UNIVERSITÉ DE LORRAINE

Université de Lorraine - Licence informatique

Rapport de Stage

Documentation technique ExtJs & Développement de fonctionnalités Angular

egis asset management solutions

Mathieu Levy
Entreprise : Egis Asset Management Solutions
Année universitaire 2018-2019

Avant propos :

Initialement l'intitulé du stage était "Mise en place de tests unitaires ExtJS*[1] et Angular*[2]". Le stage devait durer du 08/04/2019 au 31/05/2019 pour une durée totale de 8 semaines dans l'entreprise EAMS (Egis Asset Management* Solution).

Lors de mon arrivée dans l'entreprise, le travail qui m'a été demandé correspondait à un autre sujet : "Mise en place d'une documentation technique*[3] ExtJs".

Après avoir terminé ce sujet, j'ai pu continuer à travailler et naturellement j'ai ajouté un intitulé : "Développement de fonctionnalités Angular".

Lors de ce stage, j'ai pu me former sur la notion de documentation technique englobée par la dette technique*[4], apprendre l'utilisation et le champ d'action du framework[5] ExtJS mais aussi celui du framework Angular, qui composent les applications appartenant à EAMS.

Tout ceci m'a permis de produire des livrables que je vais vous présenter tout au long de ce rapport de stage.

Un emploi du temps, est disponible dans le dossier joint à ce rapport.

Les explications de tous les termes techniques munis d'une astérisque, sont regroupées dans le glossaire situé en fin de rapport avant les annexes.

Remerciements

Je souhaite remercier chaleureusement Thomas Thouvenot pour m’avoir permis de travailler dans son entreprise, Pierre Danella pour m’avoir accompagné lors du premier mois de stage, Ludovic Michel pour m’avoir permis de rentrer dans le circuit de sprint ainsi que le reste de l’équipe EAMS pour m’avoir accueilli avec bienveillance.

Introduction

Lors du développement d'une application, il est nécessaire d'y associer de la documentation logicielle qui peut prendre plusieurs formes. Cette documentation permettra à l'utilisateur de l'application de comprendre comment l'utiliser.

La documentation technique est une partie de la documentation logicielle, créée par les développeurs pour les développeurs. Elle permet au nouveau développeur de comprendre comment utiliser les diverses méthodes produites par les développeurs précédents de l'application. Avec une documentation technique complète, l'application est accessible. Un manque de documentation génère de la dette technique. C'est la quantité de travail nécessaire pour remettre un logiciel à jour.

Ainsi mon sujet de stage concerne la réduction de la dette technique produite par la documentation technique dans le cadre d'une bibliothèque logicielle : ExtJS.

Lors de ce stage,j'ai pu, suite à la résolution de ce sujet, travailler en tant que développeur.

Dans ce rapport, je vais commencer par présenter l'entreprise qui m'a accueilli puis détailler la première et seconde partie de mon stage. Ces deux parties étant indépendantes, elles disposent chacune d'une présentation du travail demandé puis de leurs résultats.

Une conclusion clôturera ce rapport en faisant le bilan de ces deux mois de stage.

Description de l'entreprise

EAMS (Egis Asset Management Solution) est une filiale d'ERO (Egis Road Operation SA) qui elle même est une filiale d'EGIS PROJECT, filiale d'EGIS.

L'entreprise travaille avec deux types de sociétés d'autoroute. Les concessionnaires, comme APRR, qui ont une stratégie contractuelle avec l'État, et les exploitants, comme EEA (Egis Exploitant Aquitaine) et EROP (Egis Road Operation Portugal) qui sont mandatés par les concessionnaires pour effectuer les travaux de gestions courants.

L'exploitant est fournisseur du concessionnaire.

EAMS propose des solutions de gestion de maintenance autoroutière, ferroviaire ou de voirie. Pour la maintenance ferroviaire et de voirie, l'entreprise travaille avec Egis Rail ainsi que les villes de Metz, Nantes et Montpellier. Elle prévoit aussi de s'exporter en Inde et en Angleterre.

Situés proche de la gare de Part-Dieu à Lyon, les locaux de l'entreprise sont installés dans un bâtiment comportant de nombreuses autres filiales d'Egis, comme Egis Rail.

EAMS est passée à la méthode AGILE[6] SCRUM[7] avec sprints de deux semaines en février 2019. Le principe général de la méthode AGILE repose sur le fait de pouvoir adapter rapidement un projet en cas de changement d'aspect fonctionnel. Le SCRUM s'appuie sur le découpage en boîtes de temps, nommées « sprints », ayant une certaine durée. Un sprint commence par une estimation suivie d'une planification des opérations. Il se termine par une démonstration de ce qui a été achevé. Entre chaque sprint, l'équipe fait une rétrospective afin d'améliorer ses pratiques.

Au moment où ce rapport est rédigé, l'entreprise est encore en phase d'expérimentation par rapport à tout ce qu'implique cette transformation en terme d'organisation.

Elle est en pleine restructuration avec un besoin constant de nouveaux collaborateurs.

Ces changements impliquent un apprentissage, sur le tas ou par la formation, de nombreuses notions par les collaborateurs EAMS.

Il est donc très intéressant de pouvoir observer les changements et les expériences qui en découlent pendant cette phase de transition.

1.Sujet du stage

1.1.Mission

La mission qui m'a été confiée s'est déroulée sur deux grands axes :

- La documentation technique ExtJs
- Le développement de fonctionnalités en Angular

En parallèle, il m'a été demandé de bêta-tester la procédure "Nouvel arrivant" qui devrait permettre à tout nouveau collaborateur d'être en position de travail le plus rapidement possible.

1.1.1.Premier axe:

L'axe concernant la documentation ExtJs était, à la base, censé être unique, mais j'ai pu être intégré dans le circuit AGILE pour participer au travail de développeur après avoir terminé la documentation ExtJs.
Je vais commencer par décrire le premier axe :
EAMS possède deux applications qui se servent du framework ExtJs : AMS et Ask.
L'application AMS, qui est un acronyme pour Asset Management Software, propose un panel de modules permettant, par exemple, aux clients autoroutiers de gérer leur patrimoine.
Le module phare d'AMS est AmsGis, GIS étant un acronyme pour Geographical Information System (Système d'Information Géographique) .
AmsGis est un outil de consultation, administration et recensement de l'état du patrimoine de ses utilisateurs, permettant d'avoir un suivi optimal de leurs ressources.
L'application Ask n'est pas un acronyme mais son nom définit bien son ambition. Elle permet aux clients EAMS de déposer des demandes diverses, concernant les produits, sous forme de tickets, aux collaborateurs. Nous ne nous concentrerons pas plus sur cette application lors de la suite de ce rapport.
AMS a été construite malgré un processus de gestion qui était trop lourd, avec un effectif qui n'était pas suffisant. Dès lors, elle a acquis une dette technique imposante.
Une partie de cette dette technique est caractérisée par un manque de documentation technique. La problématique ici, est qu'il n'existait pas de solution permettant de générer cette documentation spécialement pour le framework ExtJs

