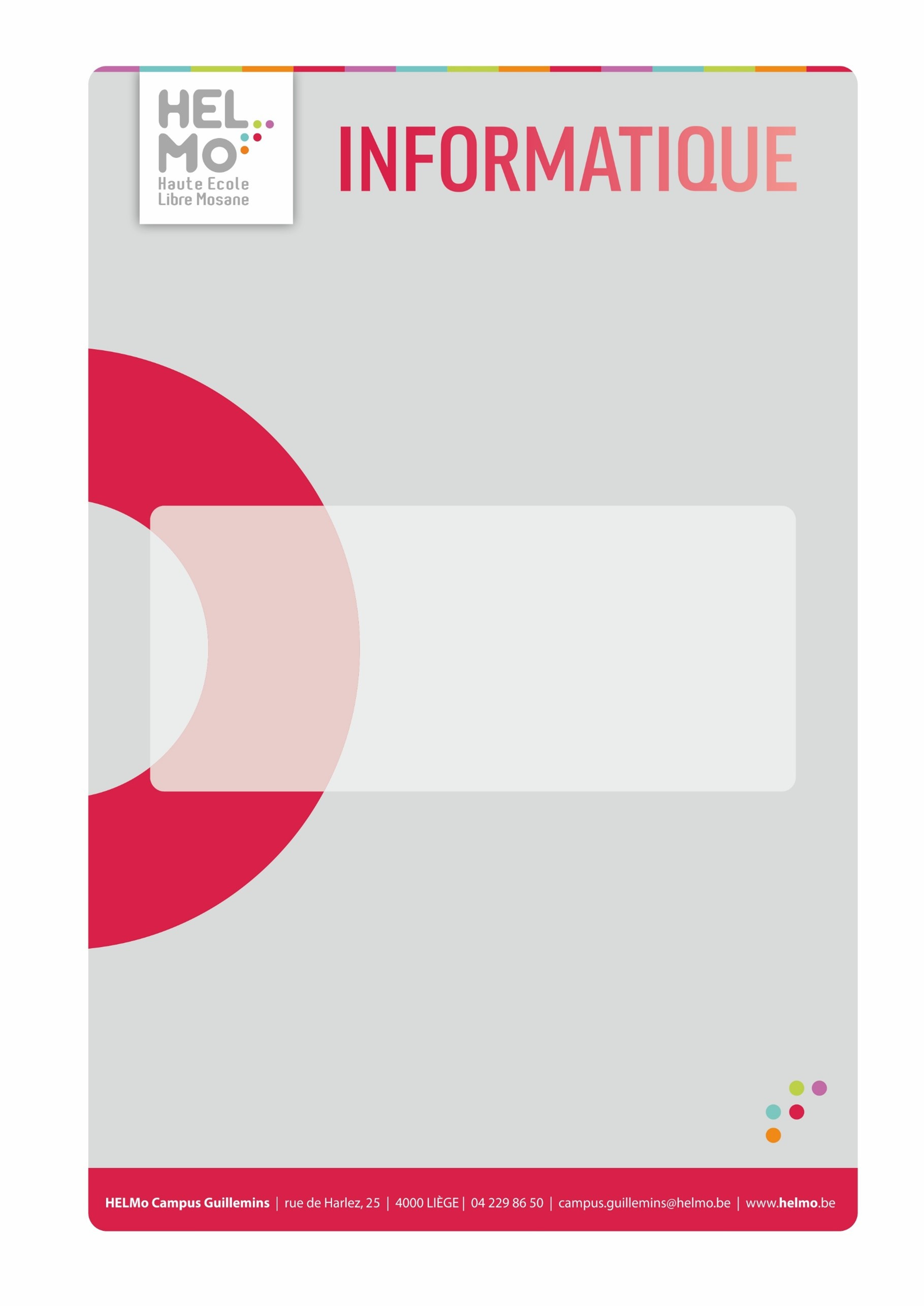
****

Mémoire de fin d’études

en vue de l'obtention du titre de

**Bachelier en Informatique   
orientation Développement d’applications**

Année académique 2024-2025

**! Veuillez effacer les infos en fluo après rédaction**

**Titre centré**

**(titre exact qui paraîtra sur le diplôme)**

|  |  |
| --- | --- |
| **TECHNORD**  Rue de la Lys, 21 7500 Tournai | Présenté par  **Zachary VANVLASSELAER** |

Contents

[1. Présentation de l’entreprise 4](#_Toc197856077)

[1.1 L’entreprise 4](#_Toc197856078)

[1.2 L’équipe 4](#_Toc197856079)

[2. Présentation du TFE 4](#_Toc197856080)

[2.1. Avant 4](#_Toc197856081)

[2.2. La problématique 5](#_Toc197856082)

[2.3. Le projet 5](#_Toc197856083)

[2.4. Les objectifs 5](#_Toc197856084)

[3. Méthodologie 6](#_Toc197856085)

[3.1 SCRUM 6](#_Toc197856086)

[3.2 Jira 6](#_Toc197856087)

[3.3 Intégration continue 8](#_Toc197856088)

[4. Analyse 8](#_Toc197856089)

[4.1 Fonctionnalités 8](#_Toc197856090)

[4.1.1 Gestion des facteurs 9](#_Toc197856091)

1. Présentation de l’entreprise

1.1 L’entreprise

Technord est une entreprise de service spécialisé dans les domaines de l’électricité, de l’automation, de l’IoT, de l’informatique industriel et de l’intelligence artificielle. Elle s’est donnée comme mission d’apporter des solutions fiables et efficaces sur le long terme à ses clients, venant principalement du monde de l’industrie.

L’une des valeurs fondamentales de Technord est son amour et sa confiance du client. En effet, l’entreprise met beaucoup d’importance sur sa relation avec celui-ci. D’autres lignes directrices sont sa vision, son agilité ainsi que son expertise, sans bien évidemment oublier de garder une touche d’humour et de fun.

Elle est composée de plusieurs centaines d’employés à travers différents pays d’Europe. Son siège social se situe à Tournai, en Belgique.

Son chiffre d’affaire dépasse les 80 millions d’euros, amassant plus d’1 millions d’euros de bénéfices par an.

