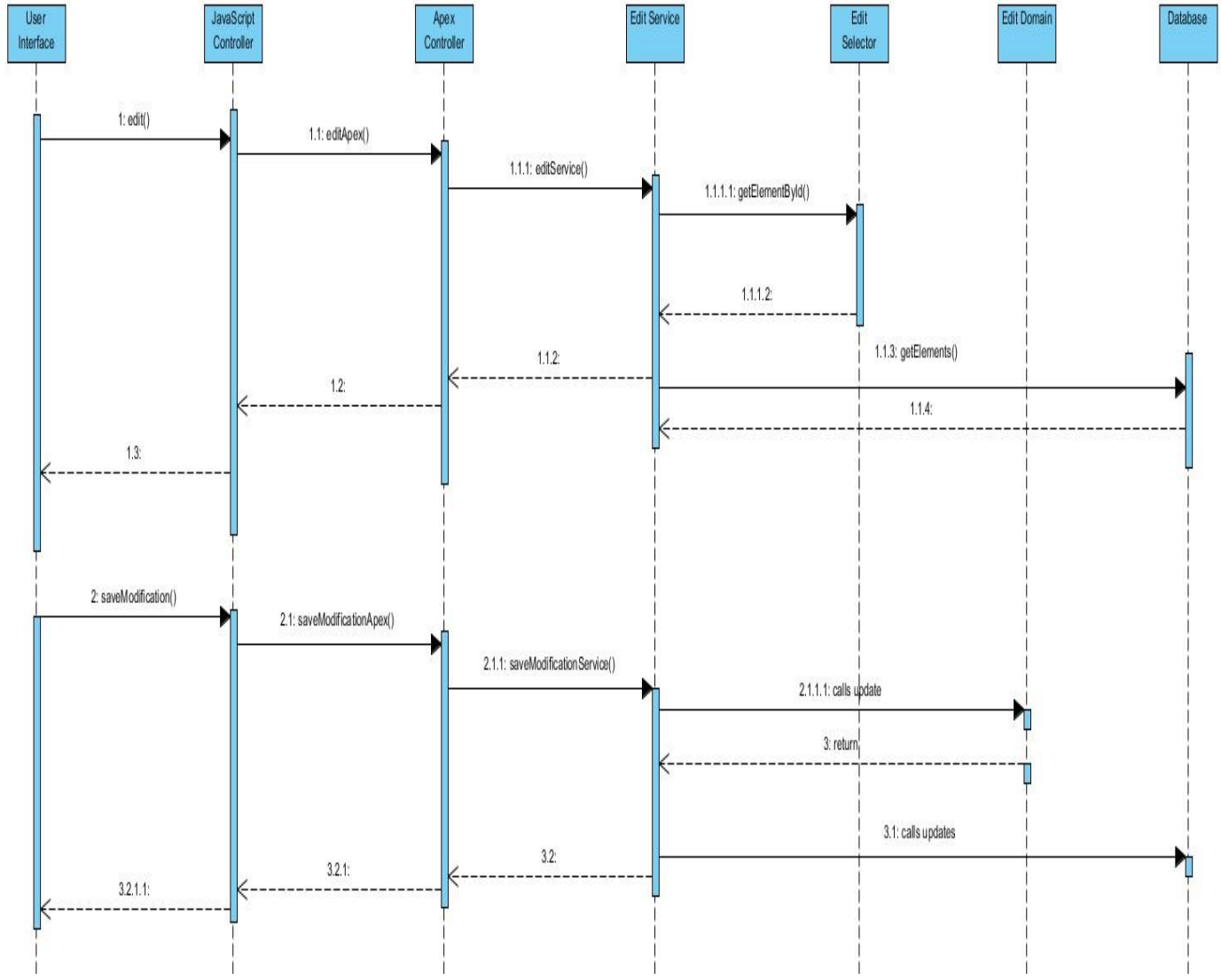


Figure 16 Modification d'une dépense

Delete Expense : Nous allons présenter ce cas d'utilisation qui est la suppression d'une dépense d'un projet.

Description textuelle

Nom : Suppression d'une dépense

Acteur : Développeur

Description : le développeur supprime une dépense de la base.

Date : 08/09/2019

Auteur : Youssouph Niaouba Diedhiou

Précondition : la dépense ne doit être soumise ni validée

Démarrage : une dépense doit être supprimé de la base

Description des scénarios

Scénario nominal

1. Le système affiche la liste des dépenses
2. Le développeur sélectionne la dépense à supprimer
3. Il appuie sur le bouton supprimer de la liste
4. Le système demande une confirmation
5. Il confirme la suppression
6. Le système libère de l'espace

Scénario alternatif

1.a) le développeur souhaite annuler la suppression de la dépense (étapes : 2.a), 3.a))

Le développeur appui sur le bouton de fermeture de la liste

Le système demande une confirmation de fermeture

Il confirme

La liste est fermée

2.b) le développeur annule la sélection

Il appui sur le bouton annuler de la liste

4.a) le développeur ne confirme pas la validation (étapes : 5.a), 6.a))

Le développeur appui sur le bouton annuler de la boîte de dialogue

5.b) le développeur quitte sans confirmé (étapes : 6.b))

Scénario d'exception

3.b) la dépense ne peut pas être supprimée pour le moment (elle est soumise à une approbation ou déjà validée)

Le système envoie un message d'erreur

Le système demande à l'administrateur de rectifier

Retour à l'étape 2. **Fin**

Étapes 1, 2, 3, 4, 5 sur décision de l'utilisateur

Étape 7 après libération d'espace

Post condition

L'utilisateur est supprimé de la base.

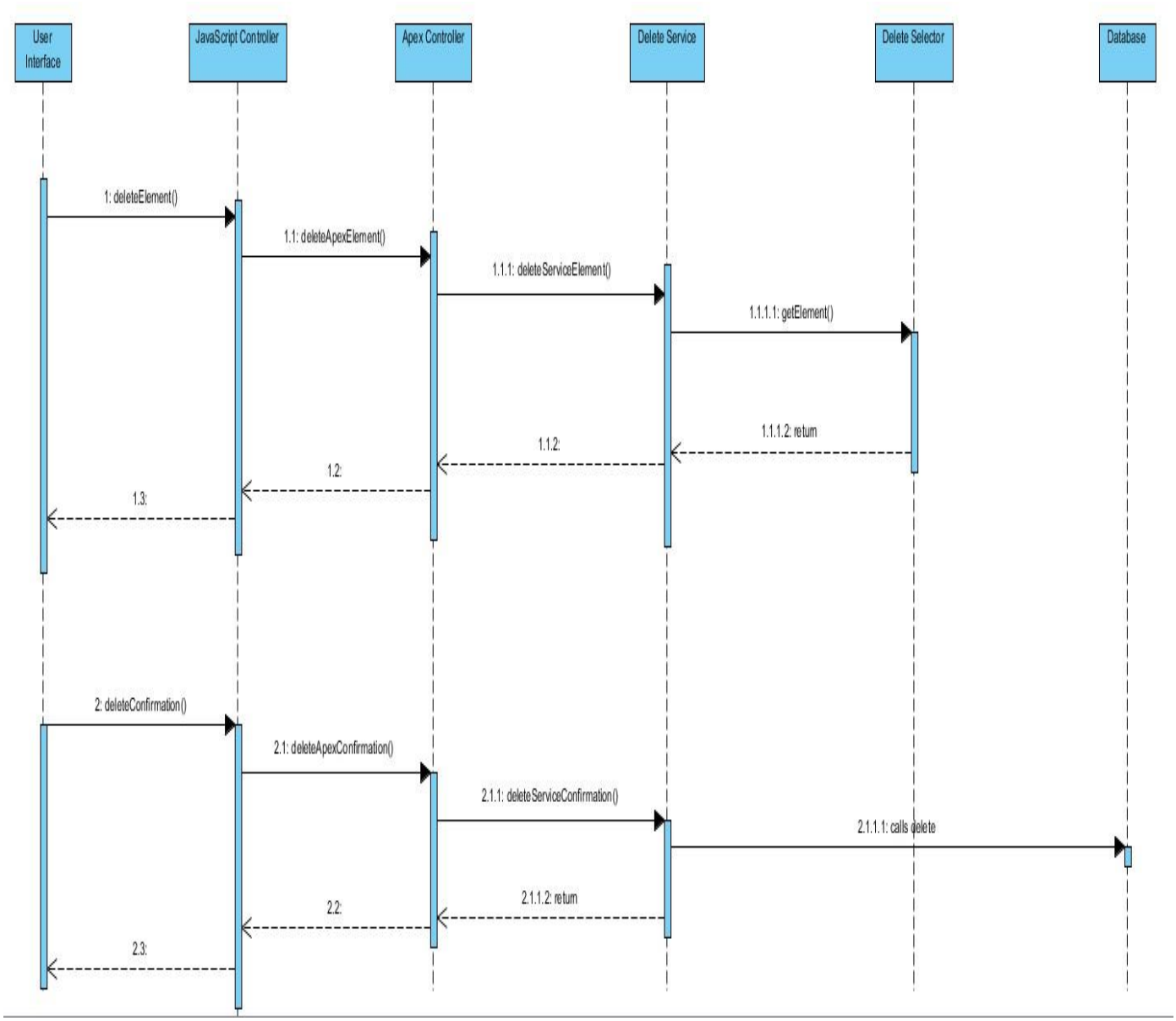


Figure 17 Suppression d'une dépense

Conclusion

Dans ce chapitre nous avons essayé de faire la spécification des besoins fonctionnels et non fonctionnels du futur système, ce qui correspondait aux différentes activités de la première phase du cycle de développement du notre système. Dans le chapitre suivant, nous étudierons la phase de conception.

Chapitre 5: L'architecture

5.1 L'architecture SOC[24] en relation avec notre projet :

La séparation des préoccupations est un modèle ou un principe de conception d'architecture logicielle permettant de séparer une application en sections distinctes, de sorte que chaque section aborde une préoccupation distincte. Elle a pour objectif principal la séparation des différentes préoccupations est d'établir un système bien organisé où chaque partie remplit un rôle significatif. Lorsque des modifications sont réalisées, les impacts et les régressions sur d'autres zones sont réduits et des programmes peuvent évoluer facilement. Notre projet a été développé en utilisant la séparation des préoccupations. De ce fait on différentes couches avec chacun un rôle bien défini.

5.2 Les différentes couches du projet :

Dans le cadre de notre projet différentes couches ont été utilisées : Domaine Service Selector.

Chaque couche va effectuer un travail lorsque le controlleur apex fait un appel.

- **La couche service :**

La couche service elle est un moyen d'encapsuler les processus métier de notre application. Un processus métier est une opération qui peut déclencher un processus.