1.1.2.Second axe:

Le second axe concerne tout simplement mon entrée dans le circuit de sprint. Il n'y a donc pas de livrable clairement défini pour cet axe. En effet, Thomas Thouvenot avait délégué la responsabilité de référent au product owner : Pierre D'Annela.
Celui-ci s'est absenté pour raison de santé à partir du jour où j'ai dû présenter mon travail au reste de l'équipe EAMS. Mon maître de stage m'a donc dirigé vers le second product owner, Ludovic Michel qui m'a proposé plusieurs tâches à réaliser. Je souhaitais me former sur le framework Angular. Je suis donc allé travailler avec les développeurs de l'application MapRail Visite, dont le client est Egis Rail, qui possède une version mobile en Angular.
L'application MapRail Visite permet à une ville de centraliser les relevés des défauts sur les voies ferrées, réalisés par ses techniciens. Ces derniers possèdent une tablette ayant l'application, sur laquelle il peuvent entrer les défauts de manière personnalisée. Les développeurs de MapRail travaillent en parallèle sur l'application CADIX qui a pour objectif la maintenance de la voirie.

1.2.Environnement de travail

Mon environnement de travail a changé plusieurs fois au cours de ces deux mois de stage.
Je vais commencer par décrire la disposition des bureaux puis vous présenter les différentes configurations techniques et spatiales mises à ma disposition.
EAMS possède un open-space, pouvant contenir une dizaine de stations de travail, occupé au milieu par le bureau du directeur.
Les développeurs travaillent dans cet open-space.
Un bureau, dans lequel se trouve une cheffe de projet junior, est accolé à cet open-space, séparé par une cloison simple.
Lorsque je suis arrivé dans les locaux, mon poste de travail devait être installé temporairement dans ce bureau.
J'ai par la suite été invité à installer mon poste de travail dans l'open-space suite au départ du précédent stagiaire.
Le fait de changer de bureau m'a permis de découvrir plusieurs aspects du monde de l'entreprise.
Dans le cadre des contraintes méthodologiques, lors du premier mois de stage, j'avais une réunion d'une heure avec Pierre D'Anella de manière régulière le vendredi matin, pour lui présenter mes avancées sur le travail de documentation technique.
Je pouvais, en plus, aller à sa rencontre à n'importe quel moment si j'avais des questions, à condition qu'il ne soit pas en réunion.
Le second mois, lors de mon entrée dans le circuit de sprint, je ne faisais plus de compte rendu au product owner comme avant mais directement lors de la réunion régulière de fin de sprint.
J'ai trouvé que les conditions de travail de ce second mois étaient optimales. En effet, j'étais intégré dans l'entreprise ce qui entraîne un confort de travail supplémentaire.

Voici maintenant les équipements et configurations techniques qui étaient à ma disposition:

Le bâtiment Le CARAT, dans lequel se trouve EAMS, possède un réseau local virtuel appelé VLAN EGIS. Mais pour des raisons de sécurité, EAMS étant la seule entreprise de développement dans le bâtiment, il a été décidé de séparer EAMS du reste du réseau.

EAMS possède donc son propre réseau local virtuel connecté à l'extérieur, appelé le VLAN EAMS.

L'entreprise se sert du logiciel Bitbucket pour gérer les dépôts de ses applications. Elle se sert de Skype Entreprise pour communiquer mais l'équipe de développement utilise en interne un autre logiciel de messagerie, Slack.

Le sprint est matérialisé par un tableau physique séparé en plusieurs zones sur lesquelles des tickets* vont naviguer. Un ticket est créé dans une zone appelée "NEW".

Une fois pris en charge, il passe en "IN PROGRESS". Un ticket terminé par le développeur passe en "CODE REVIEW", c'est à dire qu'un second développeur doit étudier le code du premier et valider son travail. Le ticket passe ensuite en "RECETTE USINE". Un product owner* vérifie que les fonctionnalités sur lesquelles le ticket intervient ne comportent pas de problèmes. Si le résultat ne correspond pas au ticket, il est alors "REJETÉ" et devra être traité à nouveau lors du sprint suivant.

Il existe en plus un système d'urgence pour permettre aux développeurs de résoudre rapidement des demandes exceptionnelles de clients.

Le logiciel Redmine est utilisé en parallèle du tableau pour pouvoir récupérer des statistiques sur les tickets réalisés.

Lors de mon arrivée dans l'entreprise, on m'a confié, dans un premier temps, une station de travail destinée à la bureautique, fonctionnant avec le système d'exploitation Windows et des applications d'entreprise comme la Suite Microsoft Office et Skype entreprise.

Pour les besoins de mon travail, et grâce à l'aide des techniciens compétents du support informatique, j'ai pu installer une machine virtuelle Linux.

L'application AMS, sur laquelle je devais travailler au départ, utilisait Docker* pour fonctionner. J'avais donc besoin de beaucoup de puissance et la machine virtuelle, avec le peu de mémoire vive que je lui allouais, n'était pas capable de supporter le docker. Il fallait donc que je redémarre régulièrement la machine virtuelle en fonction des plantages de celle-ci.

En allant confier ces complications retardant mon travail à Pierre Danella, j'ai appris deux choses :

- Je n'avais pas besoin de faire fonctionner l'application, car je ne travaillais que sur son code.
- Une station de travail avait été commandée pour tester une procédure "nouvel arrivant".

J'ai reçu cet ordinateur lorsque j'ai déménagé dans l'open space. C'était un portable destiné au départ à la CAO (Conception Assisté par Ordinateur, des applications en demande de puissance) et, d'après les techniciens du support informatique, le meilleur rapport poids/puissance sur le marché actuellement.

Cet ordinateur était en configuration de test sous Ubuntu avec machine virtuelle Windows. A ce moment, a démarré ma mission secondaire de testeur de la station de travail dans le cadre de la procédure "nouvel arrivant". Cette mission n'attendait pas un livrable particulier mais simplement un retour régulier informel de la situation.

Mon positionnement dans l'open-space m'a également permis de communiquer facilement avec les développeurs et j'ai ressenti une amélioration dans mon travail.

On m'a confié par la suite un second ordinateur car la puissante machine qui me servait d'outil de travail n'était pas nominalement à ma charge. J'avais donc deux ordinateurs pour travailler.

2.Documentation technique ExtJS

2.1.Objectifs

Le but ici était d'obtenir un outil de documentation technique utilisable par les développeurs de l'application AMS. Ceci pour réduire la dette technique.
Les livrables qui m'ont été demandés lors de cet axe sont les suivants :

- Un outil de documentation technique
- Procédure technique de mise en place de cet outil
- Méthodes d'utilisation de cet outil
- Des normes de développement basées sur le code[8]
- Un guide utilisateur
- Une présentation destinée au reste de l'équipe EAMS

Ces livrables devaient être validés par le Lead developer* ExtJS. J'avais ensuite la responsabilité de lui transmettre les livrables et je devais m'assurer qu'il avait la possibilité de s'en servir.
Je n'avais pas de date butoir car le product owner responsable de cette mission ne savait pas le temps que ça pourrait prendre. Je devais donc faire un travail d'analyse des différentes solutions présentes puis en proposer une qui pourrait être améliorée par la suite.
J'avais en plus comme objectif personnel de fournir rapidement des documents clairs et des solutions fonctionnelles faciles et rapides d'utilisation.

