RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
DANS LE CADRE DU COURS
LOG795 PROJET DE FIN D'ÉTUDES EN GÉNIE LOGICIEL

DOODLE POUR AVOCATS - PROTOGEST

GUILLAUME LARENTE LARG23089406
JULIEN MÉTIVIER METJ11059506
PATRICE ROLLAND ROLP08069104
ALEXANDRE DAIGLE-SAM YENG DAIA17089504

DÉPARTEMENT DE GÉNIE LOGICIEL ET DES TI

Professeur-superviseur

Alain April

MONTRÉAL, 22 AVRIL 2019
HIVER 2019

REMERCIEMENTS

Tout d'abord, l'équipe du projet de fin d'études aimeraient remercier le professeur Alain April qui nous a aidé tout au long du projet sur les décisions à prendre au niveau du développement. Il nous rencontrait à chaque semaine et était disponible pour répondre à nos questions sur Slack. Il a aussi réussi à bien nous expliquer l'historique du projet et la priorité que nous devions mettre sur ce projet cette session, ce qui nous a permis de définir des objectifs pour le projet.

Aussi, nous aimerions remercier Frank Calendriello, le promoteur du projet, pour s'être déplacé à l'ÉTS et nous avoir permis de mieux comprendre le protocole d'instance de la cour supérieure qui était requis pour faire avancer le projet. Son aide, nous a évité une perte de temps énorme à comprendre comme le monde des procès au tribunal fonctionne et qui est très loin de notre champ d'expertise. Merci aussi à Mathieu Dupuis pour ses conseils avec AWS Lambda.

PROTOGEST
DOODLE POUR AVOCATS
RÉSUMÉ

Le but de ce projet est de répondre au besoin précis d'un client, qui est un avocat, Frank Calendriello. Le client veut accélérer le temps requis et diminuer l'effort de coordination nécessaire pour débuter un procès. En effet, présentement les avocats doivent manuellement remplir un document, nommé protocole d'instance, et ce document contient beaucoup de dates que toutes les parties impliquées dans le procès sur lesquelles elles doivent s'entendre.

La solution proposée est d'automatiser le processus afin de faciliter la gestion des dates et la résolution des conflits de dates entre les différents participants. Pour ce faire, le logiciel serait en mesure de reproduire le protocole d'instance dans l'application. Par la suite, lorsqu'un protocole serait créé, on devrait être en mesure de coordonner un calendrier central à partir des calendriers individuels des intervenants. Les personnes impliquées pourront accéder au calendrier et approuver ou modifier les dates proposées. Si une modification à une date est faite, toutes les parties obtiendront une notification et pourront valider à nouveau si toutes les dates leur conviennent. Dès que touts ont approuvé les dates, les invitations seront envoyées à leur calendriers personnels et le protocole d'instance sera produit automatiquement.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

Terme	Définition
REST	Architecture web pour développer des services.
Protocole d'instance	Document à remplir pour entamer une poursuite au tribunal
npm	Logiciel de gestion de dépendances pour JavaScript
AWS	Système de cloud qui permet d'héberger notre application sur leur serveur
API	Application Programming Interface
S3	Système sur AWS pour héberger l'application front-end
IDE	Environnement de travail intégré, qui est un logiciel qui facilite la programmation.
RDS	Relational Database Service. Service de gestion de base de données offert par Amazon
EC2	Elastic Cloud Computing. Plateforme de calcul dans le nuage offert par Amazon
ElasticBeanstalk	Couche d'abstraction sur EC2 facilitant la configuration de l'environnement
AZ	Availability Zone. C'est une zone de disponibilité dans le Cloud d'Amazon représentant un endroit géographique.

Tableau 1 : Définitions préalables utiles

PROJET

Mandat

Notre équipe est la troisième équipe à travailler sur ce projet de fin d'études. Tout d'abord, la première équipe avait respectée les besoins du client, mais le développement front-end n'était pas terminé. En effet, ils utilisaient une librairie payante pour le calendrier ce qui n'est pas adéquat pour développer un logiciel complètement libre. L'équipe qui a poursuivi le projet à effectuée de la ré-ingénierie afin d'améliorer la structure du code. Cependant, cette équipe s'est éloignée des besoins du client. Notre mandat était donc de reprendre le projet et de faire en sorte que les besoins du client soient revus et satisfaits.

Le besoin principal des avocatsviseà faciliter la création d'un protocole d'instance avant d'aller en cour. Un protocole d'instance est un document qui doit être rempli avant d'entamer la poursuite. Le protocole contient les informations concernant l'entité qui entame la poursuite (c.-à-d. le demandeur) et la partie qui défend. Ce protocole contient beaucoup de dates à remplir et toutes les parties impliquées doivent s'entendre ce qui peut être long et demander beaucoup d'effort de coordination. L'objectif du projet est de pouvoir utiliser le calendrier personnel de chaque partie impliquée et accéder aux plages libres afin d'identifier des dates potentielles qui correspondent à tout le monde. Ainsi, beaucoup de temps sera épargné et les clients paieraient moins cher ce qui serait bénéfique pour tous.

Le mandat de l'équipe était donc de répondre à ces besoins en faisant en sorte qu'un avocat qui se connecte sur le logiciel puisse importer son calendrier personnel Outlook, créer un protocole d'instance et gérer des conflits de date entre deux parties impliquées.

Équipe

L'équipe fut décidée aléatoirement par le système de l'ÉTS sur le site Moodle du cours. Malgré tout, l'équipe a très bien travaillé ensemble et nous n'avons rencontré aucun problème majeur lié aux membres de l'équipe et leur capacité à faire avancer le projet.

Membre de l'équipe	Rôles
Alain April	Superviser le projet et s'assurer du bon déroulement de celui-ci
Julien Métivier	Développement front-end de l'application avec emphase sur la gestion du calendrier et AWS
Guillaume Larente	Développement front-end de l'application avec emphase sur la gestion du protocole d'instance
Alexandre daigle sam-yeng	Développement back-end de l'application
Patrice Rolland	Développement back-end de l'application et AWS
Frank Calendriello	Promoteur du projet et connaissance approfondie du domaine d'affaires.