Exemple de processus métier : Click d'un bouton pour enregistrer ou modifier une dépense. La couche service permet aussi de réutiliser plusieurs fois le code relatif à un objet quelconque. La couche service en cas de besoin peut faire aussi appel à la couche Selector ou à la couche Domaine.

Exemple : Quand on veut enregistrer une nouvelle dépense la couche service peut appeler directement la couche selector pour enregistrer au niveau de la base ou appeler la couche domaine pour faire une vérification de la sémantique avant d'enregistrer.

- **La couche Domaine :**

La couche service a pour rôle de fournir un niveau plus précis d'encapsulation et de réutilisation de code. Elle gère aussi la validation complexe, le retour aux valeurs par défaut ainsi que d'autres logiques liées au calcul et à la manipulation complexe des données.

C'est à ce niveau que la vérification de la sémantique va se faire. Cependant après la vérification la couche domaine peut faire appel fait appel à la couche service pour une autre opération.

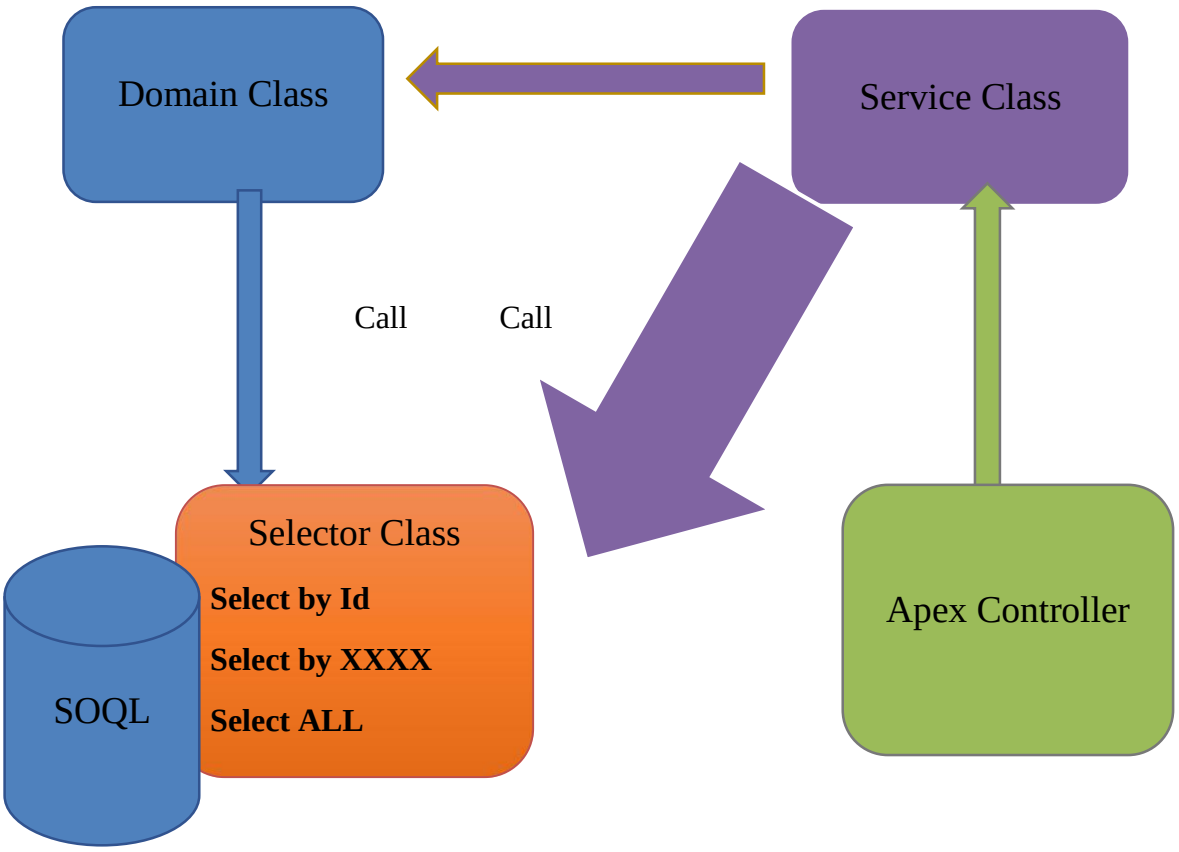
- **La couche Selector :**

Elle contient la logique chargée de demander des informations à nos objets personnalisés. Elle peut être utilisée par n'importe qu'elle autre classe (domaine, service contrôleur apex …).

Au niveau de la couche Selector on y stocke tout ce qu'il y'a comme code qui permet de faire des requêtes de sélection au niveau de la base.

[24] Separation of concern: https://medium.com/@rkay301/programming-fundamentals-part-5-separation-ofconcerns-software-architecture-f04a900a7c50

Exemple : Récupérer une dépense à partir de son identifiant.



5.3 Liste des différentes classes domaine et service:

Dans le cadre de notre projet différents objets ont été utilisés et pour chaque objet il existe des champs pour stocker différentes informations.

Les différents SObjects utilisés sont :

Account, Contact, Contract, Case et Asset

Pour l'utilisation de ces SObjects nous avons utilise les types d'enregistrement (RecordType).
En effet, les types d'enregistrement déterminent les processus, les mises en page et les valeurs de liste de selection que les agents utilisent. C'est ainsi que nous avons créé des types d'enregistrement pour être en conformité avec notre modèle.

Champs des différents SObject et rôle :

Project est un type d'enregistrement de l'objet Account

Invoice est un type d'enregistrement de l'objet Contract

Member Has Projecct est un type d'enregistrement de l'objet Contract

Expense est un type d'enregistrement de l'objet Case.

Champs	Type	Contraintes	Signification
invoice_nr	number	Elle doit être unique	C'est le numéro de la facture du client
invoice_date	Date	Cette date doit être différente de celle du délai	Elle correspond à la date de la création de cette dernière
invoice_paymentTerms	varchar	La date de paiement ne doit pas dépasser le délai fixé	Le délai fixé pour le paiement de la facture client
invoice_paymentMethod	varchar	Sa valeur doit être un élément de la liste	C'est le mode de paiement utilisé pour la facture
invoice_status	varchar	Sa valeur doit être un des différents états de la facture	L'état de la facture client
invoice_type	varchar	Sa valeur doit être un des différents types de facture	Le type de facture choisi

Tableau 3: Les caractéristiques de l'objet Invoice

Expense__c

Champs	Type	Contraintes	Signification
projectExp	number	Le projet doit exister	le projet qui est en rapport avec la dépense
amountExp	number		Le montant de la dépense
currencyExp	varchar	Sa valeur doit être une des différentes monnaies de la liste	La monnaie utilisée
vat_inclExp	varchar		Si la taxe est inclue ou pas
dateExp	varchar		La date de la dépense
Receipt_refExp	number	Elle doit être unique	La référence de la dépense associée à cette dernière
typeExp	varchar	Sa valeur doit être un des différents types de dépense	Le type de dépense choisi
billableExp	varchar	Sa valeur doit impérativement être oui ou non	Il permet de dire si la dépense est facturable ou non
reimbursableExp	number	Ce montant ne doit pas être supérieur au montant de la dépense	C'est quand un employé fait une dépense avec son propre argent.
paymentExp	varchar	Sa valeur doit être un des différents modes de paiement	Le mode de paiement utilisé

noteExp	varchar		La note ajoutée à la dépense

Tableau 4: Les caractéristiques de l'objet Expense

Pour chaque sObject on aura besoin d'utiliser les classes :

Service : définit les frontières d'une application par une couche de services qui établit un ensemble d'opérations disponibles et coordonne la réponse de l'application dans chaque opération. **Expense Service listerExp :** permet de récupérer les champs des dépenses nécessaires à l'affiche de la liste.

calculerExp : permet de faire la somme des dépenses.

enregistrerExp : permet l'ajout d'une nouvelle liste de dépenses.

convertirExp : permet de convertir une dépense en facture **dupliquerExp**

: permet de dupliquer une dépense

Invoice Service

Emettre : permet de soumettre une facture

Payer : permet de régler une facture

Annuler : permet d'annuler une facture

Exporter : permet

Enregistrer : permet l'ajout d'une facture

Importer : permet d'importer une liste de dépenses

Domain : On entend par domaine un champ d'étude définissant un ensemble d'exigences, une terminologie et des fonctionnalités communes à tous les programmes logiciels construits pour résoudre un problème dans le secteur de la programmation informatique, connu sous le nom de génie de domaine

Selector : une couche de code qui encapsule la logique chargée de demander des informations à vos objets personnalisés et de les insérer dans votre code de couche Domaine et de couche Service.

5.4 La modélisation:

5.4.1 Différence entre la modélisation classique et la modélisation sous Salesforce:

Dans la modélisation classique plusieurs logiciels sont utilisés pour représenter les différentes classes et leurs relations. Dans Salesforce il existe déjà des objets natifs prêt à être utilisés.

Il existe deux objets sous Salesforce :

Les objets Standards :

Les objets standards (standards objects) sont les objets inclus dans Salesforce (on les appelle aussi les objets métiers).

Exemple : Account, Contact, Opportunity, Campaign,Task etc.

Les Objets personnalisés :
Un objet personnalisé (Custom Object) est un objet qui n'est pas inclus dans Salesforce. C'est un objet qu'on crée et qui est spécifique à un secteur d'activité. Comme par exemple on crée l'objet Expense pour stocker une dépense.

Le processus de modélisation du projet :

Notion d'objet et de données dans Salesforce :

Dans Salesforce tout est métadonnées donc un objet crée est un fichier XML[25]. Cependant quand on manipule un objet Salesforce c'est comme si on manipule un fichier XML.

Pour représenter les objets de Salesforce et leur relations (modéliser) on utilise le Schéma Builder.

5.4.2 Les différentes étapes du processus de modélisation de notre projet :

Dans le cadre de notre projet différentes étapes ont été utilisées en partant du diagramme de classe de base jusqu'à l'obtention et à la stabilisation d'un modèle.

Le processus de modélisation de notre projet comporte plusieurs étapes.