1.2 L’équipe

Pour ma part, je vais évoluer sur le site de Seraing*.* C’est un lieu important car il est proche des industries avec lesquelles Technord interagit au quotidien.Le bâtiment est divisé en 2 étages, représentant les 2 équipes, les 2 spécialités de l’endroit : l’automation et l’informatique industrielle MoM[[1]](#footnote-1) / MES[[2]](#footnote-2). J’ai intégré cette dernière avec 2 autres stagiaires, formant une team de 11 personnes travaillant main dans la main.

L’équipe est dirigée par Kevin Putzeys, responsable projet, qui s’assure que la réalisation des applications se fait en adéquation avec les attentes des clients. Ce sont lui et Alessandro Missoul, un IPS[[3]](#footnote-3), qui vont servir de principal intermédiaire entre les clients et les développeurs, bien que certains interagissent directement avec les concernés en cas de besoin.

Les 2 équipes sont entièrement indépendantes et s’occupent d’objectifs et de clients différents. Malgré cela, ils leur arrivent de collaborer sur certains projets communs.

Les interactions avec les autres départements sont majoritairement limitées à l’équipe d’informatique industrielle principale, située à Tournai. Celle-ci comprend des experts dans des domaines spécifiques que le site de Liège ne possède pas. C’est donc vers eux que l’on se dirige en cas de problème plus important.

2. Présentation du TFE

2.1. Avant

Tout commence lorsqu’un client de Technord introduit une demande pour la réalisation d’une application aidant à la réalisation d’ouverture de chantiers. En effet, celles-ci peuvent devenir facilement compliquées à mettre en place proprement au vu du grand nombre de variables à prendre en compte.

Pour répondre à ce besoin, l’équipe MES de chez Technord a développé une application web en ASP.Net avec l’aide du framework Telerik qui s’occupe de faciliter tout ce processus.

Durant les années qui vont suivre, d’autres clients vont faire des demandes similaires, ce qui va conduire Technord à, dès lors, proposer l’application déjà existante à tous ceux le voulant sous forme de « package », offrant en plus de la personnalisation et une initialisation des données. Ces clients vont aussi demander de nouvelles fonctionnalités, qui vont être petit à petit ajoutées au projet initial.

2.2 Contexte sur l’ouverture de chantier

Mais alors, qu’est-ce qu’une ouverture de chantier et en quoi est-ce important ?

Tout d’abord une ouverture de chantier est un document que chaque entreprise voulant entreprendre un chantier doit rédiger. Ce document a une grande importance légale. En effet, un chantier possède plusieurs possibles dangers qui, dans le pire des cas, peuvent mener à d’importants dommages physiques ou matériels, menant parfois à un procès.

L’ouverture de chantier a donc pour but d’établir à l’avance ces différents risques afin de mettre en place toute une série de moyens de préventions pour les limiter un maximum. Une bonne ouverture de chantier permet donc de dédouaner une entreprise en cas de pépin en montrant que celle-ci a tout mis en place pour éviter l’incident.

Plus d’info sur ouverture de chantier

2.2. La problématique

L’application a été développée sur plusieurs années, par différentes personnes et avec des technologies déjà relativement anciennes pour l’époque. De plus, en essayant de répondre à certains besoins trop vite, elle fut réalisée sans forcément porter attention à la bonne réalisation des choses, la rendant pas toujours fiable et peu attirante visuellement. Par ailleurs, le processus de création s’étant fait de manière dispersée, le code de l’application est désordonné, rendant l’ajout de nouvelles fonctionnalités plus difficile que ce que ça ne devrait l’être. Pour toutes ces raisons, l’équipe de Technord a décidé de tout recommencer de zéro, avec de nouvelles technologies et une interface grandement améliorée.

2.3. Le projet

Ce qui nous a été demandé, à mon coéquipier et moi, est de migrer l’entièreté de l’ancienne solution monolithique vers une application composé de deux services comme l’est maintenant courant : un front-end et un back-end.

Le front-end permettra à l’utilisateur d’interagir avec les fonctionnalités de l’application et sera réalisé en Angular à l’aide du framework Kendo.

Le back-end servira d’intermédiaire entre l’interface visuelle et la base de données et s’occupera du traitement des requêtes. Il sera réalisé en .Net Framework avec le langage de programmation C#.

La base de données restera majoritairement inchangée pour permettre une installation simple de la nouvelle solution chez le client.

2.4. Les objectifs

D’abord, il est très important que la totalité des fonctionnalités de l’ancienne solution soit disponibles sur la nouvelle. Il est question fournir une version améliorée aux clients. L’application sera donc testée rigoureusement afin de ne pas créer une perte de productions chez ces derniers.

Ensuite, comme déjà mentionné, la nouvelle application sera un renouvellement visuel. La barre sera donc grandement relevée en ce qui concerne le design de l’interface.

Enfin, il devra être possible de facilement ajouter de nouvelles fonctionnalités au projet, que ça soit dans un future lointain ou non. Certains ajouts vont d’ailleurs être proposés comme tâches bonus aux stagiaires en fonction de l’avancement du stage.

3. Méthodologie

3.1 SCRUM

Technord utilise la méthode SCRUM afin de mener à bien ses projets. Ceux-ci sont divisés en plusieurs petites tâches réalisables par une seule personne qui composeront ce que l’on appelle le backlog. La réalisation d’un projet se fait durant des sprints, qui sont des unités de temps arbitraires choisies au début du projet. Dans notre cas, chaque sprint dure 2 semaines, divisant notre stage en 7 sprints. La première semaine n’est pas prise en compte, ayant majoritairement servi à se mettre en place, à analyser la solution existante et à se familiariser avec l’équipe.

C’est durant une réunion appelée le « Sprint Planning », faites à chaque début de sprint, que l’on attribue les tâches à chacun. Les 2 maîtres de stage proposent des tâches intéressantes à faire pour le prochain sprint et les 2 stagiaires donnent leur avis sur la faisabilité de celles-ci. Il est important d’attribuer à chaque tâche un certain nombre de story points, qui est une unité arbitraire représentant le temps estimé que prendra la réalisation de cette dernière.

A la fin d’un sprint, les membres de l’équipe organise une autre réunion, nommée sprint review, dans laquelle chacun pourra exprimer son ressenti par rapport au travail réalisé. Les points difficiles seront passés en revue ainsi que les raisons expliquant ces difficultés. Les story points seront aussi comparé à la véritable durée de chaque tâche. Tout ce processus permet d’améliorer le prochain sprint planning, afin d’estimer de manière plus précise le travail qui pourra être accompli.