Tableau 2 : Rôle de l'équipe

Comme nous voyons dans le tableau ci-haut, les rôles ont bien été définis et chaque personne avait des tâches bien précises à faire et tous les gens de l'équipe étaient capables d'avancer le projet indépendamment. Chaque personne a contribuée également au projet et est satisfaite de tous les membres de l'équipe sur l'effort mis sur le projet.

Livrables

Dans le cadre de ce projet de fin d'études, les livrables suivants ont été requis. Tout d'abord, un des livrables principal et primordial était le code source du code. En plus du code source pour l'application en entier, nous devions fournir les instructions de lancement et de mise en place de l'environnement de développement. On peut notamment identifier le script de création de la base de données et les étapes de création de l'environnement AWS comme livrables de mise en place de l'environnement. Également, nous devions faire une présentation orale pour montrer l'avancement du projet ainsi qu'un rapport de projet technique. De plus nous devions communiquer à l'aide de Slack.

Cas d'utilisations

Afin d'orienter nos efforts de développement d'une version fonctionnelle de Protogest, nous avons développé des cas d'utilisation simples permettant d'effectuer les actions que nous avons identifiés avec le promoteur du projet .Voici ces Cas d'utilisation:

Cas d'utilisation	Description
CU01 - Authentifier un utilisateur	L'utilisateur s'authentifie sécuritairement avec un courriel et un mot de passe.
CU02 - Créer un protocole d'instance de La Cour Supérieure du Québec	L'utilisateur peut créer un protocole d'instance de La Cour Supérieure du Québec, remplir les des champs et le sauvegarder.
CU03 - Afficher les conflits de dates entre un protocole d'instance et un calendrier personnel	L'utilisateur peut importer son calendrier Outlook et voir les conflits présents entre les évènements de son calendrier et les évènements du protocole d'instance qu'il consulte.
CU04 - Création d'un compte utilisateur	L'utilisateur veut se créer un compte dans Protogest.
CU05 - Afficher les protocoles d'instances associés à un utilisateur	L'utilisateur peut voir les protocoles d'instance qu'il a initiés ainsi que ceux auquel il est associé.
CU06 - Déconnecter un utilisateur	L'utilisateur peut se déconnecter du système.
CU07 - Exporter un protocole d'instance dans un calendrier Outlook	L'utilisateur peut exporter les dates d'un protocole d'instance dans un fichier de format calendrier qu'il peut importer dans son calendrier personnel.
CU08 - Accepter ou refuser une date	L'utilisateur peut consulter un protocole d'instance et accepter et refuser les propositions de date.

Tableau 3 : Cas d'utilisation

Méthodologie de travail

Gestion de projet

Afin de bien mener à terme le projet, une gestion adéquate de celui-ci était primordiale. Nous avons utilisé une approche Agile tout au long du projet. Celle-ci consiste à se rencontrer sur une base hebdomadaire afin de faire état d'avancement des travaux. Nous faisons des Sprints de deux semaines. Dû aux contraintes de temps et d'autres préoccupations comme l'école, les autres cours, nous avons déterminé qu'en une semaine nous n'avions pas le temps de faire beaucoup de développement. C'est pourquoi nous faisons des Sprints sur deux semaines. Ainsi lors de nos rencontres hebdomadaires, nous faisions un suivi sur l'avancement des tâches en cours, ceux qui seraient terminés et les nouvelles tâches à faire.

Outils

Slack

Cet outil de collaboration entre collègues fût très utile pour la communication à distance. Dès qu'on avait une question, on la posait et quelqu'un répondait rapidement. Également, grâce à cet outil, nous pouvions facilement communiquer avec notre professeur pour toute question sur le projet et il pouvait suivre le développement du projet. Le Prof. April était aussi disponible en tout temps sur Slack.

Google Drive

Cet outil de travail collaboratif a été utile pour partager des documents et travailler ensemble sur ces mêmes documents. Les documents principaux utilisés avec le drive sont le PowerPoint pour la présentation ainsi que bien entendu le rapport technique du projet.

Github

Cet outil de gestion de code source fût utile afin de collaborer entre nous et pouvoir intégrer le code de chaque personne en un endroit commun sans trop de difficultés.

Trello

Trello est un outil gratuit de gestion de tâches par tableau. Les tâches à faire étaient écrites, catégorisées et assignées sur un tableau représentant l'avancement du travail. À chaque rencontre, l'équipe consultait le tableau des tâches pour confirmer l'avancement du travail et ajouter les tâches manquantes au tableau.

Google Hangouts

À l'occasion, certains membres de l'équipe ne pouvaient pas se présenter à l'école pour les réunions hebdomadaires. Il nous fallait donc un moyen de communiquer à distance entre nous. Nous avons choisi Google Hangouts, car il permet de partager l'écran avec les autres. Donc, il était possible de présenter le développement fait durant la semaine même si la personne n'était pas présente en personne à l'école.

Technologies utilisées

Angular

L'architecture « front-end » utilisée est celle des plus connue dans le milieu du Web. Angular est basé sur le langage TypeScript et vise à régler tous les problèmes que son ancienne itération, AngularJS, a su laisser à ses utilisateurs. Elle a été sélectionnée pour deux raisons particulières. La première s'explique par le fait que l'équipe précédente l'utilisait et l'a fait évoluer de façon à ce que l'on puisse la réutiliser facilement. Ensuite, notre deuxième critère décisif s'explique par l'expérience que plusieurs membres d'équipes avaient avec cette architecture. D'autres arguments basés sur les performances de l'architecture, la facilité à apprendre pour les membres qui n'ont jamais eu la chance de travailler avec celle-ci et les fonctionnalités variées ont aidé le choix de poursuivre avec le projet fait précédemment.