Présentation du modèle de base : Notre modèle de base comprend plusieurs classes

[25] Extensible Markup language

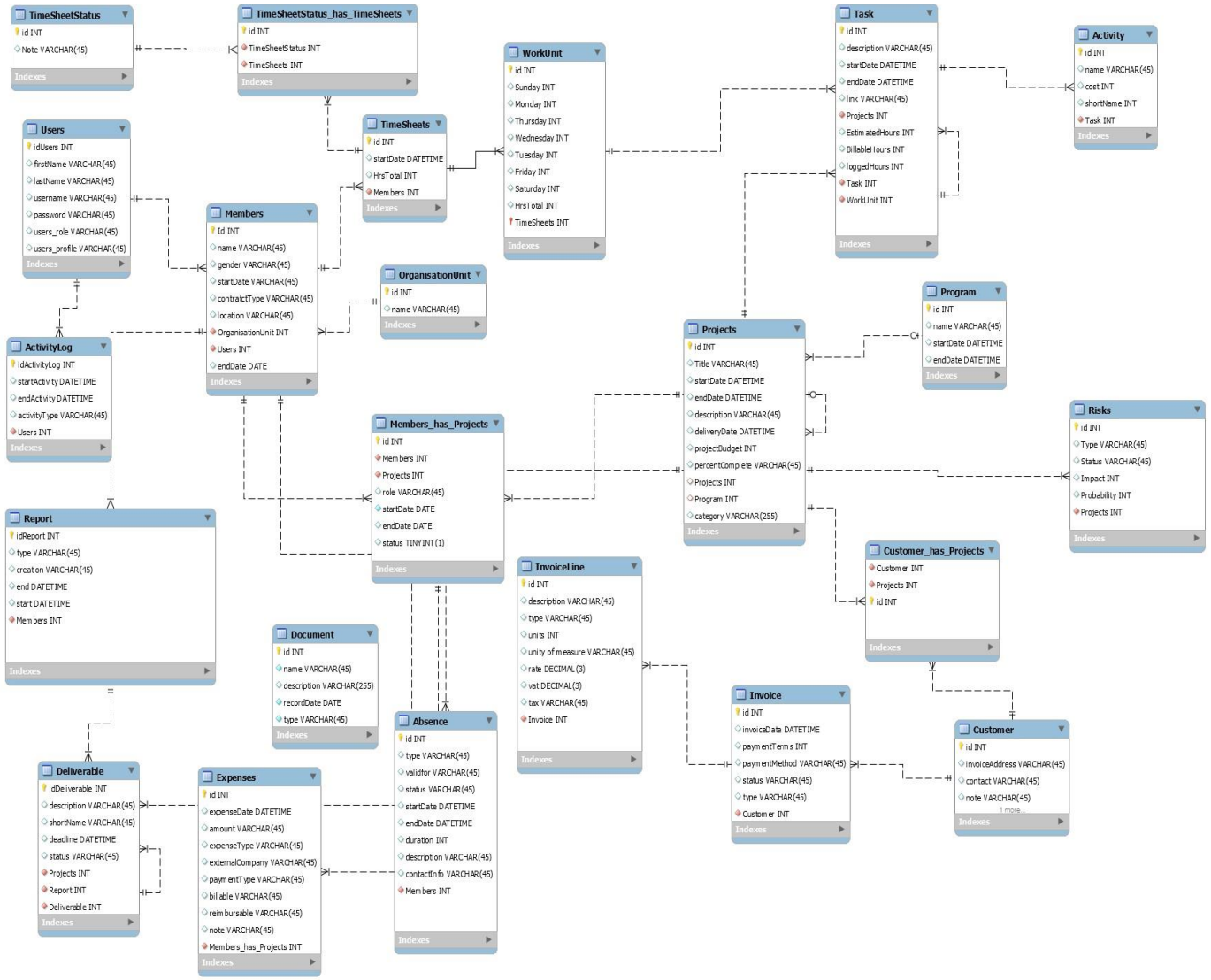


Figure 18 Diagramme de classe de base

Correspondance entre les objets Salesforce et les Classes du diagramme :

Dans cette étape on fait une comparaison entre une classe du modèle de base et un objet Salesforce. La correspondance se fait suivant les types des champs. Ainsi on vérifie s'il existe une correspondance des types de champs entre la classe et l'objet Salesforce. Si cette correspondance est égale à 75% cet objet Salesforce peut être utilisé pour représenter la classe de notre modèle de base. Pour les champs manquants on complète par des champs personnalisés.

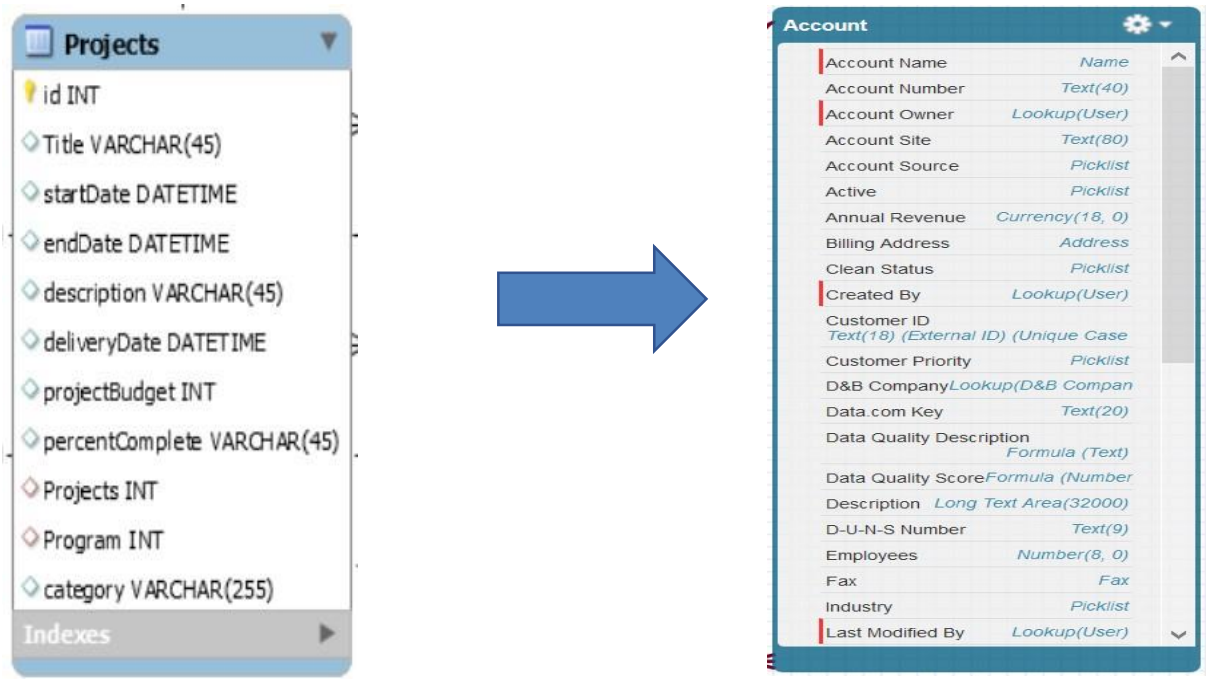


Figure 19 Comparaison entre les types des champs

Vérification des différentes relations entre les objets :

La deuxième étape consiste à recenser les autres SObjects[26] qui sont liés au SObject choisi premièrement. Le choix se fera suivant deux critères D'abord on regarde le type de relation qui existe entre le SObject choisit et les autres auxquels il est lié. Ensuite on regarde les types des champs de ces SObject.

Si le premier critère est n'est pas satisfait on utilise ce SObject et on crée la relation entre ces deux SObjects. Si les deux critères ne sont pas satisfaits on utilise un autre SObject.

NB : Il existe deux types de relations sous Salesforce. La relation master Détails (plusieurs à plusieurs) et la relation Look Up (un à plusieurs).

[26] https://trailhead.salesforce.com/fr/content/learn/modules/apex_database/apex_database_sobjects

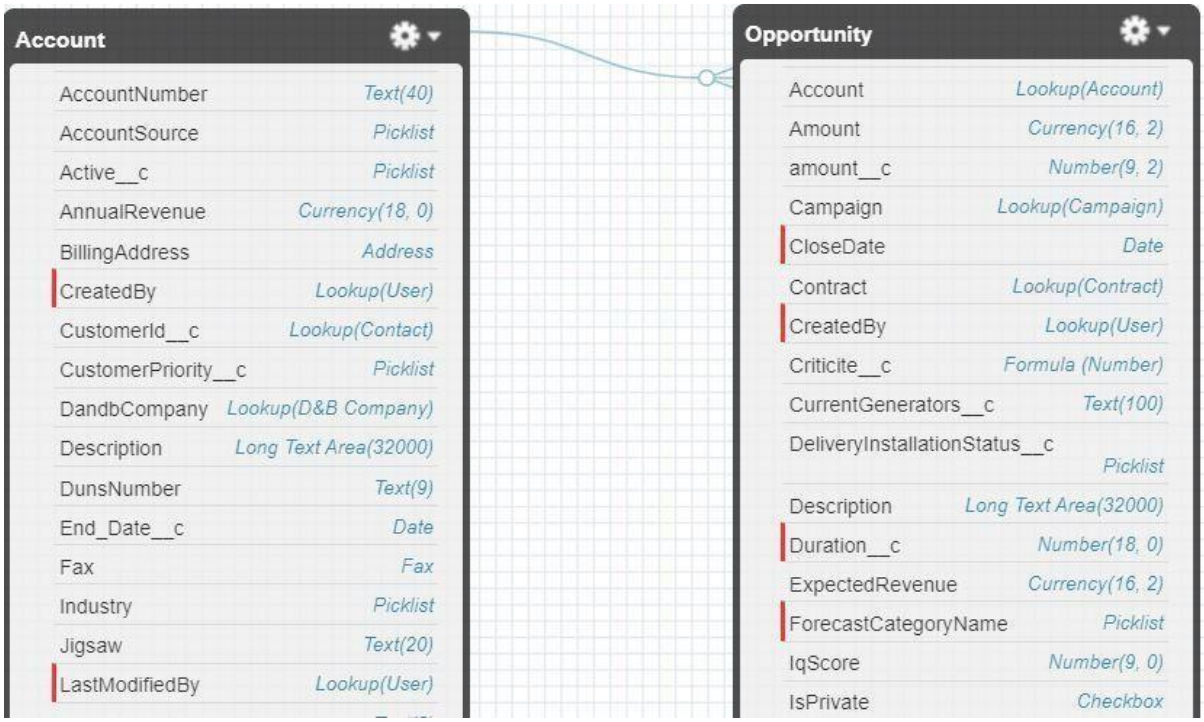


Figure 20 Liaison entre les différents objets

Création d'une relation pour un objet dans Salesforce :