Daily reviews

3.2 Jira

Afin de mener à bien leurs projets, Technord utilise Jira, une application très populaire permettant la gestion des projets, des incidents et des bugs développé par Atlassian. Elle se base sur un principe de ticket, correspondant à une tâche SCRUM, et propose plusieurs manières d’organiser ceux-ci.

Tout d’abord, avant même notre arrivée chez Technord, un employé chez Technord a passé l’ancienne application en revue afin de créer la majorité des tickets dont que nous devrons réaliser au cours de notre stage. Ces tickets ne sont pas fixes et il va nous arriver de les modifier ou d’en ajouter de nouveaux selon les aléas du développement.

Ensuite, à chaque « Sprint Planning », les maîtres de stage ajoutent les tickets au sprint créé sur Jira. Ainsi, ceux-ci sont affiché sur la page « Tableaux » de Jira, qui est la page où Noah et moi allons passer la majorité de notre temps. Cette page permettre de répartir les tickets du sprint sous formes de différentes colonne intitulées qui représentent l’état d’avancement du ticket.

Les différentes étapes du processus sont les suivantes : « Attente d’information », « A faire », « En cours », « A corriger », « En attente de déploiement », « A valider », « Validé ». Chaque ticket ne passe pas forcément par chaque colonne, mais cette organisation permet d’établir un principe bien pratique : plus un ticket est avancé dans le processus, plus il est important. C’est pour cette raison que « A corriger » est après « A faire », étant donné qu’il est préférable de d’abord faire des corrections avant de se lancer dans quelque chose de nouveau, afin d’éviter de se disperser. Je vais maintenant vous détailler rapidement chaque étape.

« Attente d’information » signifie qu’il manque quelque chose au développeur pour pouvoir réaliser la tâche. Cela peut être, évidemment, une information, mais aussi plus rarement lorsqu’il faut attendre la réalisation d’une autre tâche. La majeure utilité de cette colonne est pour les tickets demandant des questions aux maîtres de stages qui ne sont pas disponible sur le moment même.

« A faire » est relativement simple et directe : ce sont les tickets qui ont été ajouté au sprint lors du « Sprint Planning » qui n’ont pas encore été réalisés.

« En cours » représente les tickets commencé mais pas encore fini, ce qui permet de tenir au courant nos maîtres de stages de notre avancement.

« A corriger » est pour les tickets qui ont été réalisés mais qui présentent un manque de fonctionnalité ou des bugs. Ce sont principalement les maîtres de stages qui vont déplacer les tickets dans cette colonne, mais il va nous arriver de plus en plus souvent au cours du stage à Noah et moi de faire de même.

« En attente de déploiement » sont les tickets qui ont été entièrement réalisé, mais pas encore déployé sur la machine de test. En effet, nos maîtres de stages ne testent pas directement nos tickets sur notre environnement de développement. A la place, on nous a assigné à Noah et moi une machine virtuelle sur laquelle on déploie notre application comme on la déploiera chez un vrai client. Cela permet aux développeurs de travailler en toute sérénité sans devoir se soucier de casser quelque chose en train d’être testé. Ce système s’appelle de l’intégration continue, que je détaillerai dans le chapitre suivant.

« A valider » est la colonne sur la laquelle se concentre principalement nos maîtres de stages. Ce sont les tickets compris ici qu’ils vont tester afin de le mettre dans une autre colonne correspondant à leur satisfaction avec la réalisation de la tâche.

« Validé », finalement, contient tous les tickets qui ont été validés par nos maîtres de stages.

Ce système des différentes étapes n’a pas été réalisé tout de suite, ou en tout cas pas d’une manière aussi détaillée. Nous l’avons mis en place pour répondre à un problème qui s’est posé durant les premières semaines du stage : un manque de test. En effet, Noah et moi étant habitués aux environnements de développement des travaux de groupes pour l’école, nous ne prenions pas assez au sérieux toute la phase de test des fonctionnalités. Cela avait pour conséquence que peu de nos tickets passaient les standards de nos maîtres de stages.

Afin de régler cette problématique, nous avons donc mis en place le système que je vous ai présenté auparavant, celui-ci présentant 3 phases de test différent au cours de son cycle de développement. La première lorsque l’on finit une tâche, Noah et moi s’assurons qu’elle marche comme on pense qu’elle le devrait avant de passer le ticket correspondant en « En attente de déploiement ». Ensuite, lorsque l’on déploie la solution sur la machine virtuelle, on passe tous les tickets en revue de manière plus poussée avant de les faire passer à l’étape suivante. Cela permet une arrivée de tickets relativement raffinés aux maîtres de stages, ce qui rend le cycle des tickets plus linéaire. Enfin, comme mentionné précédemment, les tickets sont testés une dernière fois par les maîtres de stages, ce qui rend la découverte d’un bug après coup très rare.

3.3 Intégration continue

**Comme mentionné plus tôt, nous le principe d’intégration continue afin de mener à bien notre projet de stage. Cependant, nous dévions un peu de l’idée de base, qui est de fournir des versions provisoires de l’application au client tout au long du développement afin qu’il puisse donner ses retours. En effet, étant donné que les clients ont déjà une application existante qui marche et qu’il nous a fallu beaucoup de temps avant d’apporter autant de fonctionnalité que celle-ci au vu de la taille du projet, ce n’est pas à eux que nous présentons l’avancement de l’application.**

**Comme mentionné précédemment, ce sont nos maîtres de stages qui s’assurent de donner des retours sur ce que nous avons réalisés en déplaçant les tickets dans les colonnes adéquates et en écrivant des commentaires pour nous guider. M. Putzeys, le maître de stage de Noah et gestionnaire de projet, est la personne en charge de l’ancienne application d’ouverture de chantier depuis plus de 5 ans. Il va aussi souvent à la rencontre des clients, ce qui les rend très apte à savoir leur attentes et les choses qui vont les bloqués. C’est donc lui qui jouera le rôle de client tout au long de notre stage.**

4. Analyse

4.1 Fonctionnalités

Au vu de la taille relativement conséquente qu’était cette application d’ouverture de chantier, Technord a décidé de remettre ce travail à 2 personnes. L’HELMo ne voulant pas que 2 étudiants aient exactement le même stage, les maîtres de stages se sont concertés afin de diviser les fonctionnalités développées par Noah et moi de la manière la plus équitable possible. Il a donc été décidé que mon collègue s’occuperait plutôt de la partie centrale de l’application, l’ouverture de chantier en elle-même, tandis que je serai taché de gérer tous les éléments gravitant autour de cette dernière.