FullCalendar IO

FullCalendar est un paquet JavaScript permettant facilement l'affichage et la gestion d'évènements d'un calendrier. Tout comme plusieurs paquets JavaScript, il utilise JQuery pour la quasi-totalité de ces interactions avec la page. Il est extrêmement configurable et permet à l'usager un total contrôle sur les données d'évènements, leurs durées et la façon de les afficher à l'usager. Dans le cas de Protogest, il sera utile pour afficher les conflits d'horaire pour une étape particulière, l'étendue totale de l'horaire du protocole en cours et généralement les informations reliées au cas de cour d'un avocat.

Exemple de configuration de FullCalendar:

```
loadCalendar() {
this.calendarOptions = {
editable: true,
eventLimit: false,
      header: {
left: 'prev,nexttoday',
        center: 'title',
        right: 'month,agendaWeek,agendaDay'
      },
events: this.events,
displayEventTime: true,
```

```
allDayText: 'All day',
allDayDefault: true,
timeFormat: 'H:mm',
selectable: true,
eventTextColor: 'white',
eventRender: (v,el) => {console.log(v, el)}
    };
  }
```

Figure 1: Exemple de configuration de FullCalendar

La plupart des configurations sont explicites, par contre les deux lignes qui sont un peu plus intéressantes dans le contexte de notre application sont la définition des évènements et la définition du «eventRender» des évènements ou autrement dit la génération visuelle dans l'interface de calendrier.

Cognito

Dans un souci de réduction du temps de développement et de gestion complexe des usagers, nous avons utilisé le service Cognito d'Amazon Web Service qui permet de créer un bassin d'usagers accessible sur presque toutes les architectures, langages de programmation et autres services d'AWS. Cognito permet d'effectuer les opérations de base sur les usagers comme la création, la connexion, la vérification d'identité, la suppression, la suspension de droits et même le stockage d'informations de base comme le courriel, l'adresse, les préférences utilisateurs et autres. Il devient particulièrement utile dans un logiciel séparé en plusieurs couches (c.-à-d. « front-end » et « back-end » dans notre cas), car il gère les connexions facilement grâce à un jeton d'identification se passant dans une entête de requête HTTP. Dans notre cas, le frontend permet la création et la connexion avec Cognito et une fois l'usager connecté, le jeton est stocké en mémoire et utiliser à chaque appel au « back-end ». Le « back-end » fera une vérification au bassin d'usagers Cognito afin de savoir si la personne est bien connectée et si son jeton est encore valide.

Cognito effectue aussi une identification à double facteur afin de rester dans les normes de sécurité des systèmes en 2019. Dans notre processus, nous avons choisi l'envoi d'un courriel,

mais il est aussi possible de passer par un message texte. L'usager s'enregistre sur l'application. Si ces informations entrent dans les critères de notre bassin d'usager, on lui envoie un code à six chiffres par courriel. Il peut ensuite entrer ce code lors de sa première connexion au site, confirmant ainsi son identité à Cognito.

Ce service nous a vraiment permis de nous départir de toutes gestions de rôles, d'informations personnelles et de connexions sécurisées et nous a fait gagner de précieuses heures de développement. Voici un aperçu en diagramme du lien entre Cognito et notre solution.