2.2.Installation & Veille technologique

Pour installer mon environnement de travail, j'ai pu récupérer les fichiers de l'application Ams via le gestionnaire de dépôts Bitbucket. C'est ainsi que j'ai découvert Docker, qui permet de simuler l'environnement d'Ams et donc de tester l'application directement. Elle sert aussi à récupérer les différents modules d'Ams qui appartiennent chacun à un dépôt git* différent. Ainsi, j'ai pu prendre connaissance de l'architecture d'Ams.

L'application est composée de nombreux modules dont le plus important est AmsGis. Chaque module ayant des ressources partagées, il existe un répertoire appelé "packages" qui contient comme son nom l'indique des packages ayant eux aussi chacun leur dépôt git.

En étudiant le code et les différents commentaires déjà présents, j'ai pu remarquer que certains avaient une structure particulière. Ils sont destinés à un parseur* maison multitâche. Vous trouverez un exemple en annexe.

Une fois tous les paramètres pris en compte, j'ai pu démarrer une veille technologique* dès la première semaine pour découvrir et tester différents générateurs de documentation technique sur le code du module AmsGis.

En effet le code de l'application Ams possédait très peu de commentaires de développeurs.

Ainsi, la génération de documentation technique s'est avérée inefficace dans de nombreux cas. Lorsque je l'ai signalé au product owner, il a insisté sur le fait que mon outil servirait d'abord à supprimer l'augmentation de la dette technique. Une fois les processus en place, la dette pourra être réduite par les développeurs

Ma démarche, dans la recherche de l'outil qui correspondrait le mieux à l'objectif demandé, était tout simplement de recenser dans un fichier tous les générateurs de documentation du langage javascript ou qui pourraient fonctionner avec ce langage. Ce fichier se nomme "Comparatif Outils ExtJS" et vous le trouverez associé à ce rapport. Vous avez donc accès à des informations supplémentaires sur tous les outils que je vais vous présenter ci-après. J'ai visité beaucoup de sites et forums différents et toutes les informations sont sourcées. La bibliographie de ce rapport contiendra les sources principales.

Ma recherche n'avait pas pour vocation d'être exhaustive mais bien de trouver un outil basique, fiable et capable d'évoluer facilement.

Dans le cas de la documentation technique ExtJS, un outil très souvent cité par la communauté utilisant le framework est jsDuck[9]. Cet outil était maintenu par Sencha, l'entreprise qui développe ExtJS.
Cet outil est devenu obsolète lorsque Sencha a choisi de travailler sur Doxi[10] .
C'est un outil qui génère un fichier json* qui sera ensuite utilisé pour produire une page web contenant toute la documentation.
Le souci, ici, est que le parseur utilisé par Sencha pour produire cette documentation est privé. Il aurait donc fallu faire un parseur maison pour traduire le fichier json.
Finalement, mes recherches m'ont orientées vers JSDoc[11][12][13][14] .
Cet outil qui, à la base, permet de générer de la documentation pour du code javascript, fonctionne également pour du code écrit à l'aide du framework ExtJS; vous pourrez trouver des captures d'écran exemple en annexe.

2.3.Implémentation de JSDoc

Mon choix s'est donc porté sur JSDoc pour générer la documentation technique.
Étant donné que j'avais le champ libre sur ce que je devais préconiser au développeur, j'ai voulu lui fournir une solution qui lui permette de n'avoir qu'à exécuter un script pour générer cette documentation.
Certaines conditions d'utilisation ont été spécifiées dans un fichier appelé "UG_DT_ExtJS" que vous trouverez également associé à ce rapport.
UG est un acronyme pour User-Guide qui signifie Guide Utilisateur, Dt est l'acronyme pour Documentation Technique.
Mon premier choix d'implémentation s'est d'abord porté sur l'écriture d'un script qui mettrait à jour directement toutes les documentations techniques de tous les modules et packages d'Ams.
Je ne savais pas alors que ces entités possédaient chacune un dépôt git.
Le product owner m'a donc dirigé vers une solution qui ne provoquerait pas un nombre trop important de pull request*.
Cette solution consiste à intégrer un sous-dossier, qui sera nommé "DocumentationTechnique", dans chacun de ces dossiers.
Vous en avez un exemple en annexe.
La conception de ce sous-dossier avait pour objectif de comprendre facilement comment générer de la documentation technique.
Il comporte :

- un dossier "documentation" destiné à contenir la documentation générée
- un dossier "exemples" comportant divers exemples de conventions de codage à respecter. Il permet également de s'entraîner à générer de la documentation.
- Un fichier de configuration de JSDoc facilement modifiable
- Le script principal de génération de documentation
- Un Readme écrit en Markdown* contenant la procédure

Une copie de ce dossier est associée à ce rapport.
Vous pouvez avoir accès à l'exemple de documentation technique en allant dans le sous-dossier "documentation" du dossier fourni et en ouvrant le fichier index.html avec votre navigateur web favori.
Tous les livrables étaient prêts à partir du début de la troisième semaine. Je l'ai signalé lors de la réunion de fin de sprint et Thomas Thouvenot m'a demandé de faire une présentation pour la semaine suivante, seul créneau disponible pour ces deux mois.

Afin de faire une présentation de qualité dans le temps imparti, j'ai préféré y travailler en plus sur mon temps libre.
Cette présentation est jointe au rapport.
J'ai eu de très bons retours de la part de toute l'équipe par rapport à cette présentation.
La principale critique est que je n'avais pas respecté la charte de présentation EAMS.
Je n'avais pas regardé les documents officiels en détail et j'avais respecté seulement le code couleur.
Les autres critiques étaient surtout liées au stress que j'éprouvais.
Finalement, JSDoc pourra être amélioré pour fournir un champ de recherche, ou encore modifier la mise en page des applications.

3. Développement de fonctionnalités Angular

3.1.Objectif

L'objectif ici était d'apprendre le fonctionnement du framework Angular tout en réalisant des tickets pour l'entreprise.

3.2.Formation à Angular

Avant de commencer à travailler en Angular, Ludovic Michel, mon nouveau référent, m'a donné une tâche sur l'application AMS pour manipuler ExtJs. Il s'agissait de remplacer un écran d'erreur qui apparaissait lors de la saisie de mauvais identifiants de connexion dans l'application. Vous avez le résultat en annexe.

Il m'a, par la suite, proposé de travailler en Angular sur le projet MapRail Visite.