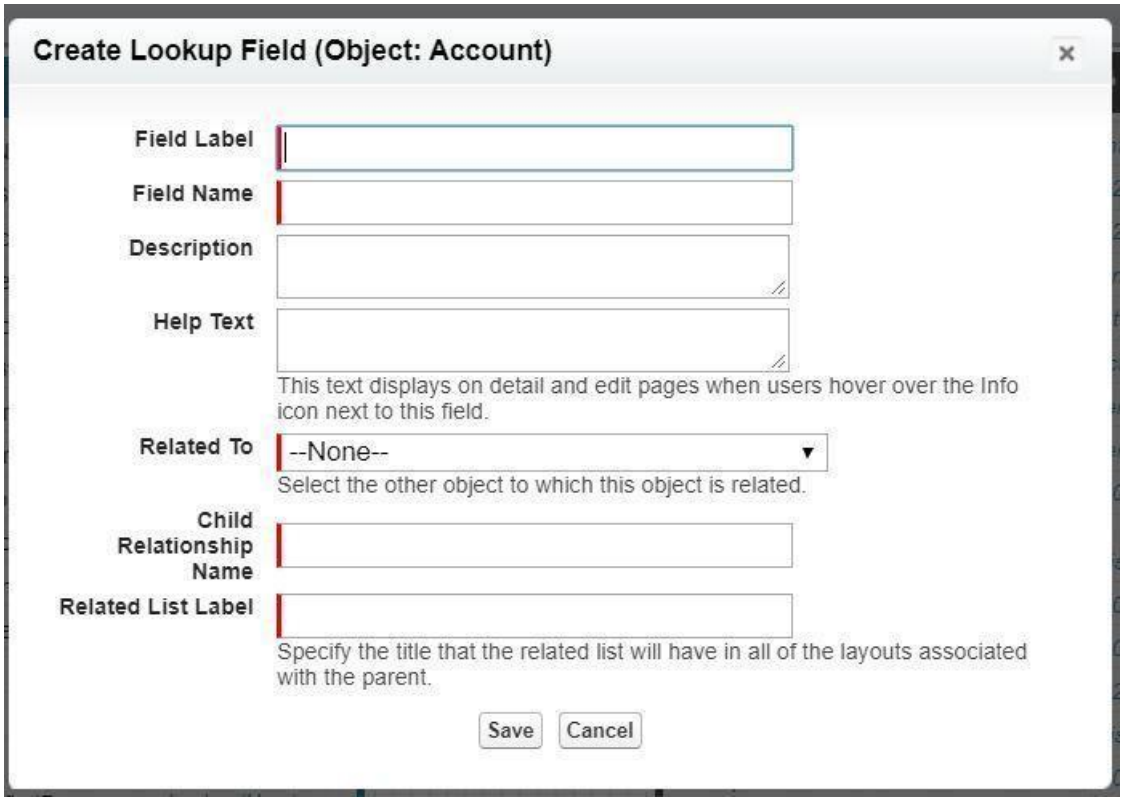


Figure 21 Création d'une liaison entre deux objets

Si les deux premières étapes ne sont pas satisfaites on crée un objet personnalisé.
Cependant sous Salesforce il existe une limitation de création d'objets personnalisés vu que celleci n'est pas gratuite.

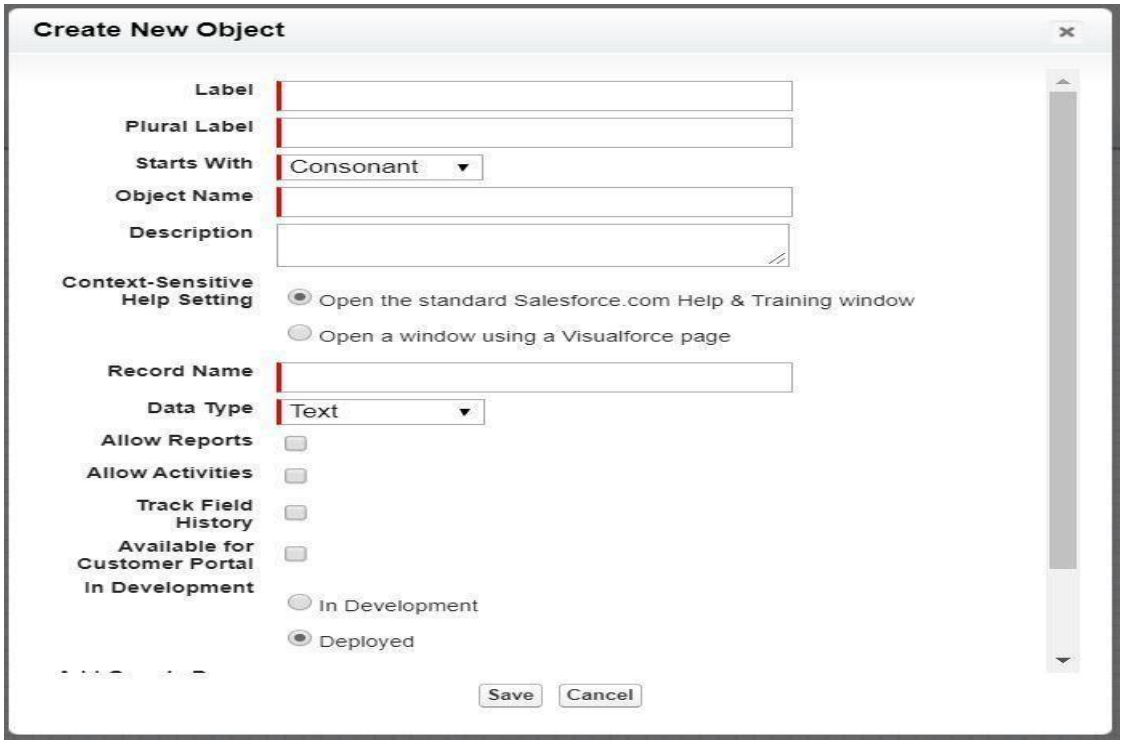


Figure 22 Création d'un Objet personnalisé

Ces différentes étapes ont permis d'avoir le modèle suivant :

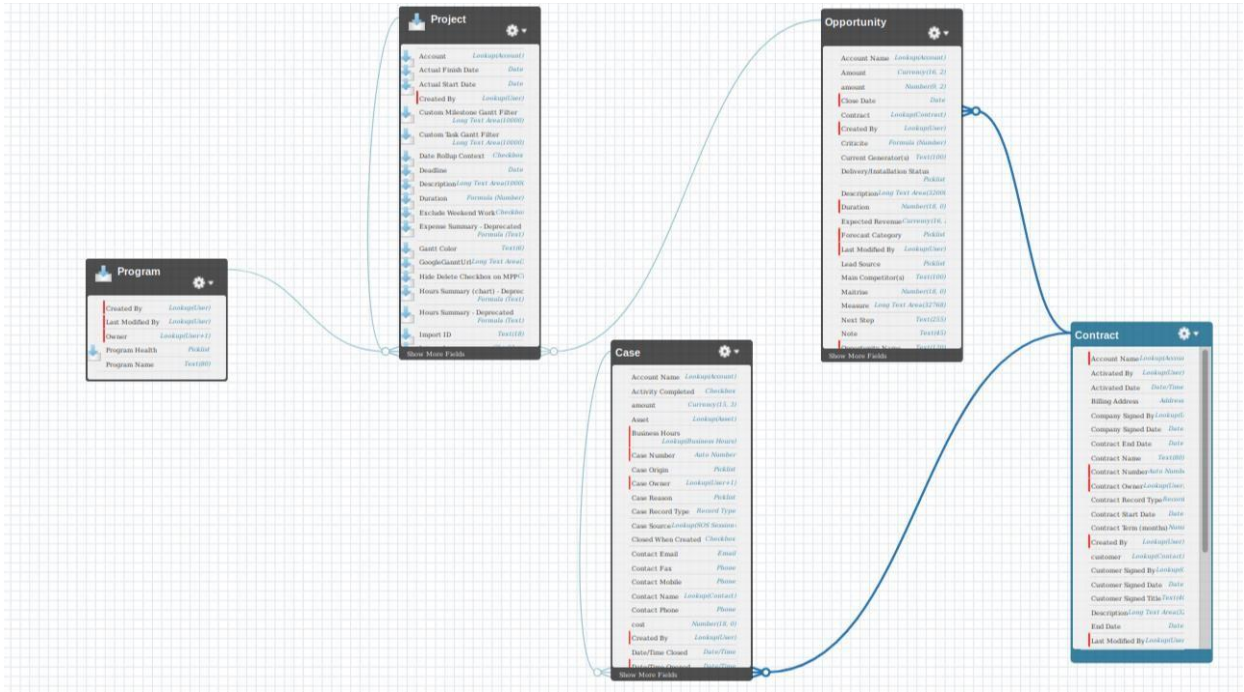


Figure 23 Diagramme de classe final

Lorsqu'un objet est utilisé pour représenter plusieurs classes on les différencie en créant un Record Type.

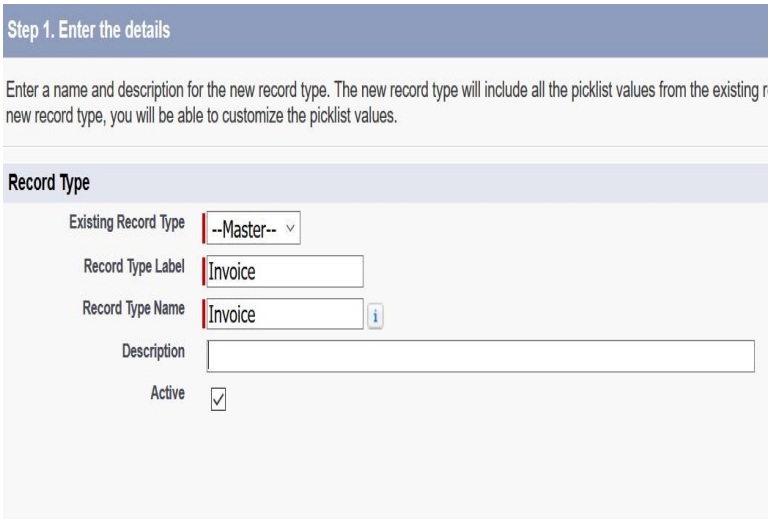


Figure 24 Création d'un record type 1

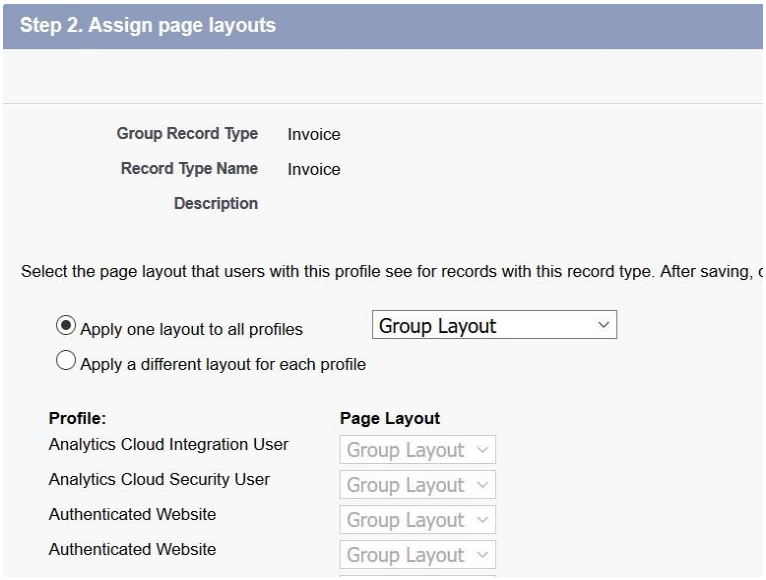


Figure 25 Création d'un record type 2

5.5 Les autres outils du projet

5.5.1 Le design:

Pour le compte de notre projet nous avons utilisé comme outil de Design l'outil par défaut de Salesforce :SLDS[27]

Le système SLDS nous aide à créer des applications avec l'aspect et la convivialité de Lightning Experience sans avoir à écrire une seule ligne CSS. SLDS est un framework CSS qui nous donne accès aux icônes, palettes de couleurs et polices que nos développeurs utilisent pour créer Lightning Experience.