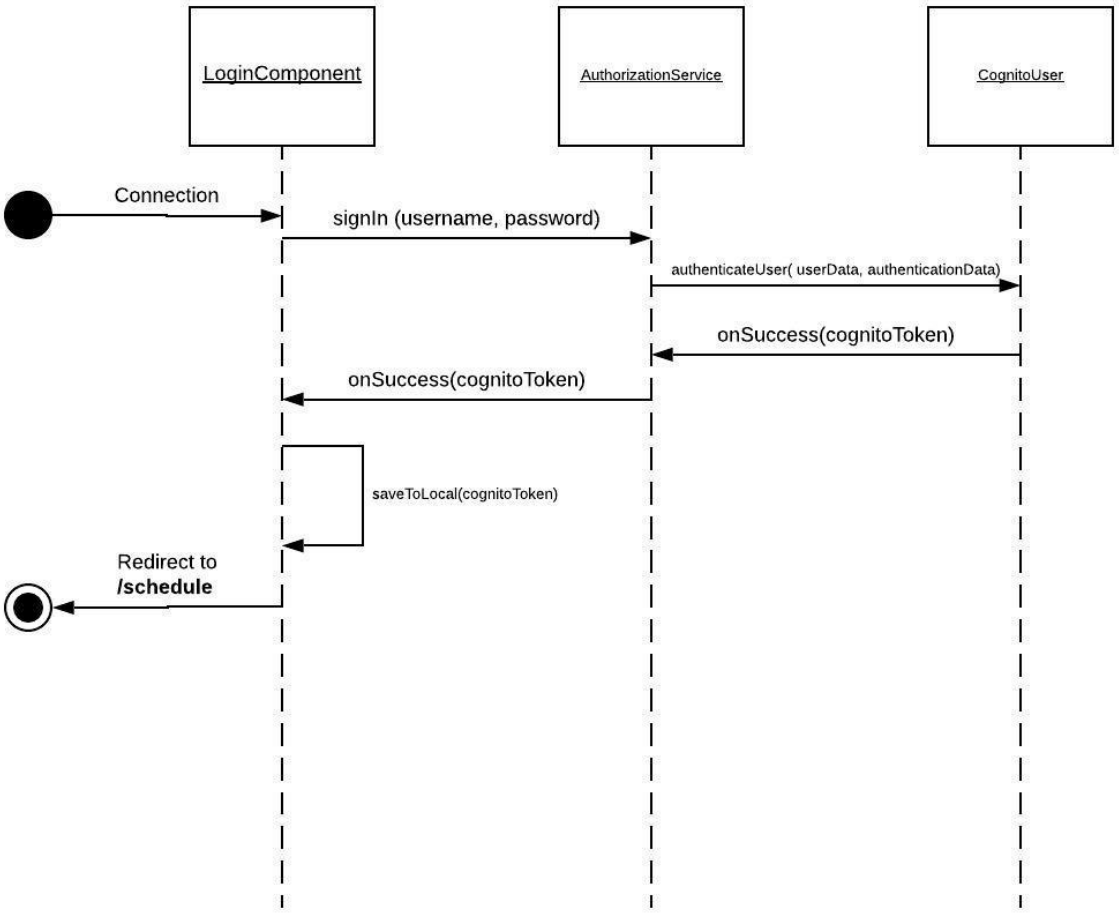

Figure 2: Diagramme de séquence pour l'authentification avec Cognito

S3

Nous avons choisi d'utiliser S3 pour héberger le « front-end ». Cet autre service d'Amazon est un environnement de stockage d'objets de tous genres (c.-à-d. images, vidéos, fichiers JavaScript, etc.) et permet un accès facile et rapide avec Cloudfront. Dans notre cas, nous avons simplifié au maximum son utilisation grâce à la fonction «build» de Angular qui permet de transformer tous les fichiers TypeScript en fichier JavaScript pour ensuite pouvoir les déployer dans un «bucket» S3. Une fois le « bucket » créé, on peut y accéder sur le Web via une URL fournie par Cloudfront. C'est une méthode facile d'utilisation pour tous les développeurs, peu coûteuse et peu complexe à gérer à l'intérieur de l'interface AWS.

Node Package Manager

NPM, qui signifie Node Package Manager, est un outil qui permet d'obtenir et gérer les paquets JavaScript requis pour faire fonctionner l'application. Le logiciel est uniquement disponible en invite de commande et on peut télécharger des paquets qui sont disponible sur internet. Il permet aussi de gérer les paquets et de pouvoir les installer facilement sur un nouveau environnement de travail.

AWS Command Line Interface

Amazon fournit un outil de ligne de commande qui interface avec ses services Web. Dans le cadre du développement du « back-end » de Protogest, nous avons utilisé cet outil pour faire le déploiement de notre application sur ElasticBeanstalk. Nous avons cependant abandonné cet outil puisque durant le développement, nous avons passé d'une instance EC2 à en environnement sur ElasticBeanstalk. Amazon a développé un outil de ligne de commande spécifiquement pour ElasticBeanstalk (EB CLI) que nous avons fini par utiliser pour la création et le déploiement de notre « back-end » sur AWS. Il ne s'agit pas exactement d'un choix technologique, mais plus de l'adoption de l'outil rattaché à une technologie utilisée.

Java

Le langage de programmation du « back-end » de l'équipe précédente était le Java. Dans notre but premier de vouloir continuer le projet, nous avons aussi adopté ce langage. De plus, les deux membres de l'équipe « back-end » étaient très habiles avec Java, ce qui a renforcé

notre choix. À cause des limitations imposées par ElasticBeanstalk, nous étions limités à la version 8 du langage.

Spring Boot
Nous avons utilisé Spring Boot à titre de cadriciel de développement principalement parce que le travail de l'équipe précédente était fait avec ce dernier. Nous nous sommes basés sur le travail effectué pour faire de la rétro-ingénierie sur l'outil et l'utiliser dans notre contexte. Spring Boot utilise beaucoup les annotations dans son exécution et nous avons eu un peu de difficultés à identifier la source des erreurs lorsqu'elles se présentaient.

Spring Boot a vraiment aidé à réduire le code répétitif en fournissant des intégrations avec JPA par exemple. En seulement quelques minutes il était possible de créer un nouveau service lié à de nouvelles tables de base de données en créant simplement une classe d'accès à l'objet de base de donnée, la création d'une interface et d'une méthode dans la couche service. C'est par injection de l'interface qu'il était possible d'accéder à une implémentation générée par Spring donnant accès à des opérations de base sur les objets de la base de données.

Maven
Apache Maven a été utilisé comme outil de gestion et de configuration de projet. Cet outil permet de facilement exposer la structure du projet et de gérer les librairies externes nécessaires au fonctionnement de l'application.

JPA
Nous avons utilisé le Java Persistence API (JPA) pour faire le lien entre nos modèles d'objets et les tables de la base de données. C'est en utilisant les annotations que JPA accède à l'information sur un objet. Spring Boot utilise notamment JPA pour fournir une couche d'abstraction supplémentaire.

Swagger
L'implémentation de Swagger était fortement recommandée par le Prof. April. Il s'agit d'un outil qui permet aux APIs de se documenter automatiquement en fournissant une page web servant à la fois de plateforme de tests fonctionnels que de documentation sur les services implémentés dans l'application. Il était important pour nous de pouvoir bien communiquer

les services implémentés aux développeurs « front-end » mais aussi de fournir de l'information sur le corps de la requête attendu et le de la réponse renvoyée. Swagger fourni une intégration avec Spring Boot se basant sur les annotations et les signatures de méthodes dans le code pour générer la documentation désirée. Il nous a tout simplement fallu ajouter la bonne librairie à notre projet Maven pour avoir accès à ces fonctionnalités.

Github

Nous avons utilisé un compte gratuit sur Github pour créer un dépôt Git privé. C'est avec Github que nous avons fait les revues de code et hébergé la gestion des versions de code.

IntelliJIdea

Les membres de l'équipe étaient tous familiers avec l'environnement de programmation de IntelliJIdea, c'est pourquoi nous avons tous opté pour ce dernier. Plusieurs «plugins» dans cet IDE ont été très utiles pour le formatage du code, l'intégration avec Spring Boot et au maintien d'une bonne qualité de code.

Architecture

Infrastructure

L'infrastructure de la solution utilise au maximum les services de Amazon. De la gestion des utilisateurs sur Cognito jusqu'à la persistance des données sur RDS en passant par l'hébergement du « front-end » sur S3. Ce n'était pas un requis que de mettre nos services sur une plateforme infonuagique, mais c'était une technologie que nous voulions explorer et que certains membres de l'équipe connaissaient de par leur expérience antérieure.