Les développeurs de l'application m'ont d'abord donné des tâches simples pour prendre connaissance de l'application en détail et comprendre comment le framework était utilisé.

Il a fallu modifier l'affichage d'un élément puis supprimer les phrases écrites en dur dans le code HTML, pour les remplacer par une chaîne de caractères spécifique. Cette chaîne sera modifiée par le fichier de traduction. Enfin, j'ai participé à l'ajout d'une réglette sur la partie "map" de l'application afin de connaître l'échelle d'affichage.

Une fois ces tâches effectuées, j'ai eu pour mission de développer un générateur de clés de licence pour pouvoir verrouiller l'application.

Il fallait terminer ce générateur pour pouvoir sortir l'application.

Je devais fournir une application desktop en utilisant la technologie que je désirais et fournir son pendant mobile en utilisant Angular.

La tablette étant hors connexion, nous avons décidé de partir sur un schéma de validation parallèle qui se présente comme suit :

- Allumage de l'application.
- Si la clé est passée de date, une page spéciale s'affiche.
- La page affiche une "clé client" aléatoire et demande au client de contacter l'entreprise.
- Le client donne la "clé client" à l'entreprise.
- L'entreprise utilise son générateur de "clé licence" qui en donne une nouvelle en fonction de la "clé client" fournie.
- La "clé licence" obtenue est envoyée au client, et l'entreprise note le fait qu'une licence lui a été fournie.
- Le client rentre la "clé licence" dans la zone prévue à cet effet sur la tablette.
- La tablette vérifie la "clé licence" obtenue en générant sa propre "clé de vérification" grâce à une clé secrète partagée.
- L'application est maintenant débloquée pour une durée de deux ans.

La technologie que j'ai choisie pour faire l'application desktop est Java 11* avec javaFX 11*. En effet, je maitrisais bien cette technologie et mon intérêt était de terminer rapidement et proprement la partie desktop pour passer à la version mobile et en apprendre plus sur Angular.
Il m'a été demandé de faire une interface graphique basique. Vous trouverez des captures d'écran de l'application et de l'interface en annexe.
La seule grosse difficulté rencontrée ici est qu'il m'était impossible de produire un exécutable via l'IDE* IntelliJ[15] . En effet, à partir de la version 11 de Java, JavaFX a été retiré du pack commun[16] et il fallait donc ajouter des options spécifiques de compilation.
Le fait surprenant est que l'IDE parvenait à utiliser ces options pour exécuter l'application mais pas pour produire un exécutable.
Si je souhaitais fournir l'application à un développeur ne possédant pas IntelliJ, il fallait compiler puis exécuter le logiciel à l'aide d'un script de ma création.

Mais un autre problème est apparu quand j'ai essayé de fournir les fichiers et le script aux autres développeurs. En effet, certains travaillant sous Windows, il fallait fournir un script différent, ce que j'ai fait, mais l'application ne se lançait pas.
La solution fut de prendre la version 8 de java et javaFX, qui elle ne nécessitait pas d'options particulières. J'ai ensuite adapté l'algorithme écrit en Java au langage Typescript*[17], utilisé par Angular, pour pouvoir l'implémenter dans l'application mobile.
Nous avons ensuite ajouté une base de données SQLite* dans la tablette pour pouvoir stocker la clé de licence et la date de fin de validité de la licence. Nous n'avions plus qu'à modifier la première page affichée lors du démarrage de l'application.
Le ticket concernant la licence étant terminé, je me suis concentré sur la rédaction de ce rapport tout en me tenant disponible en cas de demande de la part d'autres développeurs comme par exemple faire une recette usine* de nouvelles fonctionnalités.
FInalement, j'ai pu en apprendre plus sur le fonctionnement de l'entreprise ainsi qu'être formé au framework Angular, tout en participant au travail des développeurs.

Conclusion

Lors de ce stage de deux mois, tous les objectifs ont pu être atteints.
La documentation technique ExtJS est en place et utilisable par n'importe quel développeur de l'entreprise.
L'apprentissage d'Angular et ExtJS s'est bien déroulé, et j'ai pu développer des fonctionnalités ou corriger des erreurs à l'aide de ces framework.
Au regard de la formation, des compétences en algorithmique, sécurité , langages orientés objets et développement web ont été nécessaires.
L'apport de la licence s'est énormément ressenti au niveau de ma capacité d'apprentissage. J'ai pu comprendre très rapidement les framework ExtJS et Angular, mais aussi le langage typescript grâce aux méthodes apprises lors de ma formation.
Il a fallu tout de même que je me remette à niveau en javascript en début de stage car le niveau demandé était plus élevé que celui que je possédais auparavant
J'ai appris énormément lors de ce stage. Je n'avais jamais travaillé dans une entreprise de développement et ce fut difficile au départ de comprendre les attentes de l'entreprise. Ce fut un plaisir de pouvoir travailler sur des sujets qui me passionnent, de pouvoir transmettre la connaissance acquise à mes collègues, et de pouvoir être formé sur les bases du travail en entreprise.
Comprendre la manière dont le système était organisé m'a aussi intéressé. J'ai ainsi pu en tirer mes propres méthodes de développement qui vont augmenter mon efficacité.
Lorsque je suis arrivé dans l'entreprise, j'avais le sentiment que je ne travaillais pas assez, j'avais un manque de confiance en mes capacités. J'ai été heureux d'avoir les retours de mes collègues, lors de la présentation de ma solution, qui ont été impressionnés par mon travail.

Bibliographie

[1] **ExtJS.** *Site officiel, Sencha* https://www.sencha.com/products/extjs/#overview [Online; access April 2019]

[2] **ANGULAR.** *site officiel d'Angular* https://angular.io/ [Online ; access May 2019]

[3] **DOCUMENTATION TECHNIQUE.** *Wikipedia* https://cutt.ly/WuORwN [Online; access April 2019]

[4] **DETTE TECHNIQUE.** *Wikipedia* https://cutt.ly/nuORl1 [Online ; access April 2019]

[5] **FRAMEWORK.** *Wikipédia* https://cutt.ly/MuOTtV [Online ; access April 2019]

[6] **MÉTHODES AGILES.** *Wikipédia* https://cutt.ly/3ujGzF [Online ; access April 2019]

[7] **SCRUM.** *Wikipedia https://cutt.ly/6uOTa7* [Online; access April 2019]

[8] **DOCUMENTATION.** *Documenter son code proprement* https://cutt.ly/YujTfr [Online; access April 2019]

[9] **JSDuck.** *senchalabs* https://cutt.ly/jujHJx [Online ; access April 2019]

[10] **Doxi** *Sencha* https://cutt.ly/OujHyy [Online; access April 2019]

[11] **JSDoc.** *Site officiel* http://usejsdoc.org/ [Online; access April 2019]

[12] **JSDoc.** *github* https://github.com/jsdoc/jsdoc [Online; access April 2019]

[13] **JSDoc.** Comparatif *FusionChart* https://cutt.ly/JujCiC [Online; access April 2019]

[14] **JSDoc.** cours, *Achref El Mouelhi https://cutt.ly/1uXfUp* [Online; access April 2019]

[15] **InteliJ.** *JetBrain, site officiel https://cutt.ly/ruOYSy* [Online ; access May 2019]

[16] **JavaFX.** *openJFX, site officiel de javaFX* https://openjfx.io/ [Online ; access May 2019]

[17] **TypeScript.** *Site officiel de Typescript* https://cutt.ly/6uj9Bf [Online ; access May 2019]

Les liens qui mènent à Wikipédia ont été validés par mon référent entreprise.
Les informations qui me sont parvenues via des documents internes sont définies dans le glossaire ci-après.