Ces tâches peuvent être divisées en plusieurs sections : la gestion des facteurs, la gestion des sous-traitants, l’encodage et réalisation des questionnaires et une petite partie de la gestion des OVC. Je vais expliquer chacune de ces sections en détails à l’aide de « User Stories » et de commentaires.

### 4.1.1 Gestion des facteurs

Comme mentionné plus tôt une ouverture de chantier dépend de beaucoup de facteurs différents qui changent en fonction de l’ouverture de chantier. Il est donc très important de laisser la possibilité aux utilisateurs de gérer ceux-ci. Chaque facteur a donc une page désignée permettant au minimum les opérations CRUD[[4]](#footnote-4), voire plus si nécessaire. Les « User Stories » se présentent de la manière suivante :

Pour les paramètres, les types de certifications, les types d’habilitations, les risques, les moyens de préventions, les emplacements, les permis, les rôles intervenants, les types d’évaluations, les catégories d’évaluations et les EPI :

* En tant qu’utilisateur, je souhaite pouvoir ajouter un facteur afin de pouvoir l’utiliser dans une ouverture de chantier.
* En tant qu’utilisateur, je souhaite pouvoir voir tous les facteurs à ma disposition afin de pouvoir apporter des changements à ceux-ci si nécessaire.
* En tant qu’utilisateur, je souhaite pouvoir modifier un facteur afin que ses changements se reflètent dans son utilisation dans une ouverture de chantier.
* En tant qu’utilisateur, je souhaite pouvoir supprimer un facteur afin de ne plus pouvoir l’utiliser.
* En tant qu’utilisateur, je souhaite pouvoir voir les facteurs supprimés afin de pouvoir potentiellement en restaurer un.
* En tant qu’utilisateur, je souhaite pouvoir restaurer un facteur afin de pouvoir l’utiliser de nouveau.

Il est important de noter que l’application dans son entièreté repose sur un système de droit. Pas tout le monde n’a le droit de faire ce qu’il veut. Chaque page et chaque action ont un droit qui leur est associé et seules les personnes ayant ce droit peuvent accéder à cette page ou réaliser cette action. Dans la majorité des cas, le fait de pouvoir voir les éléments supprimés et donc restaurer un facteur est limité aux administrateurs, qui sont les seules avec les droits nécessaires. Cependant, l’application permet aussi de gérer les droits associés à chaque type d’utilisateur. Il est donc entièrement possible qu’un utilisateur puisse restaurer un facteur si le client le décide. C’est donc pour cette raison que j’ai généralisé la personne faisant l’action à « utilisateur » dans les « User Stories ».

Bien évidemment, la gestion des facteurs ne se limitent pas qu’à ces simples opérations. La majorité des pages proposant des fonctionnalités supplémentaires. Je vais donc vous détailler celles-ci ainsi que vous donner plus de contexte pour chaque facteur.

La gestion des paramètres n’est pas vraiment ce à quoi on s’attend lorsque l’on parle de paramètres d’application. C’est plutôt une liste d’option possible que l’on peut choisir pour une famille de paramètre donnée. Cette fonctionnalité permet donc de définir par exemple tous les types d’habilitation, ou toutes les familles de risques, etc.

Schéma de comparaison

Pour cette page, on peut ajouter comme « User Story » :

* En tant qu’utilisateur, je souhaite pouvoir changer la famille de paramètre actuelle afin que les opérations que je fais sur les paramètres s’appliquent à celle-ci.

La gestion des types d’habilitations et des types de certifications suivent le même principe, étant seulement différentes dans la chose à laquelle elles s’appliquent. Ce sont des agrégations qui permettent de s’assurer qu’un membre du personnel (pour les habilitations) ou une société (pour les certifications) soit en mesure de répondre à un potentiel risque. Ces agrégations doivent potentiellement être validées par une personne compétente avant qu’elles prennent effet. Il est donc possible de définir qu’un type d’agrégation donné demande une validation et tout changement à cette valeur doit se refléter dans les agrégations héritant de ce type. Chaque agrégation possède aussi une de fin de validité, qui peut être définie à l’aide de son type. Voici donc les « User Stories » à ajouter :

* En tant qu’utilisateur, lorsque je décide qu’un type d’agrégation ne demande plus de validation, je souhaite que chaque agrégation héritant de ce type soit automatiquement validée.
* En tant qu’utilisateur, lorsque je change la date de fin de validé d’un type d’agrégation, je souhaite que chaque agrégation héritant de ce type voit sa date de fin de validité changer en accordance avec le type.

La gestion des risques et des moyens de prévention est la seule exception à « une page par facteur » étant donné qu’ils sont intrinsèquement liés. En effet, un risque possède automatiquement des moyens de préventions, vu qu’il ne serait pas cohérent de commencer une ouverture de chantier avec un risque que l’on ne peut pas éviter. Il est aussi possible de lier des habilitations, des certifications et des permis à un moyen de prévention. Ainsi, lorsqu’un risque est présent sur une ouverture de chantier, chaque moyen de prévention associé à ce risque devra avoir ses agrégations remplies par quelqu’un ou quelque chose. Je vais donner un exemple afin de mieux illustrer mes propos. Imaginons un scénario simple avec le risque unique suivant :



Si une ouverture de chantier présente le risque « Chute d’objets », il est alors impératif qu’au moins une des personne travaillant sur cette ouverture de chantier est l’habilitation « Levage / Elingage » et que la société faisant l’ouverture de chantier possède le permis « Balisage ».

Par ailleurs, chaque risque est lié à une famille de risque, permettant de mieux organiser ceux-ci. De plus, on peut associer des fichiers à un moyen de prévention. Les « User Stories » suivante doivent donc être prise en compte :

* En tant qu’utilisateur, je souhaite pouvoir changer la famille de risque actuelle afin de pouvoir gérer les risques venant de cette famille.
* En tant qu’utilisateur, je souhaite pouvoir associer des habilitations, des certifications ou des permis à un moyen de prévention afin que ceux-ci soient pris en compte dans une ouverture de chantier présentant le risque auquel le moyen de prévention est associé.
* En tant qu’utilisateur, je souhaite pouvoir faire les opérations CRUD présentées précédemment à des fichiers associés à un moyen de prévention.