C'est l'aspect de sécurité qui nous a fait choisir Cognito pour gérer notre bassin d'utilisateurs. Concevoir un système sécuritaire de gestion des données personnelles n'était pas un défi que l'équipe voulait relever dans notre itération. Il s'agissait aussi de ne pas réinventer la roue et de maximiser l'intégration avec nos autres services, ce qui faisait de Cognito le candidat idéal pour la gestion de notre bassin d'utilisateurs.

Nous gérons nos instances EC2 par ElasticBeanstalk en parallèle avec le ElasticLoad Balancer pour automatiquement gérer l'expansion horizontale et verticale de notre API.

Une façon très simple d'exposer son application « front-end » est de mettre les fichiers d'exécution dans un S3. De cette façon, le fureteur peut avoir accès aux fichiers, les exécuter, le tout sans avoir besoin de faire l'achat d'un domaine. Pour ensuite diminuer les coûts et améliorer la performance, on peut activer CloudFront sur le S3. C'est en faisant la mise en cache des fichiers et en réduisant les appels nécessaires à S3 que CloudFront diminue les coûts inhérents.

Frontend

L'architecture de base d'Angular est construite en utilisant des blocs appelés «component». Cela permet d'imbriquer des fonctionnalités d'application à travers les pages sans devoir réécrire des interfaces ou de la logique. Pour illustrer plus simplement son utilisation dans notre solution, voici un exemple de l'interface de création de protocole.

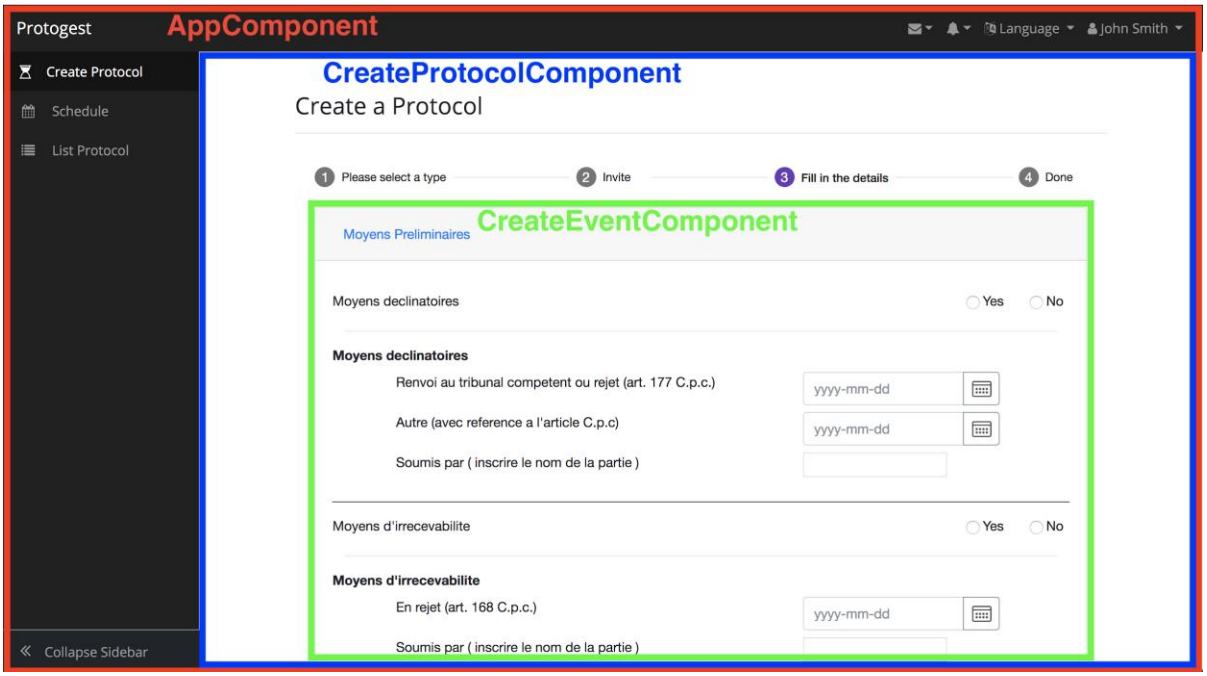

Figure 3: Exemple d'utilisation des *Components* Angular

En rouge, nous avons le «component» principal soit le *AppComponent*. Il sera toujours dans l'application, car c'est lui qui détient les fonctions et structures visuelles que l'on retrouve dans toutes les pages du site. C'est-à-dire c'est la logique de ce bloc qui s'occupe de la traduction, de changer de page selon la liste à gauche et de la gestion de l'usager connecté dans le coin supérieur droit.

Ensuite en bleu, nous avons le *CreateProtocolComponent*qui a comme fonctionnalité la logique de création de protocoles d'instance. Prenons-le en exemple pour visualiser les sept fichiers importants de chaque bloc Angular:

1. **create-protocol-routing.module.ts**: Ce fichier permet de spécifier les routes utiles dans la composante. Contrairement à plusieurs autres architectures web, ces routes ne

sont pas définies dans un seul fichier, mais bien à travers tous les modules pour rester près des interfaces.

2. **create-protocol.component.html** : C'est l'interface elle-même. C'est ce qui sera affiché quand le module sera appelé par la route ou à l'intérieur d'un autre module.
3. **create-protocol.component.scss**: Le SCSS contient le détail visuel lié au HTML. La couleur, la disposition, le format d'affichage, etc.
4. **create-protocol.component.ts**: Ce fichier contient la logique « front-end ». Elle suit de près l'interface, car elle est prioritairement appelée par des actions usagers comme des boutons ou à l'ouverture de la page.
5. **create-protocol.module.ts**: Le fichier module définit les intrants et les sortants du module soit les modules à importer pour l'affichage de la page. Elle définit aussi le constructeur du module pour ensuite être appelée par d'autres pages. C'est sensiblement l'agent de communication/liaison entre les composants.
6. **create-protocol.component.ts**: Le fichier qui définit les tests pour la logique interne du «component».
7. **create-protocol.module.ts**: Le fichier qui définit les tests pour le module. Un exemple de test serait la vérification que tous les modules nécessaires à l'affichage soient présents.