[27] Salesforce Lightning Design System

SLDS est un ensemble de modèles de conception reproductibles et de bout de code réutilisables, nommés composants. Les développeurs associent ses composants de différentes manières pour créer des applications avec du code JavaScript. Ses caractéristiques sont :

Evolutivité : Les développeurs utilisent les éléments fondements dans un grand nombre de produits et de fonctionnalités.

Efficacité : Les développeurs se concentrent sur l'ergonomie et l'utilité.

Cohésion visuelle : permet de créer des produits et fonctionnalités visuellement cohérent pour une expérience utilisateur homogène.

Partage : permet aux développeurs de récupérer les parties d'un projet et de s'appuyer sur le travail

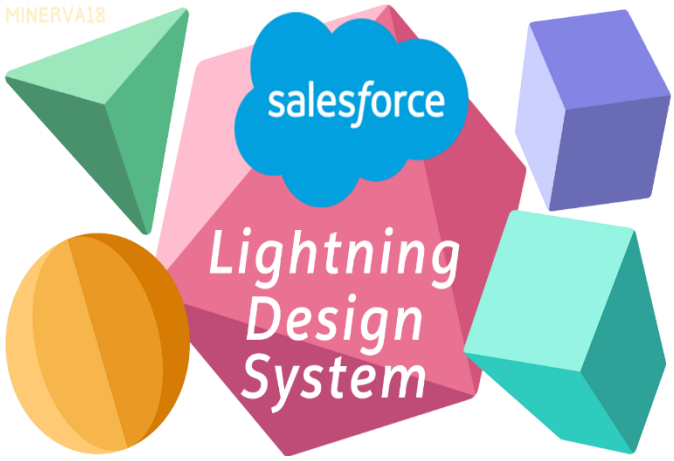


Figure 26 SLDS

L'outil SLDS propose des bouts de codes prêt à être utilisés en cas de besoin.

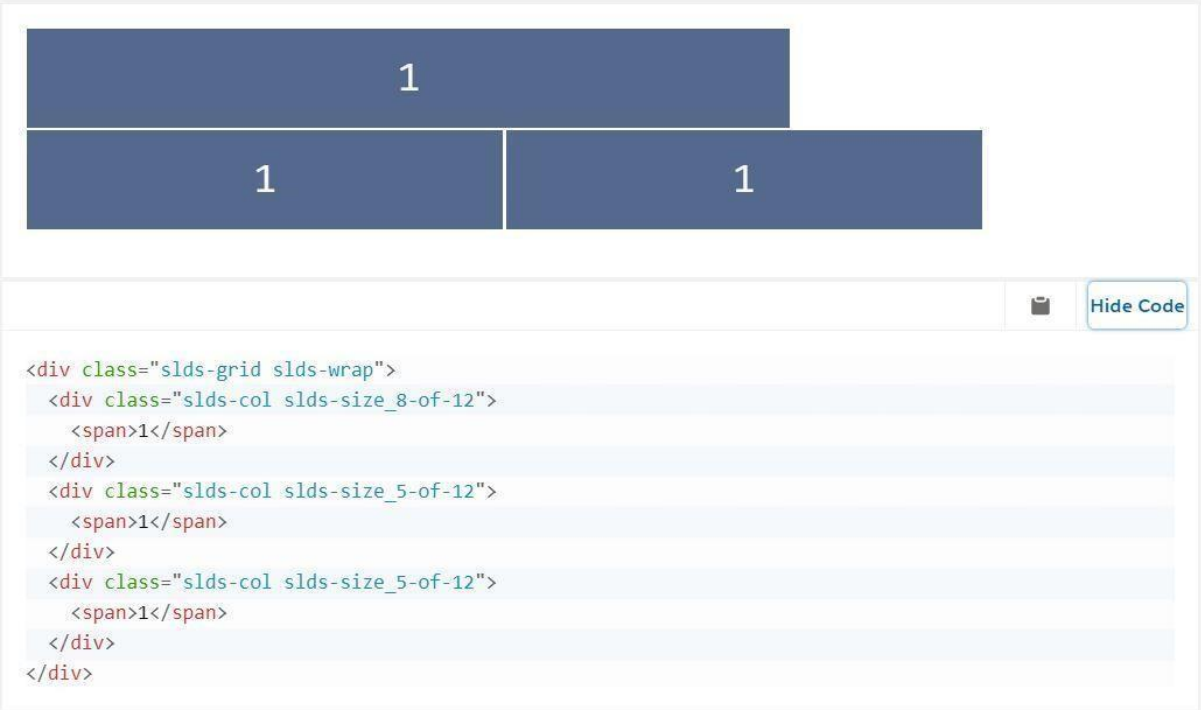


Figure 27 Exemple de code SLDS

5.5.2 Le Langage SOQL :

Salesforce Object Query Language. est utilisé pour lire les informations stockées dans la base de données de notre organisation.Le langage SOQL est syntaxiquement similaire à SQL [28]

Nous pouvons écrire et exécuter une requête SOQL dans le code Apex ou dans l'éditeur de requêtes de la console de développement.

SOQL est similaire à l'instruction SELECT dans le langage SQL largement utilisé, mais est conçu spécifiquement pour les données Salesforce.

Semblable à la commande SELECT du langage de requête structuré (SQL), SOQL vous permet de spécifier l'objet source (tel que Compte), une liste des champs à récupérer et les conditions de sélection des lignes de l'objet source.

SOQL ne supporte pas toutes les fonctions avancées de la commande SQL SELECT. Par exemple, nous ne pouvons pas utiliser SOQL pour effectuer des opérations de jointure arbitraires, utiliser des caractères génériques dans les listes de champs ou utiliser des expressions de calcul.

5.5.3 La notion de Governor Limit:

Comme Apex est exécuté dans un domaine mutualisé, le moteur d'exécution de Apex applique une limitation stricte pour empêcher qu'un emballement de code Apex ne monopolise des ressources partagées. Ces limitations, ou gouverneurs, surveillent et applique les statistiques présentées dans le tableau suivant. Si un code Apex dépasse une limite, le gouverneur associé génère une exception à l'exécution qui ne peut pas être gérée. Exemples de governor limit :

Description	Synchronous Limit	Asynchronous Limit
Total number of SOQL queries issued[1]	100	200
Total number of records retrieved by SOQL queries	50,000	
Total number of records retrieved by `Database.getQueryLocator`	10,000	
Total number of SOSL queries issued	20	
Total number of records retrieved by a single SOSL query	2,000	
Total number of DML statements issued[2]	150	

Figure 28 Governor Limit

5.5.4 Les Tests:

Les tests font partie intégrante d'Apex. Avec le langage Apex, nous avons des classes de tests séparées à développer pour tous les tests unitaires.

[28] Structured query language

Dans la console du développeur de Salesforce, le code doit avoir une couverture de 75 % pour être déployé en production. Cette couverture de code est effectuée par les classes de test. Les classes de test sont les extraits de code qui testent la fonctionnalité des autres classes Apex.

```
@isTest
private class CustomerTriggerTestClass {
   static testMethod void myUnitTest() {
      //Create Data for Customer Objet
      APEX_Customer__c objCust = new APEX_Customer__c();
      objCust.Name = 'Test Customer';
      objCust.APEX_Customer_Status__c = 'Inactive';
      insert objCust;

      // Now, our trigger will fire on After update event so update the Records
      Test.startTest();    // Starts the scope of test
      objCust.APEX_Customer_Status__c = 'Active';
      update objCust;
      Test.stopTest();     // Ends the scope of test

      // Now check if it is giving desired results using system.assert
      // Statement.New invoice should be created
      List<apex_invoice__c> invList = [SELECT Id, APEX_Customer__c FROM
         APEX_Invoice__c WHERE APEX_Customer__c = :objCust.id];
      system.assertEquals(1,invList.size());
      // Check if one record is created in Invoivce sObject
   }
}
```

Figure 29 Exemple de classe de test

5.5.5 La gestion de la Sécurité:

Comme Salesforce est un CRM entièrement basé dans le cloud l'aspect sécurité est fondamental. Dans le cadre de notre projet nous avons utilisé la sécurité sous Salesforce. La plateforme Salesforce propose quatre niveaux de sécurité : Sécurité niveau organisation, niveau Objets niveau champ et niveau enregistrement.

Organisation : on peut gérer la liste des utilisateurs autorisés, définir des stratégies de mot de passe et limiter les connexions à certains horaires et emplacements.

Objet : l'accès aux données de l'objet est l'aspect le plus facile à contrôler. En définissant des autorisations pour un type d'objet particulier, il est possible d'empêcher un groupe d'utilisateurs de créer, voir, modifier ou supprimer des enregistrements de cet objet.

Champs : on peut limiter l'accès à certains champs, même si un utilisateur a accès à l'objet.

Enregistrement : on peut autoriser des utilisateurs spécifiques à contrôler un objet, mais limiter l'accès aux enregistrements d'objet qu'ils consultent.

Chapitre 6: Présentation de la Solution

Au niveau de cette partie nous allons présenter les différentes interf aces et les pages obtenus après la réalisation de notre application. Le projet comporte deux grandes parties :

La création d'un projet et des programmes La

gestion des finances et des dépenses :

6.1 La gestion d'un projet:

Avant d'accéder à la page d'accueil on se connecte. La page de connexion est la page de connexion par défaut de Salesforce.