Glossaire

Généralités

Asset Management

Défini par la norme ISO55000 comme étant une activité coordonnée d'une organisation pour prendre conscience de la valeur des ressources.
Plus d'explications dans cette courte vidéo:
https://www.youtube.com/watch?v=bjvlWn3RC4Y

Retour rapport

Desktop

Ordinateur. Une application desktop est une application développée sur ordinateur et pas sur mobile/tablette.
Retour rapport

Dette technique

Quantité de travail pour mettre à jour un logiciel.
Cette quantité de travail est calculée via plusieurs facteurs: le manque de documentation technique est un facteur négatif dans notre cas. D'autres facteurs existent comme la présence de tests, de normes ou d'autres types de documentation avec par exemple la documentation fonctionnelle et la documentation architecturale que nous ne définirons pas ici.
Retour rapport

Documentation Technique

Documentation d'un logiciel faite par les développeurs pour les développeurs.
Cette documentation est produite à partir du code et des commentaires écrits par les développeurs. Elle est destinée aux autres développeurs pour leur permettre de comprendre le programme sans devoir étudier le code source.
Retour rapport

Lead Developer

Un développeur qui est responsable technique de l'utilisation d'un langage dans l'entreprise et qui définit les bonnes pratiques liées à celui-ci.
Retour rapport

Product owner

A pour rôle d'être le lien entre le client et les développeurs d'une application; présente et teste les fonctionnalités de celle-ci et recueille les retours des clients.

Retour rapport

Recette usine

Liste des tâches à effectuer sur une application, fournie au product owner, pour qu'il puisse vérifier que toute l'application fonctionne comme prévu par les développeurs.

Retour rapport

Sprint

Dans les méthodes Agiles, un sprint est une période courte qui dans le cas d'EAMS a une durée de deux semaines et dans laquelle les développeurs vont réaliser des tickets prévus par le product owner (un développeur qui représente le client) et le "scrum master". Un développeur doit allouer 80% de son temps au travail et 20% aux tâches annexes comme la veille technologique.

Retour rapport

Tickets

Une tâche planifiée dans le cadre d'un sprint ayant un temps estimé, dans le cas d'EAMS, sur une échelle suivant la suite de Fibonacci. Par exemple, un ticket de score de 1 est estimé à une heure de travail. Un ticket avec un score de 8 durera deux jours.

Retour rapport

Veille technologique

Consiste à s'informer et se former sur des nouvelles technologies.

Langages & Bibliothèques

Angular

C'est un framework (bibliothèque) typescript qui est une réécriture complète du framework AngularJs destiné à la production d'applications web. Il est développé par Google et a une communauté en forte croissance à l'heure actuelle.
Retour rapport

ExtJS:

C'est un framework (bibliothèque) javascript permettant de faciliter la construction d'une application web interactive. Il est destiné à être utilisé sur la partie cliente d'une application. Un client d'une application ne peut avoir un accès direct qu'à cette partie cliente.
Retour rapport

JAVA:

Langage de programmation Objet, ce n'est pas Javascript.
Retour rapport

JAVAFX

Bibliothèque permettant de créer des interfaces graphiques en Java.
Retour rapport

Javascript

Langage de programmation destiné à améliorer le dynamisme d'un site web.

Markdown

Langage destiné à mettre en forme un texte.
Retour rapport

Tooltips

Indication visuelle apparaissant lorsque l'on passe la souris sur un élément d'une application, permettant de comprendre l'utilité de cet élément.
Retour rapport

Typescript

Surcouche du langage Javascript créé par Google; langage de programmation Objet qui force l'utilisateur à respecter certaines règles. Ce langage permet de produire du code javascript.
Retour rapport

Outils divers

Docker

C'est un outil permettant de simuler un système d'exploitation comme Windows sans les inconvénients de consommation de mémoire et d'énergie qu'implique la machine virtuelle.
Retour rapport

dépôt git

Un dépôt comportant toutes les versions d'une application, généralement sauvegardé dans un serveur distant. Les versions peuvent être séparées sous forme de "branches". Chez EAMS, les développeurs travaillent sur une branche appelée "develop" et une fois le produit prêt, ils l'ajoutent sur une branche nommée "master".
Si plusieurs développeurs travaillent sur une même branche, ils vont créer chacun leur branche à partir de celle-ci puis, une fois terminé, ils doivent faire la fusion de leur branche avec la branche develop. Cette fusion peut suivre un protocole et dans ce cas on appelle ça une "Pull Request", définie plus loin.
Retour rapport

IDE

Logiciel d'édition de code ayant de nombreuses fonctionnalités utiles au développeur.
Retour rapport

Json

Format de sauvegarde de données qui permet d'écrire les données sous forme de chaînes de caractères simples.
Retour rapport

Machine virtuelle

Simule une machine entière sur votre ordinateur avec un système d'exploitation bien défini.
Retour rapport

Pull request

Demande de fusion d'une branche d'un dépôt git vers une autre. Usuellement, un second développeur va vérifier qu'il n'y a pas de conflit entre les deux branches puis valider la requête.
Retour rapport

Parseur

Un outil qui va reconnaître une certaine structure dans un texte et donc produire un résultat en fonction des textes qu'on lui fournira.
Retour rapport