La gestion des emplacements et la gestion des permis se différencient des autres avec leur représentation sous forme d’arbre. Chaque emplacement ou permis peut posséder un emplacement ou permis enfant. Leurs pages demandent donc quelques adaptations pour que l’utilisateur puisse les utiliser en toute facilité :

* En tant qu’utilisateur, je souhaite pouvoir ajouter un permis/emplacement enfant à un autre afin de créer une structure sous forme d’arbre.
* En tant qu’utilisateur, je souhaite pouvoir voir les éléments enfant d’un permis/emplacement afin de visualiser la structure.
* En tant qu’utilisateur, je souhaite pouvoir étendre l’entièreté de la structure afin de pouvoir voir tous les permis/emplacement
* En tant qu’utilisateur, je souhaite pouvoir réduire toute la structure afin de ne voir que les permis/emplacements du premier niveau.

On peut aussi associer des moyens de prévention à un emplacement. Ainsi, l’ajout d’un emplacement à une ouverture de chantier lie aussi tous ses moyens de prévention à celle-ci, de la même manière qu’un risque le fait. Par ailleurs, il est aussi possible d’attacher des fichiers à un emplacement. Cela donne donc les « User Stories » suivantes :

* En tant qu’utilisateur, je souhaite pouvoir faire les opérations CRUD présentées précédemment à des moyens de préventions associé à un emplacement.
* En tant qu’utilisateur, je souhaite pouvoir faire les opérations CRUD présentées précédemment à des fichiers associés à un moyen de prévention.

La gestion des catégories et des types d’évaluations ressemblent un peu à celles des risques et des moyens de préventions dans leur lien fort. Ces gestions ont été séparées en 2 pages par soucis de clarté mais auraient très bien pu se faire en une seule. Il est possible de laisser une évaluation sur une société ayant participé à une gestion de chantier. Ces pages de gestion permettent de définir sur quels critères ces évaluations portent. Un critère est représenté par un type d’évaluation, qui est lui-même associé une catégorie d’évaluation, ajoutant la « User Story » suivante :

* En tant qu’utilisateur, je souhaite pouvoir voir les types d’évaluations associés à une catégorie d’évaluation.

La gestion des EPI et des rôles intervenant restent fort simple, mise à part l’association d’une image à un EPI, ce qui donne la « User Story » qui suit :

* En tant qu’utilisateur, je souhaite pouvoir visualiser l’image que j’ai associée à une EPI

Toutes les gestions mentionnées dans ce chapitre ont un point commun : l’association à un site. En effet, un client possédant l’application peut posséder plusieurs sites voulant faire une ouverture de chantier. Cependant, chaque site ne fonctionne pas forcément de la même manière et n’utilise donc pas forcément les mêmes facteurs. Il est donc fondamentale de permettre à un utilisateur de dire quel facteur est présent sur quelle site. Sur l’ancienne application, un utilisateur ne pouvait associer que 2 valeurs possibles pour le site d’un facteur : le site de l’utilisateur qui s’est connecté ou « global » pour tous les sites. Cette gestion simple marchait bien lorsque le client possédait peut de site, mais pose problème si ce nombre devient trop grand. La nouvelle application étant prévue en partie pour un client ayant jusqu’à 50 sites différents, il a été essentiel de partiellement revoir ce système.

Il faut savoir qu’il existe plusieurs niveaux de droits sur l’application dont 2 qui nous intéresse ici : les administrateurs locaux et les administrateurs globaux. Les premiers gèrent le site de la même manière qu’auparavant, tandis que les seconds peuvent voir et gérer les facteurs pour n’importe quel site.

### 4.1.3 Encodage et réalisation d’un questionnaire

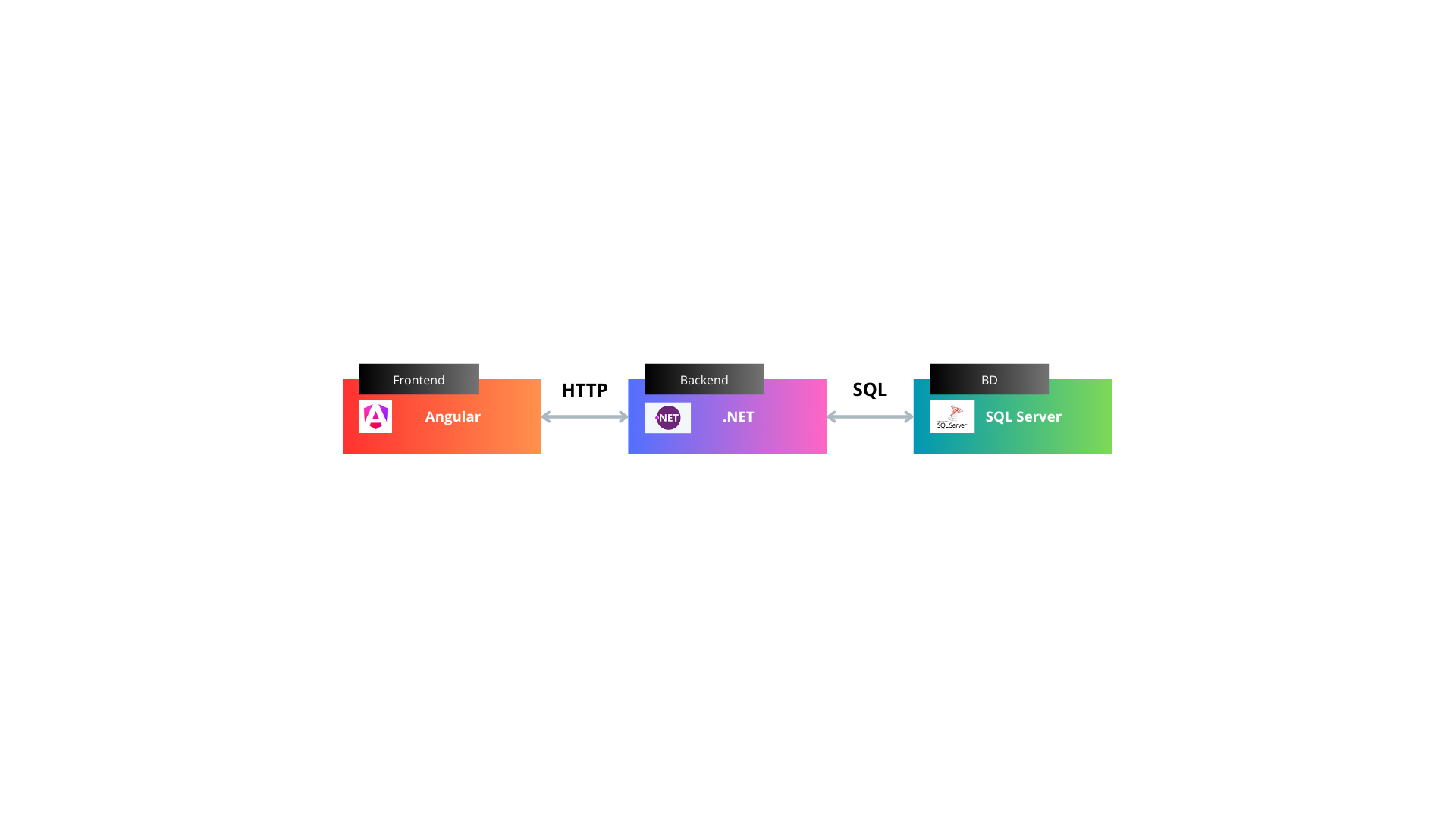
Une autre fonctionnalité importante proposée par le site est les questionnaires. Ils permettent à un membre du personnel de réaliser un petit de test dans le but d’obtenir une habilitation. Les clients intéressés par cette fonctionnalité créent alors un compte avec seulement les droits nécessaires à la réalisation d’un questionnaire et se connectent à celui-ci sur un écran d’accueil présent dans le bâtiment où travaillent les ouvriers. Ceux-ci peuvent alors s’identifier à l’aide de leur nom et de leur prénom afin de réaliser un test pour l’habilitation de leur choix. Plusieurs « User Stories » doivent être ainsi prises en compte :