Backend

Lors du développement de l'équipe précédente, une architecture en micro-services avait été adoptée et des APIs minimaux avaient été développés. Cependant, après avoir analysé les fonctionnalités développées, les cas d'utilisation que nous avons choisi de prioriser et le domaine d'affaires, nous avons décidé d'abandonner le code source laissé par l'équipe précédente, car en plus de ne pas utiliser une structure que nous trouvions adaptée au projet, il comportait beaucoup de duplication et n'avait aucun service que nous allions utiliser. De plus, notre compréhension du domaine et notre vision du projet ne correspondaient pas à leur représentation des concepts et des services mis en place, ce qui rendait logique de recommencer avec un code source que nous jugions plus approprié pour le projet.

Afin de simplifier le développement et nous permettre de rapidement produire les services nécessaires pour une version minimale fonctionnelle, nous avons aussi abandonné l'architecture en micro service pour une architecture de modules en couche, où les couches de chaque module, soit le module de calendrier, le module de protocole d'instance et le module d'approbation, sont représentées dans la figure suivante:

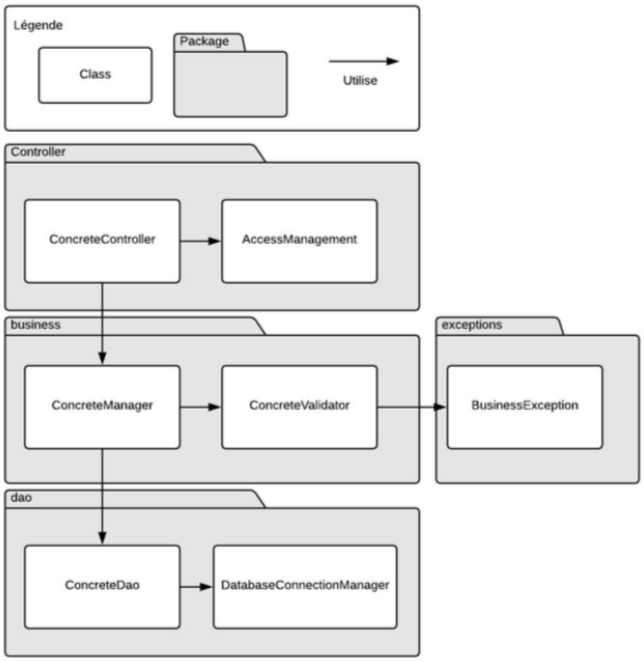

Figure 4: Couches typiques pour un module backend

Dans le code source, chaque module a donc un contrôleur, qui utilise la classe « CognitoUtils » pour gérer les accès. Ce contrôleur utilise ensuite le manager, qui représente les services du module qui eux permettent la collecte et l'enregistrement de données en utilisant la couche de persistance.

Cette architecture permet de rapidement développer des fonctionnalités sans gérer les communications entre les différents micro-services et peut aussi facilement, pour améliorer l'extensibilité, être modifiée en micro-services dans le future.

Conception

Base de données

Pour la persistance des données, suivant le développement de l'équipe précédente, nous avons continué avec le choix de la base de données MySQL. Après une courte analyse de notre part, nous avons déterminé que c'était la base données la plus portable et la plus performante.

Candidat	Indice de performance	Open-source	Portabilité
MySQL	2	Oui	Windows, Linux, Solaris, OSX, FreeBSD OS
MSSQL	3	Non	Windows
PostgreSQL	4	Oui	Windows, Linux, Solaris, OSX, HP-UX

Tableau 4 - Évaluation des bases de données

Nous avons récupéré le script de création de la base de données de l'équipe et créé notre propre base de données dans Amazon avec RDS. Lors du développement, nous nous sommes rendu compte que le désign de l'équipe précédente ne correspondait pas avec l'implémentation que nous voulions faire ce qui nous a amenés à faire notre propre désign de base de données, avec MySQL. C'est lors de notre première ébauche pour sauvegarder le formulaire dans la base de données que nous avons réalisé qu'une base de données relationnelle ne correspondait pas avec le besoin de flexibilité qu'exigeait le formulaire. L'expérience avec une base de données non-relationnelle dans l'équipe « back-end » était très faible, mais il était clair pour nous qu'on ne voulait pas lier la sauvegarde de notre formulaire dans un schéma relationnel. Une preuve de concept a été faite avec DynamoDB et a ensuite été intégrée à notre solution. Idéalement, nous aurions migré toutes les tables

MySQL vers DynamoDB, mais dans le contexte où la majorité des services étaient déjà fonctionnels, nous avons mis de côté cette tâche.

À la fin du projet, notre solution logicielle compte deux bases de données, une relationnelle et une non-relationnelle. Nous avons su identifier le besoin de sortir de notre désign original et du désign de l'équipe précédente pour répondre aux exigences de flexibilité que nous nous étions imposées et implanter une solution fonctionnelle malgré le manque d'expérience de l'équipe avec cette dernière.

Récupération d'évènements calendriers

Dans les requis établis pour le la solution logiciel, l'implémentation d'un service permettant la récupération d'évènements d'un calendrier externe afin de simplifier le choix de dates pour un protocole d'instance. Pour une première itération du logiciel, le calendrier Outlook a été choisi étant donné que c'est le calendrier utilisé par le promoteur du projet, et donc, un des premiers utilisateurs de Protogest.