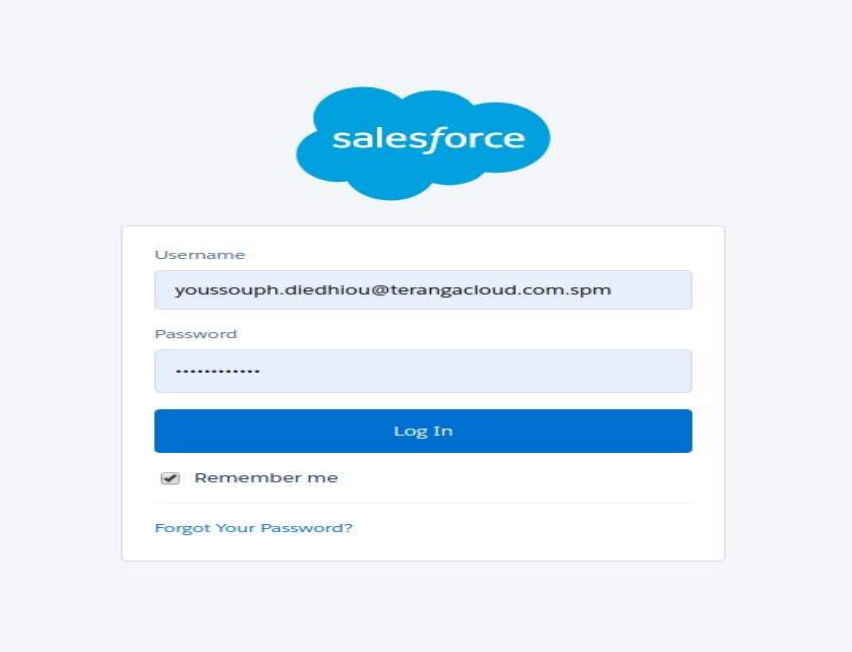


Figure 30 Page de connexion

Après connexion l'administrateur à la possibilité de créer des projets et des programmes mais il peut aussi les assigner un Project Manager en même temps. C'est ce Project Manager qui sera même chargé de gérer le projet et ses dépenses.

Un projet appartient à un programme ainsi on a les deux interfaces suivantes pour la création d'un projet et d'un programme.

D'abord l'administrateur peut créer des programmes et des projets.

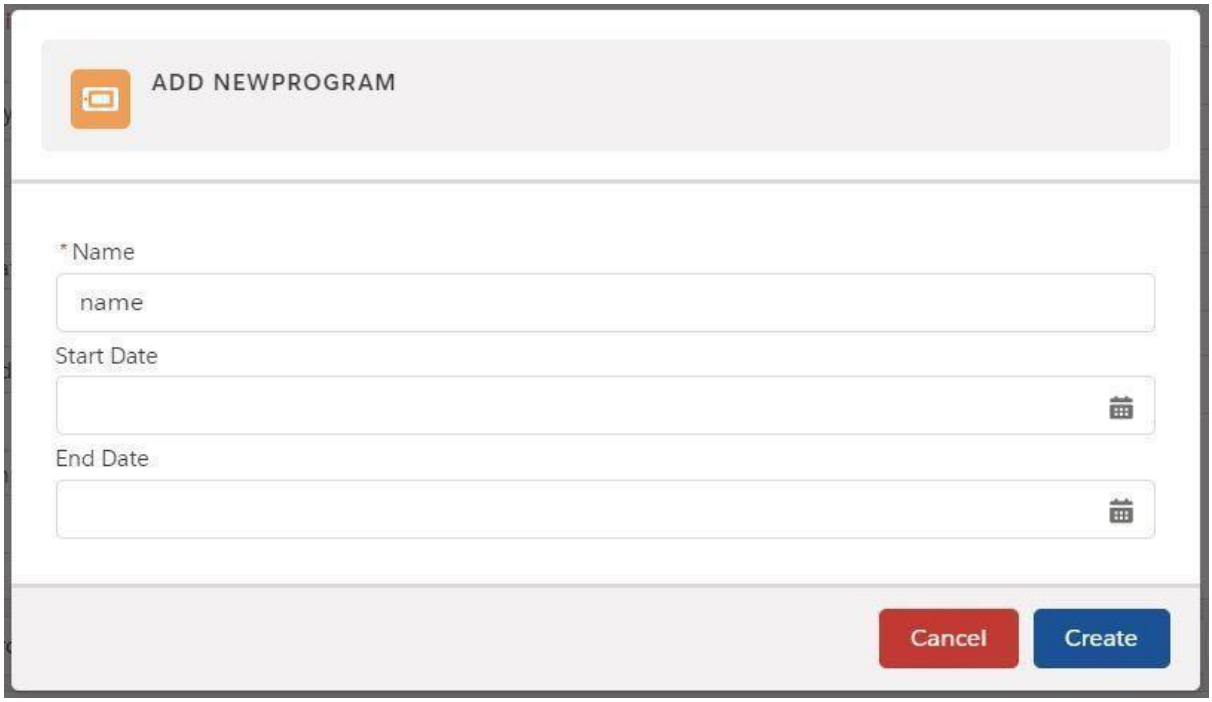


Figure 31 Création d'un program

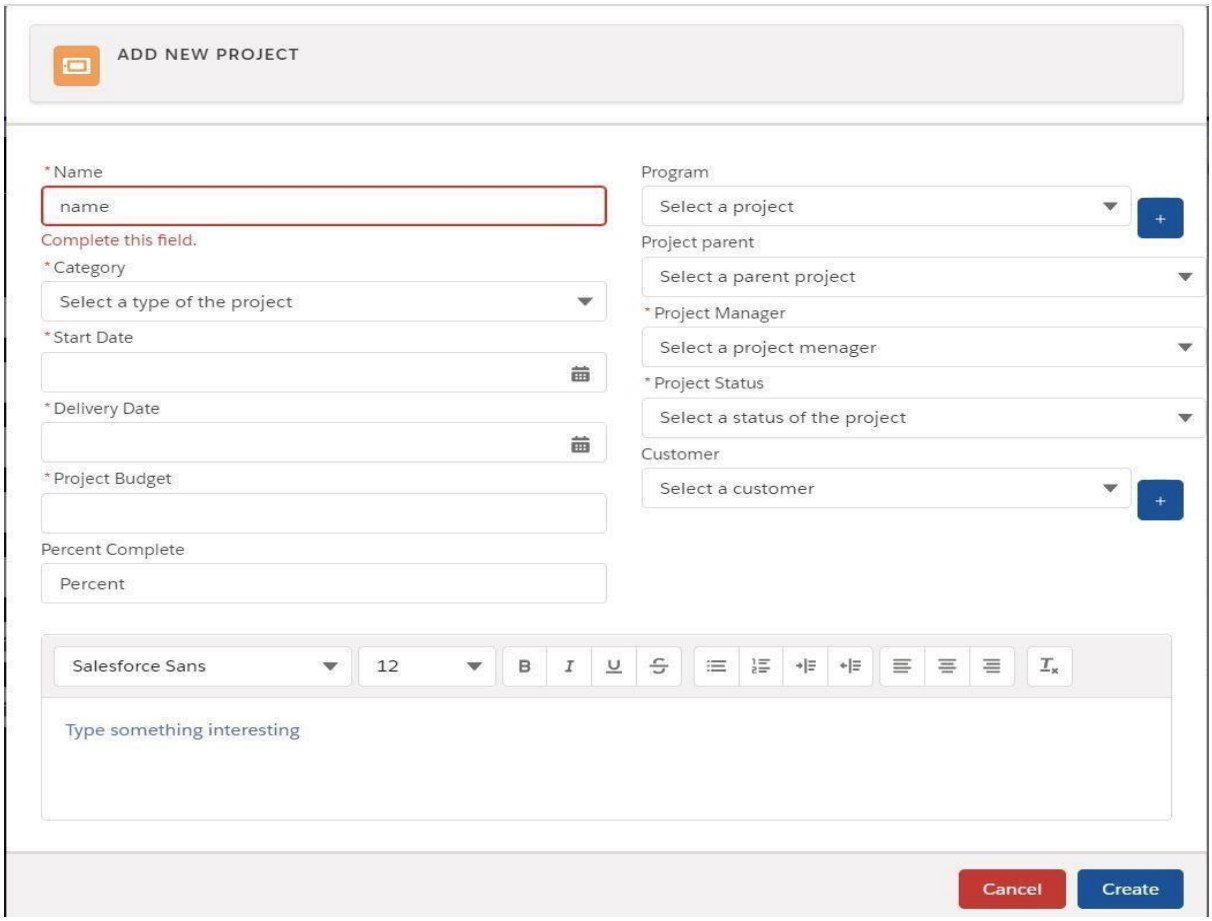


Figure 32Création d'un projet

Après la création l'administrateur peut voir la liste de tous les projets créés. Au niveau de cette liste on peut voir le type des projets leurs statuts et leurs budgets correspondants. On peut aussi au niveau de cette liste filtrer les projets suivant leur statut, type, client date de début ou date de fin.

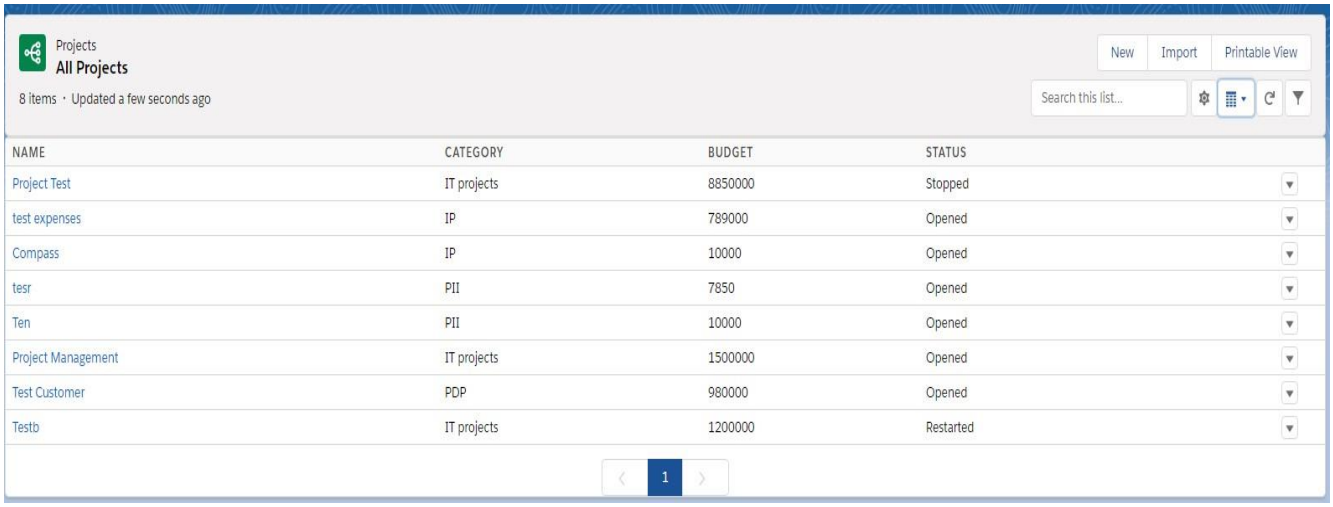


Figure 33 List des projets

6.2 Gestion des dépenses et de la facturation:

Cette partie du projet est gérée par le Financial. Celui-ci est désigné par l'administrateur lors de la création des projets.