* En tant qu’utilisateur membre d’une société sous-traitante, je souhaite m’identifier à l’aide de mon nom et prénom afin de réaliser un questionnaire.
* En tant qu’utilisateur ne faisant pas partie du système, je souhaite m’identifier à l’aide de mon nom et prénom afin de réaliser un questionnaire et, qu’à la fin de celui-ci, je sois ajouté comme membre du personnel de société de mon choix.
* En tant qu’utilisateur, je souhaite pouvoir visionner un document explicatif au début du questionnaire et confirmer ma lecture effective de celui-ci afin de continuer le test.
* En tant qu’utilisateur, je souhaite pouvoir visionner une image ou une vidéo associée à une question afin d’avoir plus de contexte sur celle-ci.
* En tant qu’utilisateur, je souhaite pouvoir répondre à chaque question du test afin d’essayer de réussir celui-ci.
* En tant qu’utilisateur, je souhaite pouvoir valider un questionnaire lorsque je l’ai fini afin de voir ma note et de potentiellement obtenir l’habilitation. .
* En tant qu’utilisateur, si je rate un questionnaire, je souhaite pouvoir réaliser de nouveau les questions que j’ai ratées.

Il est essentiel de noter que pour qu’une habilitation soit décernée à la personne réalisation le questionnaire, celui-ci obtienne une note de 100%. Il existe pour l’instant trois types de questions différentes possibles : à choix unique, à choix multiple et libre. Les réponses à ces questions sont évaluées automatiquement par le système et ne nécessite pas l’intervention d’une personne tierce. Les questions libres sont considérées comme toujours correctes et sont utiles lorsqu’elles sont mises en lien avec l’historique, que je détaillerai plus tard.

4.2 L’architecture logicielle

L’architecture initiale de l’application est l’architecture de base de toute application basée sur un frontend et un backend. Elle est très simple car toutes les données utilisées par l’application sont intérieurs à Technord. De plus, l’entreprise ne possède pas de service généralisé pour des choses comme l’authentification au la gestion des fichiers.



Le frontend de l’application, son interface graphique, est une Single Page Application, aussi appelée SPA. C’est une application web qui, au lieu de demander chaque page au serveur, charge l’entièreté de l’application en une requête et réécrit dynamiquement la page actuelle. Les interactions entre l’utilisateur et l’application sont donc beaucoup plus dynamique, ne demandant pas de temps de chargement à chaque fois.