SQLite

Base de données pour mobile.
Retour rapport

Annexes

Emploi du temps

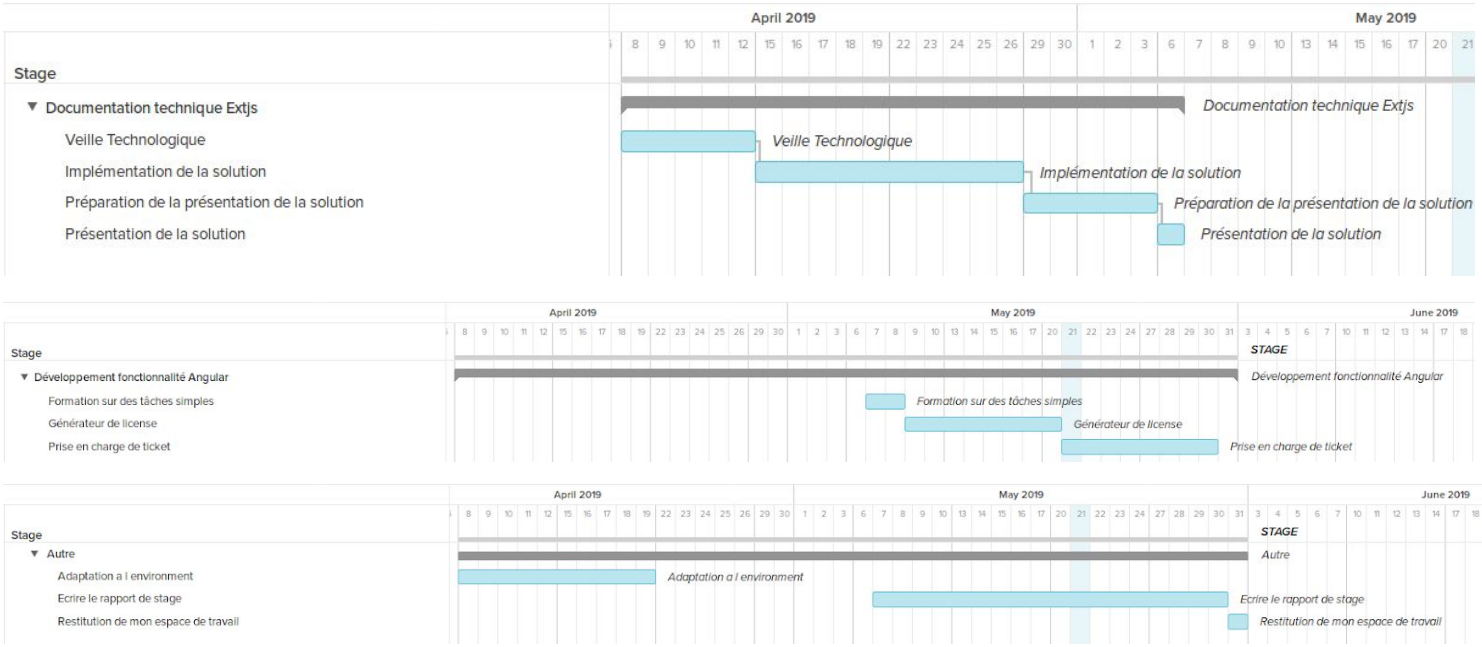
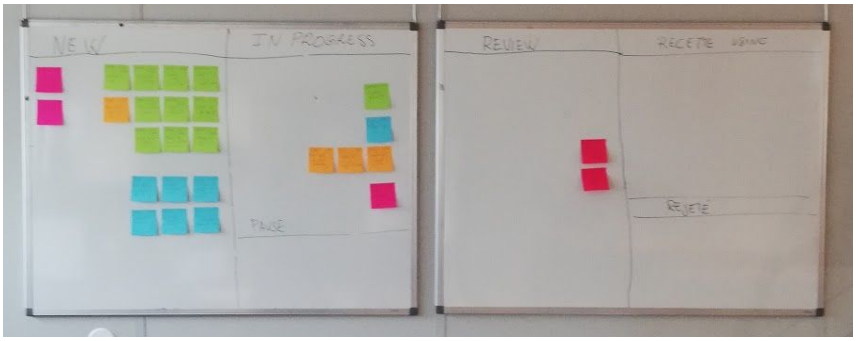


Tableau représentant le sprint en cour



[Retour rapport]

Exemples commentaires destinés au parseur:

```
/**
 * @private
 * @locale
 */
sDetail: 'Détail',
```

Retour rapport

Exemples JSDoc

Code à documenter:

```
/**
 *Descriptive text
 *@class nameOfClass
 */
Ext.define('nameOfClass',{
    //...
    /**
     *Description of the function
     *@param a {integer} explication of what a do
     *@param b {integer} explication of what b do
     */
    coolFunction: function (a,b){
        //...
    }
    //...
}
```

Résultat:

`coolFunction(a, b)`

Description of the function

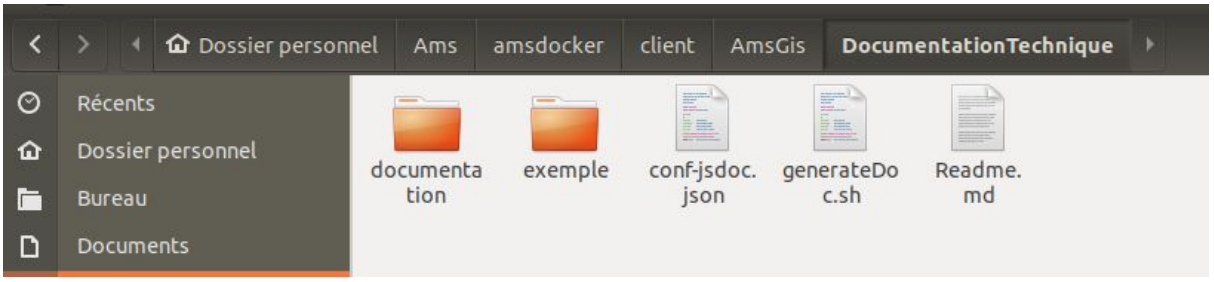
Parameters:

Name	Type	Description
a	integer	explication of what a do
b	integer	explication of what b do

Source: extjs_standard_exemple.js, line 12

Retour rapport

Exemple du dossier “DocumentationTechnique”



Code présents dans le dossier exemple

exemple_basique.js

```
/**
*Calcul the sum of a and b
*@param  a {integer} first parameter
*@param b {integer} second parameter
*@return {integer} la somme des deux parametres a et b
*/
function computeSomme(a,b) {
    var  somme =  a + b;
    return somme;
}

/**
*You don't need to pass lane
*@param a {integer} first parameter
*@param b {integer} second parameter
*@return {integer} a*b
*/
function computeProduit(a,b) {
    var produit =  a*b;
    return produit;
}
```

autre_ex_basique.js

```
/**
* this function try to find the max between a and b
*@param {integer} a first parameter
*@param {integer} b
*@return {integer} max between a and b
*@version 452
*@author Mathieu Levy
*/
function findMax(a,b) {
    if (a<b){
        return b;
    }

    return a;
}
```

extjs_standard_exemple.js

```
/**
 *Descriptive text
 *@class nameOfClass
 */
Ext.define('nameOfClass',{
    //...
    /**
     *Description of the function
     *@param a {integer} explication of what a do
     *@param b {integer} explication of what b do
     */
    coolFunction: function (a,b){
        //...
    }
    //...
});
```

Code de conf-jsdoc.json:

```
{
    "recurseDepth": 10,
    "source": {
        "includePattern": ".+\\.js(doc|x)?$",
        "excludePattern": "(^|\\/|\\\\)_",
        "exclude" : ["$"]
    },
    "sourceType": "module",
    "tags": {
        "allowUnknownTags": true,
        "dictionaries": ["jsdoc","closure"]
    },
    "templates": {
        "cleverLinks": false,
        "monospaceLinks": false
    }
}
```