Il a la possibilité de créer des dépenses en cas de besoin.

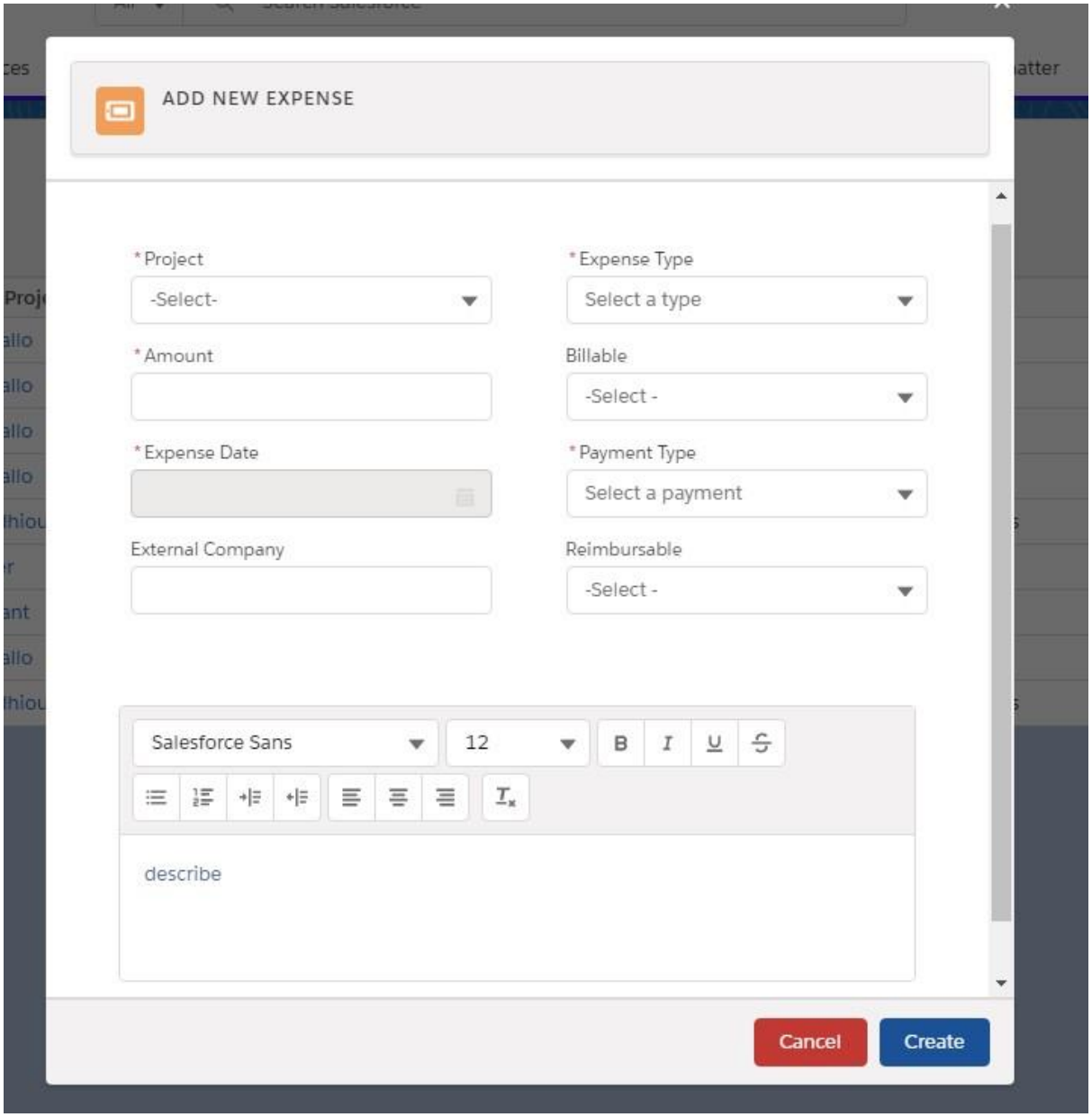
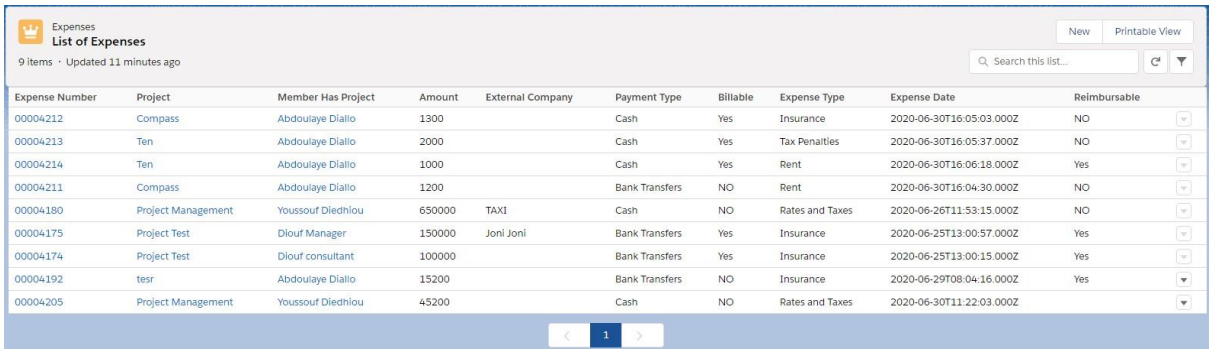


Figure 34 Formulaire d'ajout d'une nouvelle dépense

Après le formulaire de création de dépenses il a accès à la liste des différentes dépenses ce qui permet d'avoir une vue sur l'ensemble des dépenses d'un projet en fonction du budget.



Expense Number	Project	Member Has Project	Amount	External Company	Payment Type	Billable	Expense Type	Expense Date	Reimbursable
00004212	Compass	Abdoulaye Diallo	1300		Cash	Yes	Insurance	2020-06-30T16:05:03.000Z	NO
00004213	Ten	Abdoulaye Diallo	2000		Cash	Yes	Tax Penalties	2020-06-30T16:05:37.000Z	NO
00004214	Ten	Abdoulaye Diallo	1000		Cash	Yes	Rent	2020-06-30T16:06:18.000Z	Yes
00004211	Compass	Abdoulaye Diallo	1200		Bank Transfers	NO	Rent	2020-06-30T16:04:30.000Z	NO
00004180	Project Management	Youssouf Diedhiou	650000	TAXI	Cash	NO	Rates and Taxes	2020-06-26T11:53:15.000Z	NO
00004175	Project Test	Diouf Manager	150000	Joni Joni	Bank Transfers	Yes	Insurance	2020-06-25T13:00:57.000Z	Yes
00004174	Project Test	Diouf consultant	100000		Bank Transfers	Yes	Insurance	2020-06-25T13:00:15.000Z	Yes
00004192	tesr	Abdoulaye Diallo	15200		Bank Transfers	NO	Insurance	2020-06-29T08:04:16.000Z	Yes
00004205	Project Management	Youssouf Diedhiou	45200		Cash	NO	Rates and Taxes	2020-06-30T11:22:03.000Z	

Figure 35List des dépenses

Quand on clique sur le numéro de la dépense on peut voir les détails.

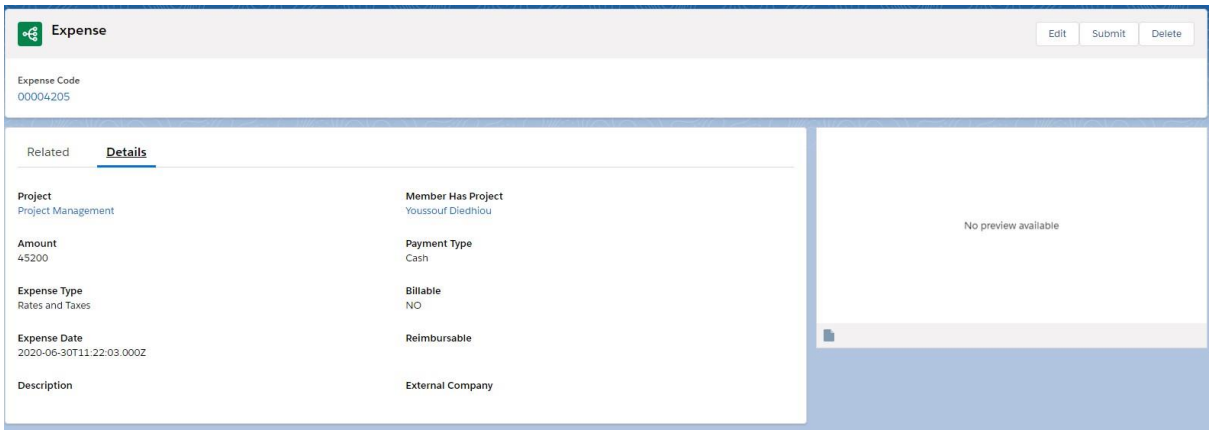


Figure 36 Détails d'une dépense

La gestion de la finance est aussi assurée par le Financial. A travers un formulaire il peut créer une nouvelle facture relative à une dépense pour un projet.