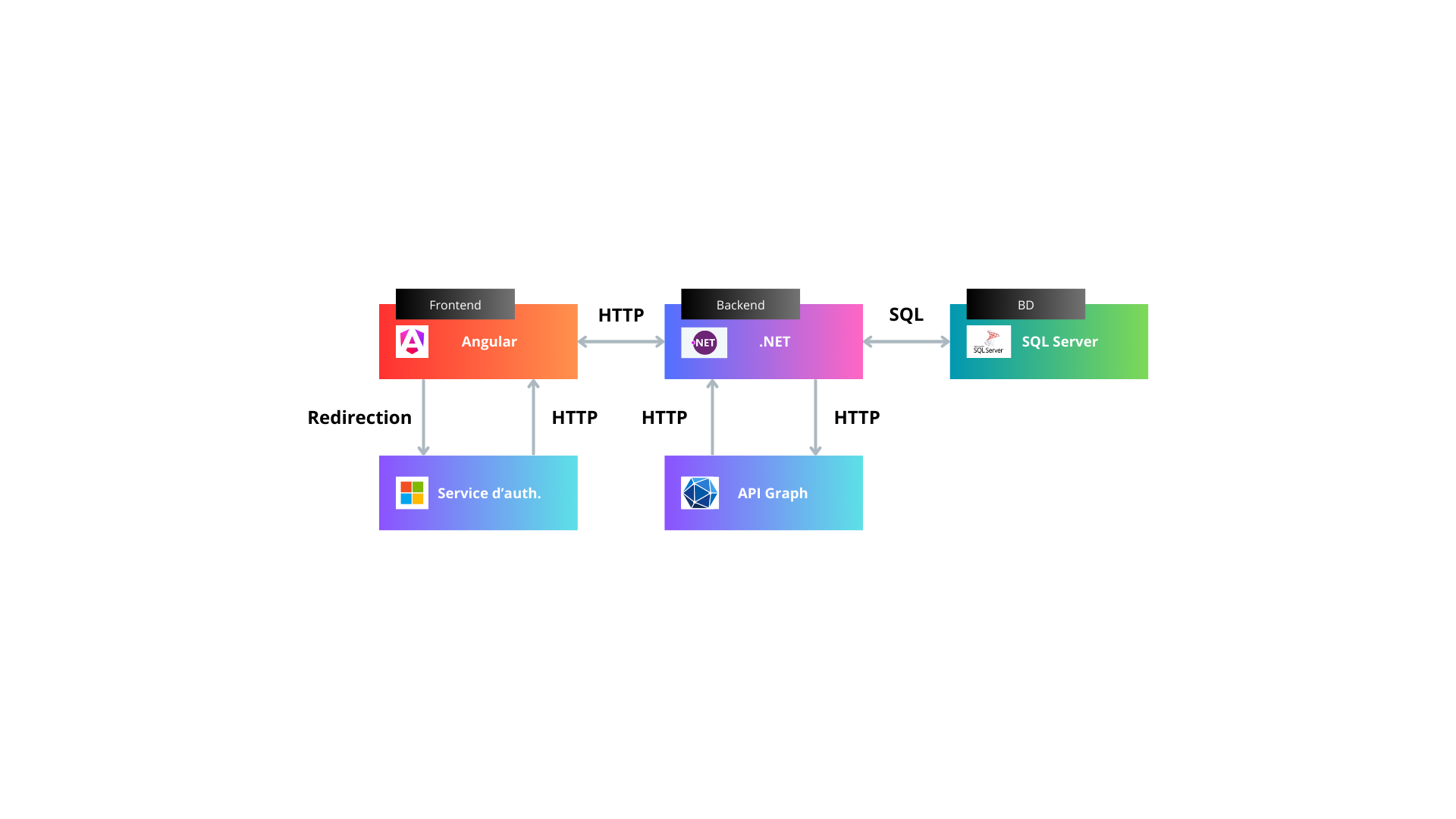
Mais comment fait l’application pour interagir avec les données de la base de données ? Elle envoie des requêtes HTTP au backend qui est une API REST, ce qui veut dire qu’elle est sans état. Chaque requête doit contenir toutes les informations nécessaires à sa réalisation. En fonction de la demande, le backend effectue des opérations afin de retourner une réponse HTTP au frontend.

La majorité du temps, cela demande d’envoyer des requêtes à la base de données. Ces requêtes utilisent chez Technord des procédures stockées, qui seront détaillées au chapitre suivant.

Pour mieux illustrer ceci, je vais donner un exemple simple. Imaginons que l’utilisateur veut voir les membres supprimés du personnel d’une société. Le frontend envoie alors une requête HTTP contenant l’ID de la société ainsi qu’un paramètre spécifiant la volonté de voir les éléments supprimés. Le backend réceptionne cette requête et vérifie que l’utilisateur a bien le droit de voir ces éléments. Si c’est le cas, il appelle une procédure stockée de la base de données qui renvoie les informations voulues sous forme de requête SQL au backend. Celui-ci à son tour envoie une réponse HTTP au frontend qui l’interprète et affiche les éléments à l’utilisateur.

Dans le cadre de l’ajout de nouveaux moyens d’authentification mentionnées précédemment, j’ai dû légèrement modifier cette architecture afin d’inclure des interactions avec les services Microsoft.

Pour l’authentification SSO, je dois d’abord rediriger l’utilisateur vers le service d’authentification Microsoft afin que l’utilisateur se connecte et que le service me renvoie un « Access Token » sous forme de réponse HTTP. Le frontend envoie alors une demande d’authentification au backend qui lui-même contacte l’API Graph de Microsoft. Cette API permet d’aller rechercher les informations concernant un utilisateur. Dans le cadre de l’application, ces informations sont les données de l’utilisateur (ID, Nom, E-mail, Prénom, Nom de famille, etc.) ainsi que les groupes auxquels il appartient. Le reste de l’opération se déroule comme une requête classique expliquée plus tôt dans ce chapitre



Pour le système d’invitation, lorsque qu’une authentification par MFA est nécessaire, le backend contacte les services de communication de Microsoft afin que ceux-ci envoie un message avec le code secret à l’utilisateur. Ce dernier peut ainsi donner le code au frontend qui envoie une 2ème requête au backend. Celui-ci vérifie alors l’identité de l’utilisateur et continue le processus d’authentification.



4.3 Les technologies utilisées

Pour la réalisation du frontend de l’application, nous avons utilisé du Javascript avec le framework Angular. Javascript a été étudié durant le cursus dans l’unité d’apprentissage *UE20 – Langages de script dynamique*.

Angular, quant à lui, est un framework spécialisé dans la création de SPA. Il a été développé par Google et est une réécriture complète de son prédécesseur : AngularJS. Il est basé sur un système de composants qui permettent de créer des parties d’interface graphique réutilisables. Ces composants sont divisés en 2 parties principales, l’HTML, décrivant le contenu de la page et le Typescript qui gère la logique derrière le composant. Pour permettre la communication entre ces 2 parties, Angular supporte de la liaison de données bidirectionnelle. D’autres aides au développeur sont comprises dans ce framework telles que les Directives[[5]](#footnote-5), la navigation et une CLI[[6]](#footnote-6).

Angular a été relativement récemment choisi comme nouveau standard pour le développement frontend par Technord. La principale raison derrière ce choix est la fiabilité de cette Technologie, étant utilisé largement depuis 2012-2013. C’est aussi le framework Javascript avec lequel l'équipe Technord était la plus familière. Lors de l’adoption du standard, l’entreprise a offert une formation à tous ses employés, permettant de mettre à jour ceux pas encore familier avec la technologie.

Afin d’accélérer le développement et de garder une expérience visuelle similaire à travers toutes ses applications, Technord utilise aussi le framework Kendo. Celui-ci est une librairie d’éléments graphiques près à l’emploi proposant de nombreuses fonctionnalités. Le lien fort entre Kendo est Technord existe depuis le vieux temps des applications ASP.NET. La librairie étant compatible avec Angular, l’entreprise a décidé de continuer de l’utiliser.