Code de generateDoc.sh

```
#!/usr/bin/env bash

#Script de Documentation générée par jsdoc
#Option :
#-r permet de fouiller récursivement les sous-dossiers.
#   le chemin des fichiers à explorer.
#-d permet de choisir le dossier destination.
#-R permet de choisir le ReadMe.
#-c permet de specifier un fichier de configuration
#author: Mathieu Levy


#Declaration des variables
pathToSource="../classic/src/*"
pathToDoc="documentation"
pathToReadMe="../Readme.md"
pathToConfig="conf-jsdoc.json"

#Commande de génération de documentation technique
jsdoc -r $pathToSource -d $pathToDoc -R $pathToReadMe -c $pathToConfig $*
```

Le Readme utilisé comme première page d'une doc jsdoc:

Home

Jsdoc

you can read this file by running jsdoc exemple/* -R Readme.md in the repository DocumentationTechnique

Installation

You can install jsdoc by two way:

- Directly on your computer:

```
sudo npm install -g jsdoc
```

- Via the repository ams-docker (if you use docker):
  jsdoc is installed when you run

```
./reload.sh
```

Follow the usual steps in amsdocker/README.md.

How to work with jsdoc

Presentation

Jsdoc is a documentation generator for javascript.
It will check the code and build a documentation thanks to the comments in the application.
You need to respect some rules to generate a good documentation.
If you have never used jsdoc before, you can do the training, you can solve most troobleshoots with it.

Rules, Comment style and Standards

To generate the documentation you need to write your comments like this:

```
/**
*This is an exemple of generation of documentation for a class.
*@class AmsGis.view.theme.Tree
*@version 2
*@author Mathieu Levy
*/
Ext.define('AmsGis.view.theme.Tree', {
//...
/**
*Initialize  component
*Have an handler who do something
*@param a {integer} who give a number to the component
*@param b {string} who give the name of the component
*@version 1
*@author Mathieu Levy
*/
    initComponent: function (a, b) {
    //...
    }
//...
}
```

As you can see, there is a lot of commenting here.
The **absolute rule to follow** when you wrote a new function is to write atleast this:

```
/**
 *Descriptive text
 *@class nameOfClass
 */
Ext.define('nameOfClass',{
    //...
    /**
     *Description of the function
     *@param a {integer} explication of what a do
     *@param b {integer} explication of what b do
     */
    coolFunction: function (a,b){
        //...
    }
    //...
}
```

Moreover, i strongly recommand to acquaint the block tag *@author*.

Sometimes, in your file, you can see some wild:

```
/**
*@private
*@locale
*/
```

As long as there is nothing more, jsdoc will ignore them.

You can learn more here: How_to_document_your_code_properly
You can find more blocks Tag exemple in usejsdoc.org

Home

Classes

Tree
nameOfClass

Global

computeProduit
computeSomme
coolFunction
findMax
initComponent

Training

To follow this tutorial you need to be in the repository *DocumentationTechnique*.

- In the repository out you can see the documentation html generated by jsdoc.
- In the repository exemple you will find 4 js files. We will run jsdoc tool on them.

Go back to the repository *DocumentationTechnique*.
jsdoc is very easy to use:
if you try :

```
jsdoc exemple/*
```

the repository called "*out*" will be edited by jsdoc.
If you do this, the doc generated don't have Home page.
To select a Home page you need to use the *-R* option.
Try to generate the doc with:

```
jsdoc exemple/* -R Readme.md
```

Look at the result.
Now you know how to generate a beautiful jsdoc documentation.
You have an explanation of all the useful options in the Every day usage.
For more information : http://usejsdoc.org/

Every day usage

Update documentation

find generateDoc.sh in the DocumentationTechnique repository of your application run generateDoc.sh with the following command:

```
sh generateDoc.sh
```

Generate documentation

If you need the documentation for a new application, then you don't have a repository called *DocumentationTechnique*.
You need to copy the *DocumentationTechnique* repository from another application who already have it (ex: *AmsAdmin*). Then paste it in your application where you think the repository *DocumentationTechnique* belong You need to edit the file generateDoc.sh so the documentation will be correctly generated.
Beware to have the good path for the source and the readme.

The command to build the documentation is like this:

```
jsdoc -r PathToSource  -d PathToDoc -R PathToReadme -c $pathToConfig
```

PathToSource is the path to the files you want to be parsed
PathToDoc is the path where you want the documentation to be generated
PathToReadme is the path of the main page of the documentation
$pathToConfig is already defined, it's the path to the configuration file
The option *-r* mean go recursively in the repository. (depth = 10 in the configuration file)

Troubleshoot

- You need npm to use jsdoc.
  To install npm on ubuntu :

  ```
  sudo apt-get install npm
  ```

- You will maybe see a lot of error while jsdoc parse your files.
  If a documentation is generated then have no worries, jsdoc go through bad commenting and warn you.

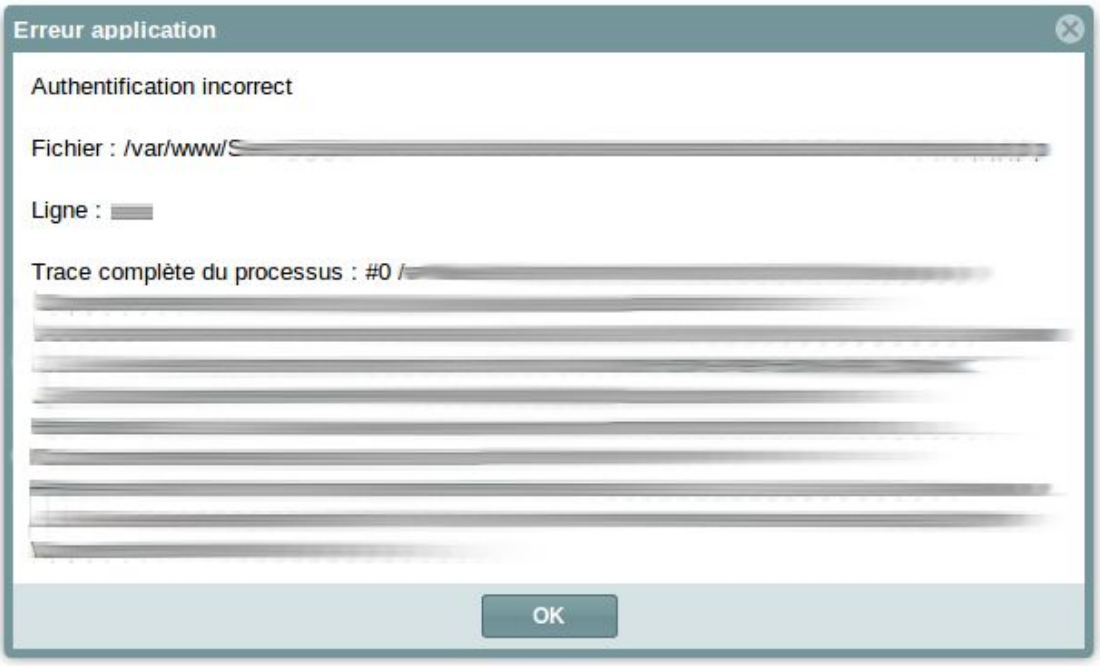
- Some files will bug very hard with a method called *getStorageKey* like this:

  ```
  getStorageKey The Following combination is used to create isolated local storage keys '_ext' is used to scope all the local storage keys
  that we internally by Ext 'location.pathname' is used to force each assets to cache by an absolute URL (/build/MyApp) (dev vs prod) 'url'
  is used to force each asset to cache relative to the page (app.json vs resources/app.css) 'profileId' is used to differentiate the builds
  of an application (neptune vs crisp) 'Microloader.appId' is unique to the application and will differentiate apps on the same host
  (dev mode running app watch against multiple apps)
  ```

  try to **start** the documentation generation in a **deeper repository**. I don't have other solution.
  (like instead of start in AmsAdmin, try to start in AmsAdmin/classic/src/)