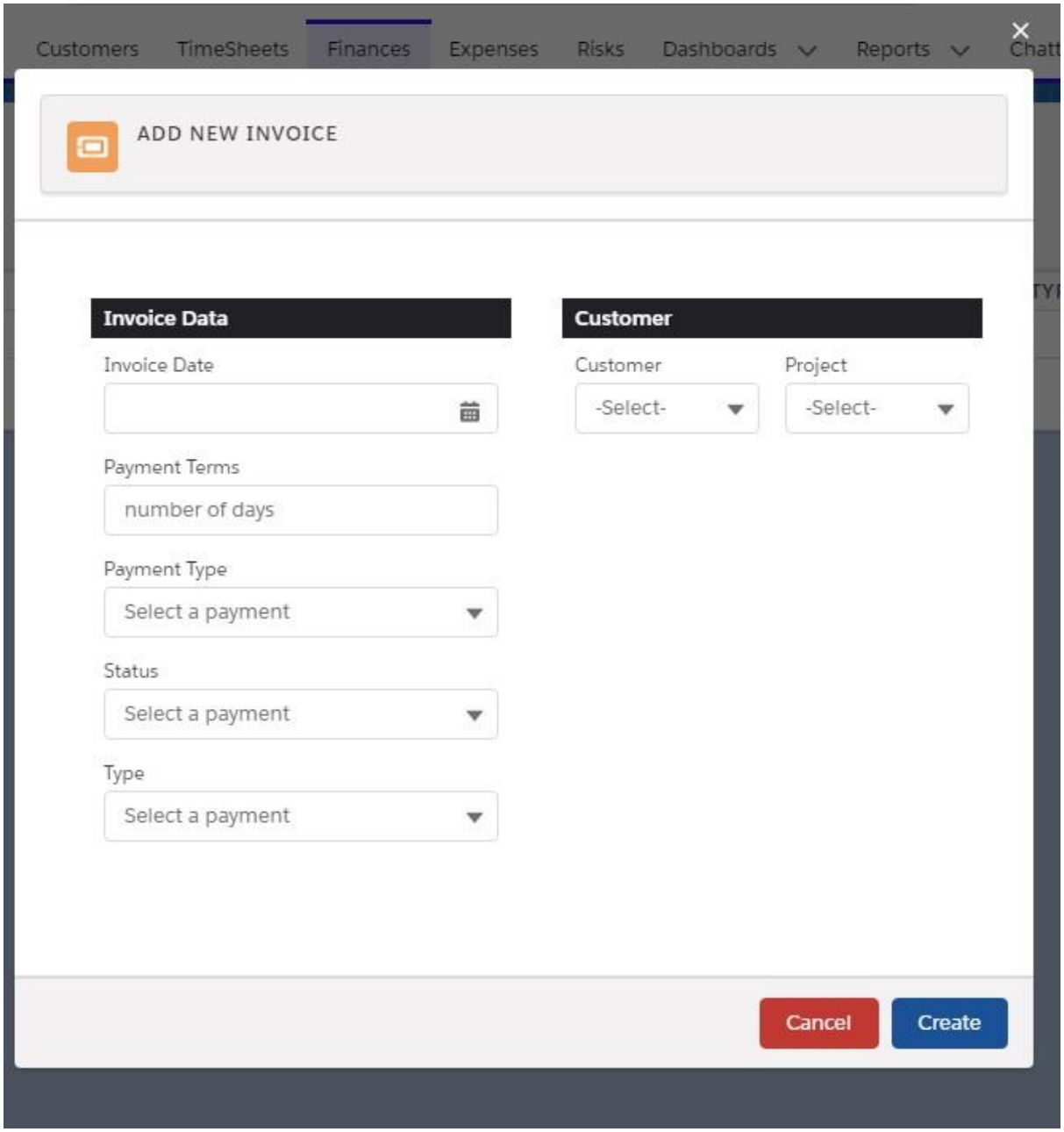


Figure 37 Formulaire Ajout d'une nouvelle facture

Après la création il a accès à la liste des différentes factures d'un projet.

Au niveau de cette liste il peut voir les différents détails d'une facture relative à un projet

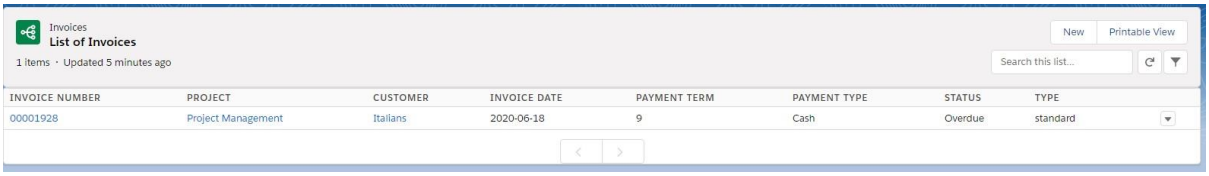


Figure 38 Liste des factures

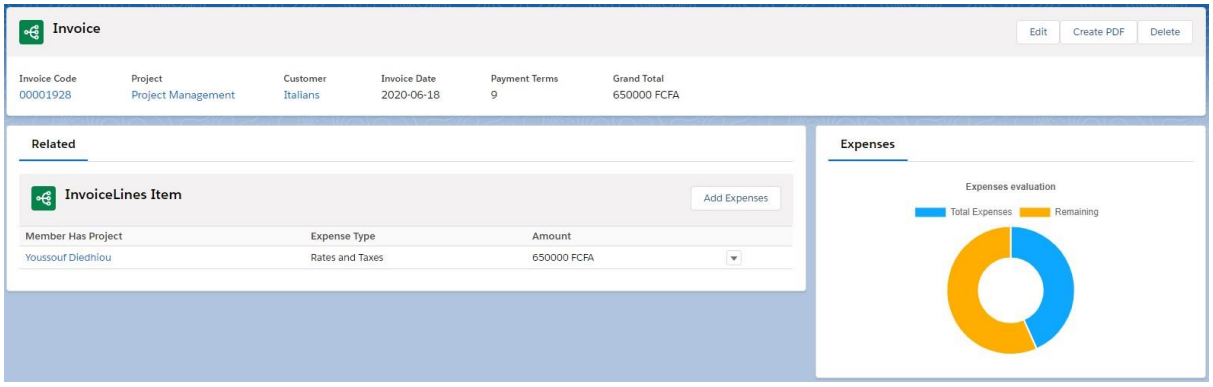


Figure 39 Détails d'une facture

Suivi des dépenses et des finances :

Pour mieux gérer les dépenses et la finance d'un projet plusieurs tableaux de bords dynamiques ont été créés.

Ces tableaux de bords permettront de suivre l'évolution des dépenses tout le long du projet. Ils permettront de ne pas dépasser le budget du projet mais aussi de voir l'évolution des dépenses pendant le projet.

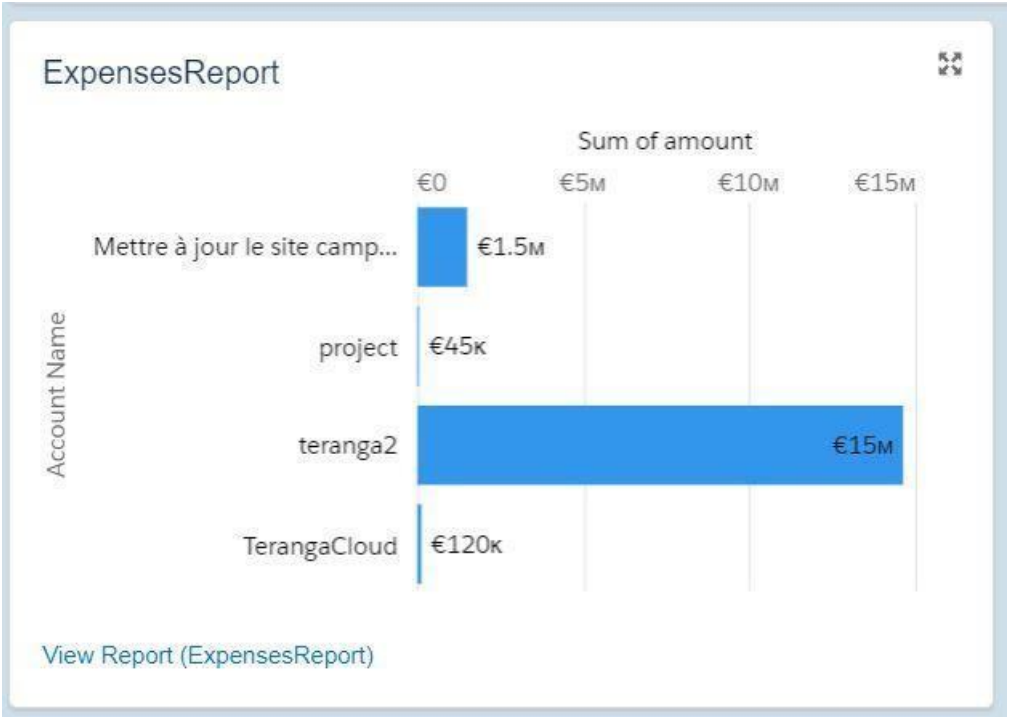


Figure 40 Tableau de bord des dépenses 1

Figure 41Tableau de bord des dépenses 2

Figure 42 Courbe des dépenses 1

Conclusion et perspectives:

La gestion de projet a pour but de permettre et de faciliter l'accomplissement des différents objectifs fixés. Notre travail consistait à développer un module de gestion des dépenses et de la finance dans un projet. La solution proposée a été développée sous Salesforce en utilisant le Framework Lightning aura component. Ainsi dans un projet quelconque notre module développé permettra de contrôler et de gérer les dépenses à travers des listes et des tableaux de bord, ce qui nous permet de suivre l'évolution des dépenses tout le long du projet. Notre module va permettre aussi de télécharger les différentes factures sous format pdf et les imprimer directement.

Les objectifs de notre stage peuvent être jugés satisfaisant. Pendant ces six mois de stage on a eu à acquérir beaucoup de connaissances sur le CRM Salesforce et son Framework Lightning mais aussi l'architecture SOC, les techniques d'intégration et de déploiement dans Salesforce.

Tout au long de notre stage les difficultés ont été nombreuses : d'abord la compréhension du modèle de données ensuite le choix des objets Salesforce à utiliser et enfin le manque d'expérience dans le domaine du développement avec le framework lightning.

Nous comptons améliorer notre module en créant d'autres tableaux de bord qui permettrons de ranger les différentes dépenses par type et des alertes pour informer l'administrateur de l'épuisement du budget alloué au projet.

Références :

[1] Salesforce module trailhead.

https://trailhead.salesforce.com/en/modules

[2] Salesforce Architecture.

https://trailhead.salesforce.com/en/content/learn/modules/starting_force_com/starting_understanding_arch

[3] Multi-Tenant architecture. https://developer.salesforce.com/page/Multi_Tenant_Architecture

[4] Understand Separation of Concern.

https://trailhead.salesforce.com/en/content/learn/modules/apex_patterns_sl/apex_patterns_sl_soc

[5] Apprendre les principes de la couche service.

https://trailhead.salesforce.com/fr/content/learn/modules/apex_patterns_sl/apex_patterns_sl_learn_sl_principles

[6] Learn Domain Layer Principles.

https://trailhead.salesforce.com/en/content/learn/modules/apex_patterns_dsl/apex_patterns_dsl_learn_dl_principles

[7] Learn Selector Layer Principles.

https://trailhead.salesforce.com/en/content/learn/modules/apex_patterns_dsl/apex_patterns_dsl_learn_selector_l_principles

[8] Cloud Computing : https://www.lebigdata.fr/definition-cloud-computing

[9] Salesforce Developer guide https://developer.salesforce.com

[10] Application Lifecycle and Development Models.

https://trailhead.salesforce.com/en/content/learn/modules/application-lifecycle-and-developmentmodels/understand-what-application-lifecycle-management-is

[11] Lightning Flows Salesforce:

https://trailhead.salesforce.com/fr/content/learn/modules/business_process_automation/flow

[12] Separation of Concern. https://en.wikipedia.org/wiki/Separation_of_concerns

[13] Martin Fowler Separation of Concern.

https://martinfowler.com/eaaCatalog/

[14] Qu’est-ce que le langage apex.

https://www.developerforce.com/guides/fr/apex_fr/Content/apex_intro_what_is_apex.htm