Comme la sécurité des informations privées et des informations nous avons dû étudier l'api offert par Microsoft Azure pour identifier comment il était possible d'obtenir les évènements calendriers de l'utilisateur en ayant besoin du moins de permissions possible et en récoltant le moins d'information possible. Bref, quand un avocat utilise Protogest, il ne donne des accès qu'à son calendrier, les informations récoltées ne sont que la date de départ des évènements dans la prochaine année, sans avoir l'heure, le nom ou les personnes impliquées, et aucunes des ces informations ou identifiants ne sont stockés de manière permanente par le logiciel. Une solution optionnelle, sécuritaire et privée a donc été mise en place afin de permettre aux utilisateurs qui le désirent de valider qu'ils n'ont pas de conflits de dates avec un protocole d'instance avec les calendriers Outlook.

De plus, étant donné qu'aucune information liée aux évènements externes n'est sauvegardée, ces évènements doivent être envoyés avec chaque appel de service les utilisant. Ceci permet aux autres « endpoints » du « back-end » d'être complètement indépendant du service

externe utilisé pour l'acquisition de ces dates, ce qui rend facilement modifiable les calendriers externes supportés ainsi que les méthodes et fonctions qui dépendent par la suite des valeurs résultantes.

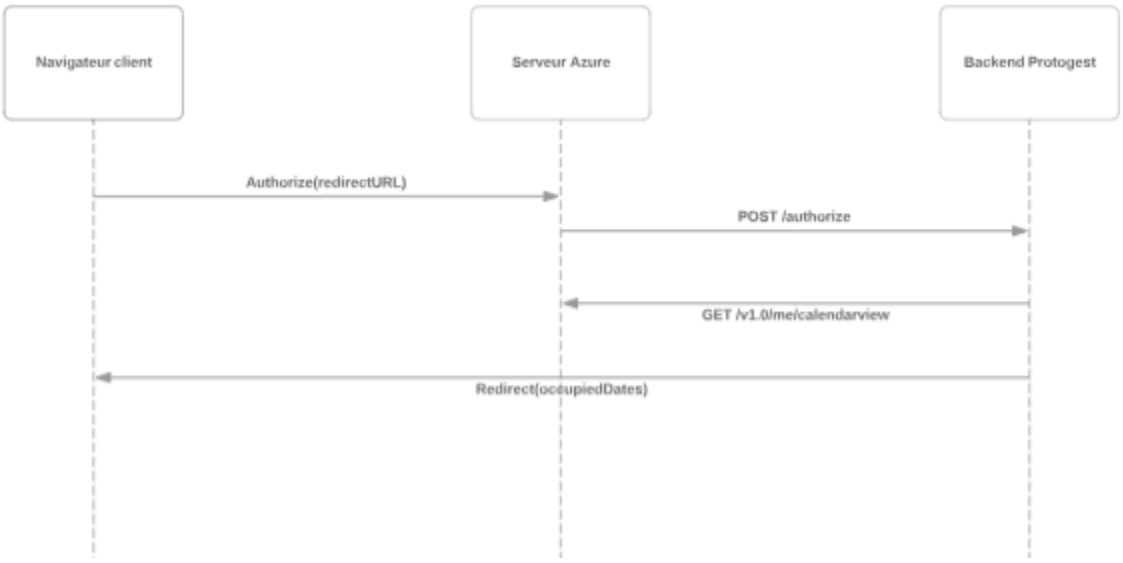

Figure 5: Diagramme de séquence pour la récupération d'évènements avec Outlook

Certificat HTTPS

Étant donné la nature des informations manipulées, sécuriser les communications est une priorité pour que Protogest ait une chance d'adoption dans le milieu légal. Ainsi, nous avons décidé d'équiper notre backend d'un certificat HTTPS pour aider à sécuriser le système. De plus, avoir un certificat valide était nécessaire pour permettre l'utilisation de l'API de Microsoft pour la récupération des évènements Outlook.

Afin de faciliter et simplifier l'acquisition de ce certificat, nous avons fait appel aux ressources de l'autorité de certification « Let's Encrypt ». L'obtention de certificat avec cette autorité est simple et gratuite, ce qui l'a favorisée dans notre choix d'autorité de certification. Nous avons donc produit un certificat en prouvant que le domaine utilisé et le serveur nous appartenaient et avons ensuite appliqué ce certificat au niveau de l'équilibreur de charge

fourni par AWS. Ainsi, notre certificat est valide pour tous les serveurs gérés par l'équilibreur de charge et restera valide, peu importe l'ampleur du système.

Module de calendrier

Avec la reprise du projet, nous avons dû faire quelques modifications au code pour mieux répondre aux exigences. Une des premières modifications et choix faits au frontend était de revoir si le module utilisé pour gérer l'affichage des évènements était toujours pertinent. C'était le package «*DayPilot*» qui avait été choisi pour ses fonctionnalités multiples et son interface linéaire. Par contre, nous l'avons remplacé rapidement par «*CalendarJS*» pour deux raisons.

Le prix du module «*DayPilot*» après 60 jours d'essai gratuit est fixé à 499$US. Cela rend son utilisation complètement inutile dans notre reprise, car le délai était déjà expiré.La deuxième raison était la vue d'ensemble des évènements sur plusieurs mois. Le module *DayPilot* permet de voir les plages de disponibilités des utilisateurs correctement, mais demande à l'usager de faire défiler le calendrier horizontalement trop longtemps.