Dans la même idée qu’Angular, Technord possède un standard pour la réalisation du backend : .NET framework Web API 2. Cette technologie utilise du C# comme langage de programmation, vu dans l’unité d’apprentissage *UE36 – Programmation avancée*.

Web API 2 est donc un framework créé par Microsoft permettant de créer des services HTTP REST. Il fait partie de la plateforme ASP.NET mais se concentre spécifiquement sur la création d’API. Il permet de spécifier des « routes » qui sont des URL spécifiques que l’on peut appeler à l’aide de requêtes HTTP, retournant une réponse HTTP. Le corps de cette réponse contient les informations demandées à l’API et peut se présenter sous différents formats. Dans notre cas, ce fut du JSON. Pour la même raison qu’Angular, Technord a choisi cette technologie pour sa robustesse, étant populaire depuis 2013-2014. Elle propose aussi une authentification améliorée utilisable facilement : OAuth. Cette technologie commence cependant à un peu à se faire vieille, avec son remplacement en .NET Core publié en 2017. Technord est donc dans une phase de migration, se dirigeant vers cette nouvelle technologie qui n’a malheureusement pas pu être utilisé pour ce projet.

Enfin, pour gérer sa base de données, Technord utilise SQL Server, qui est un système de gestion de bases de données relationnelles.

SQL Server a été créé par Microsoft et est particulièrement adapté aux systèmes de grandes envergures pour les entreprises. De plus, il est facilement utilisable en lien avec d’autres technologies Microsoft, tels que .NET framework.

Afin d’interagir avec cette base de données, nous ne créons pas de requête SQL directement dans le back ni n’utilisons d’ORM[[7]](#footnote-7). A la place, des procédures stockées, aussi appelées SP, sont utilisés. C’est quelque chose qui n’a pas été abordé au cours mais présente de nombreux avantages. Il s’agit de scripts créés par les développeurs qui sont stockés directement dans la base de données. Pour faire un requête vers celle-ci, le développeur appelle la SP par son nom et lui donne des arguments. Cette dernière s’exécute alors et retourne la réponse. Les procédures stockées permettent tout d’abord de créer un standard, réutilisé dans chacune d’elles. Pour Technord, chaque SP possède des lignes de codes destiné à attraper une erreur et à l’ajouter comme valeur dans une table spécifique, ce qui permet une gestion des problèmes bien plus efficace. Ensuite, vu que la requête ne doit pas être envoyée par le réseau, cela diminue beaucoup la charge qui lui est imposée, améliorant les performances globales de l’application. Enfin, une fois la procédure crée, elle peut être utilisée par différents systèmes, ne demandant donc pas de recréer la requête de toute pièce à chaque fois.

4.4 Les outils utilisés

Comme vous l’aurez peut-être déjà remarqué, Technord est un grand utilisateur de technologies Microsoft. En effet, celles-ci offrent un grand nombre de services et de fonctionnalités différentes, avec des interactions facilitées entre elles. C’est donc logique que cet amour s’illustre aussi dans les outils utilisé par les développeurs.

Pour la gestion du code en équipe, Technord utilise Azure Repos. C’est un service permettant la collaboration des développeurs basés sur Git comme Github ou GitLab. Ces outils ont été présentés et utilisés lors du cursus.

L’édition de code se fait à l’aide de deux applications différentes chez Technord, en fonction du langage de programmation.

Pour la réalisation du frontend, nous avons utilisé Visual Studio Code. Ce logiciel est un éditeur de texte enrichi supportant une énorme quantité de langages différents. Il est largement utilisé autant par les particuliers que par les grandes entreprises notamment grâce à sa gratuité. Il est open-source, amélioré en continu et permet une grande personnalisation. C’est donc le choix idéal pour Technord afin de développer des applications en Angular. Ce développement est d’autant plus facilité par les plugins communautaires proposés publiquement qui enrichissent les fonctionnalités de l’éditeur.

Le backend, quant à lui, a été réalisé avec Visual Studio. C’est un IDE plus lourd que son petit frère, mais qui propose donc des fonctionnalités plus avancée. Ses avantages pour Technord se situent principalement dans ces template intégrées pour les projets Web API, ainsi que le système de comparaison de schéma. Ce dernier est très utile pour les projets communs et a été utilisé tout au long du développement de l’application. Il permet de mettre en évidence les différences entre les schémas de base de données et de synchroniser celles-ci. Vu que Technord se base sur l’utilisation de procédures stockées, Noah et moi avons dû souvent modifier nos bd. Cet outil a donc permit d’importer les changements effectués par l’un des stagiaires vers la base de données de l’autre facilement.

Afin de gérer nos base des données et justement modifié les SP, nous avons utilisé SQL Server Management Studio, aussi appelé SSMS. C’est un outil dédié à la gestion des bases de données SQL Server que Technord utilise. Il permet aussi de gérer l’aspect administration de celles-ci, ce qui s’avère utile dans les projets où certains participants ne devraient pas avoir tous les droits.

Seuls quelques outils utilisés par l’entreprise ne proviennent pas de Microsoft. La majorité du temps, cela est dû à un concurrent plus populaire ou offrant plus de fonctionnalités.

Pour le déploiement d’applications sur des machines virtuelles, nous avons utilisé Remote Desktop Manager, développé par Devolutions. Celui-ci permet de gérer les connections à distance et propose une gestion des authentifications sécurisées et efficaces que les alternatives ont du mal à imiter.

La gestion du projet s’est fait avec Jira, créé par Atlassian, au vu de son énorme popularité auprès des entreprises de développement d’applications.

1. Manufacturing Operation Management [↑](#footnote-ref-1)
2. Manufacturing Execution System [↑](#footnote-ref-2)
3. Ingénieur projet senior [↑](#footnote-ref-3)
4. Create Read Update Delete : Créer, lire, mettre à jour et supprimer [↑](#footnote-ref-4)
5. Permettent l’ajout de nouveaux attributs aux balises HTML offrant des fonctionnalités personnalisables. [↑](#footnote-ref-5)
6. Command Line Interface : Outils utilisable depuis la console de commandes. [↑](#footnote-ref-6)
7. Object-Relational Mapper : Outil permettant de faciliter les interactions avec les bases de données en « mappant » les tables à des objets. [↑](#footnote-ref-7)