Correction authentification en ExtJS

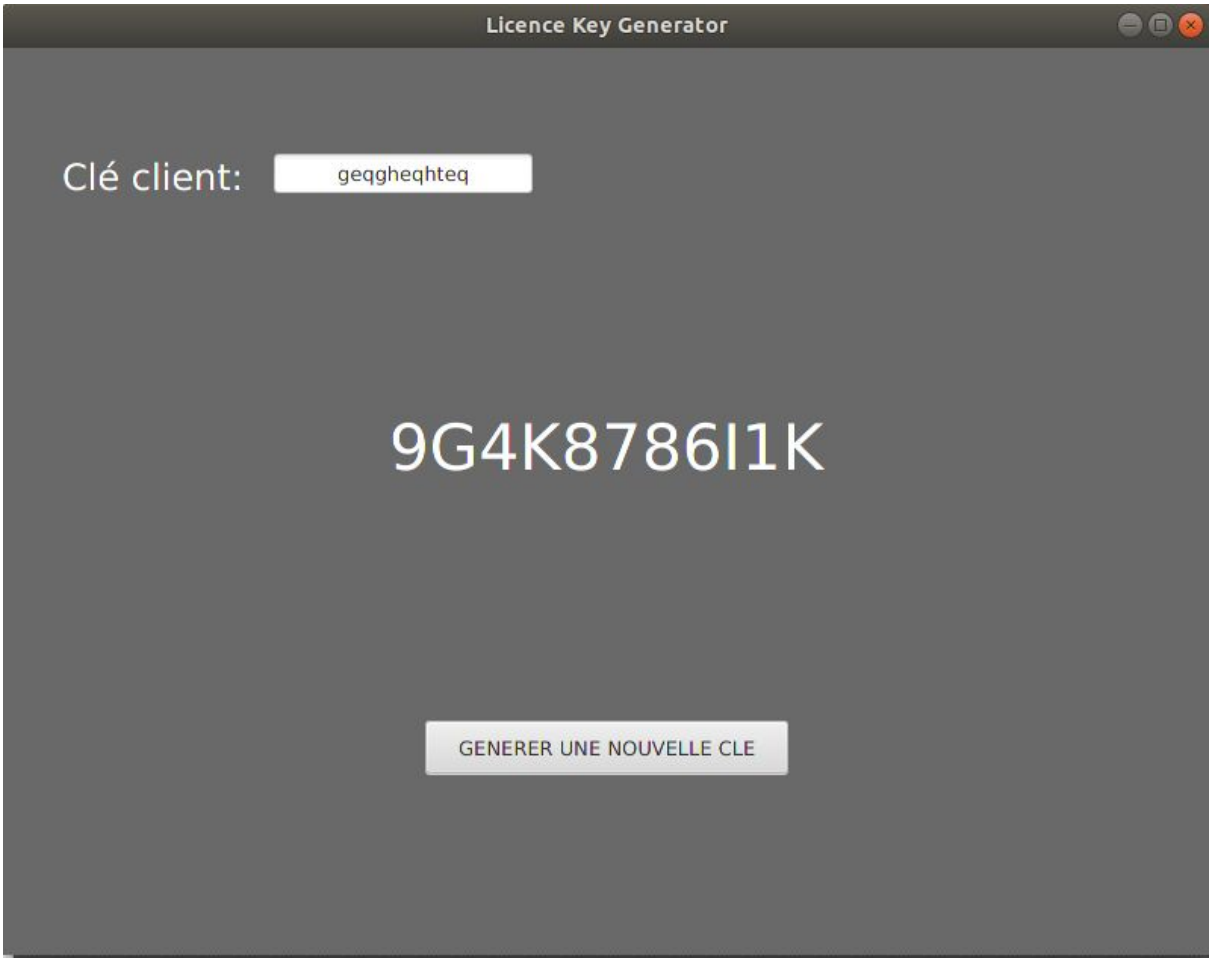
Départ: (L'erreur a dû être masquée pour raison de sécurité)



Résultat: (incorrect est faux car dépendant d'autres facteurs que je ne pouvais modifier)

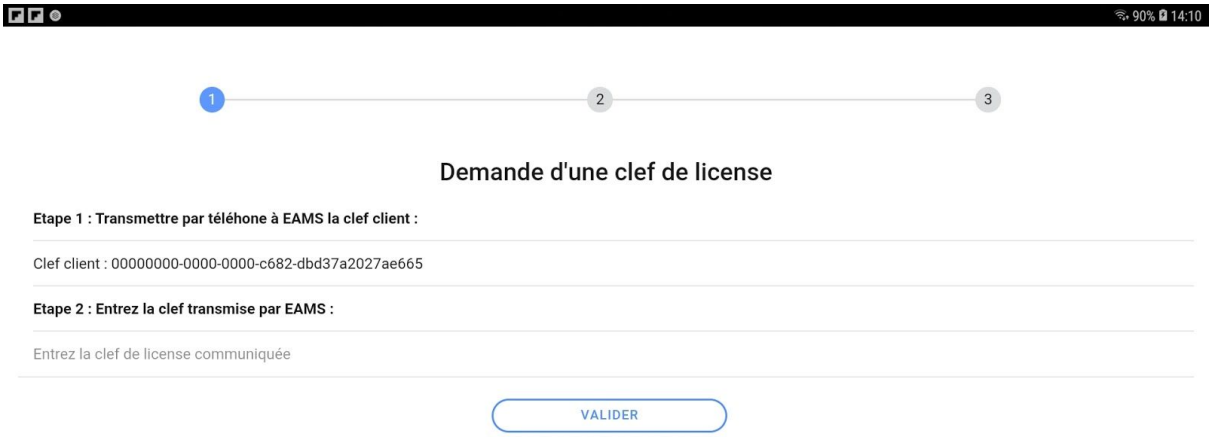
Retour rapport

Logiciel générateur de clé de licence



(suite page suivante)

Interface demandant une clé de licence



Retour rapport