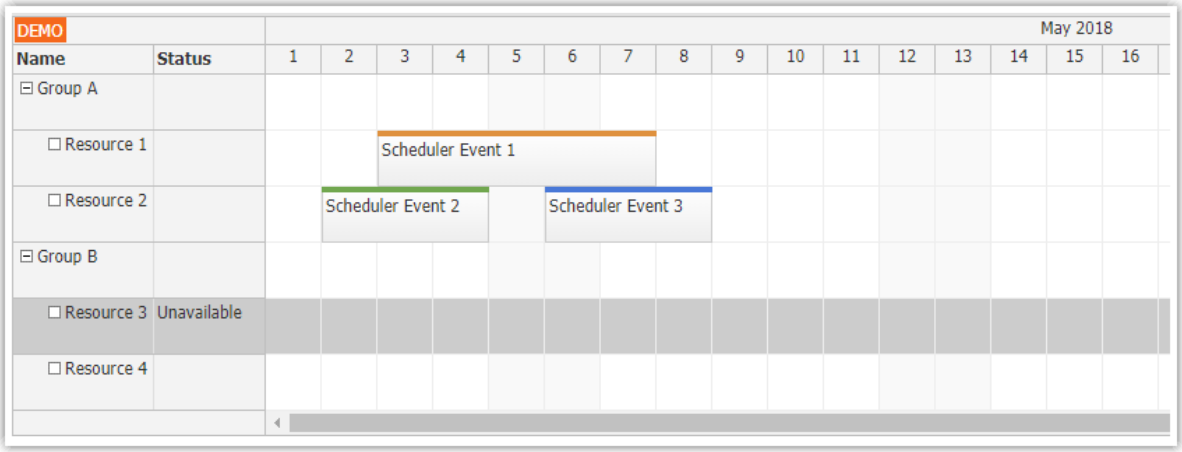

Figure 6: Exemple d'interface du module Daypilot

C'est pour ces raisons que nous avons choisi «*CalendarJS*» qui contient beaucoup moins de fonctionnalités, mais permet un meilleur affichage visuel d'ensemble et il est beaucoup plus personnalisable. De plus, c'est un paquet JavaScript avec un code source libre donc il restera gratuit en tout temps.

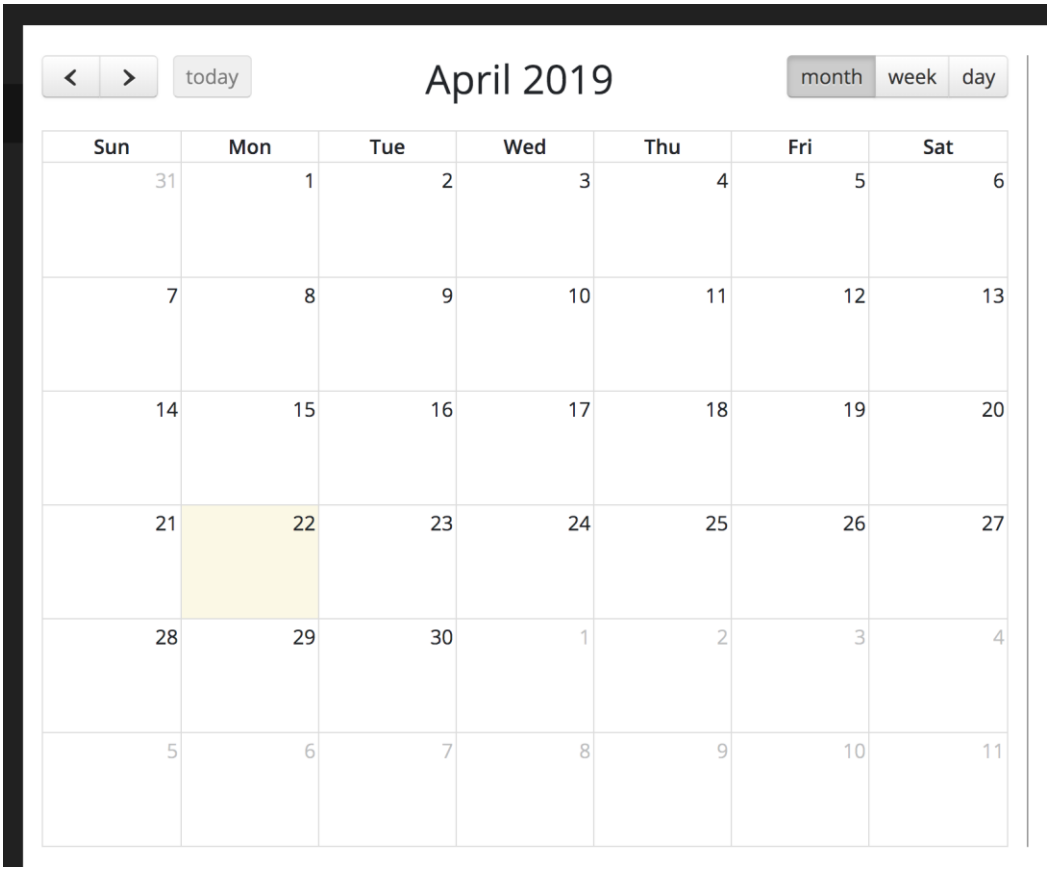

Figure 7: Exemple d'interface du module CalendarJS

Problèmes rencontrés

Tout d'abord, le projet en était à sa troisième version. En effet, deux équipes avaient travaillé sur ce projet dans le cadre de leur projet de fins d'études. Le premier défi était de comprendre ce qu'ils avaient fait, leur code et ce qu'il fallait améliorer. Nous avons tout de même perdu un peu de temps de développement à comprendre comment était conçu le prototype logiciel avant de décider de changer complètement son architecture pour la rendre plus efficace et moderne.

Également, il nous a fallu comprendre le domaine d'affaires qui n'est pas du tout familier pour nous. En effet, personne parmi nous à une expérience préalable avec les processus de poursuite et de planification des dates de tribunaux. Nous avons demandé au promoteur du projet, de nous rencontrer afin de nous expliquer le fonctionnement du protocole d'instance en détail. Ceci nous a permis de mieux comprendre le domaine d'affaires et de pouvoir aller de l'avant avec le développement.

Par la suite, après avoir réglé ces deux problèmes, nous avons pu aller de l'avant et commencer à développer l'application. Il n'y a pas eu de problèmes majeurs lorsque le développement du logiciel a commencé, seulement des décisions sur les choix de conception. Ces décisions n'ont pas retardé l'avancement du projet comparativement aux deux problèmes rencontrés en début de projet présenté ci-haut.

État des développements

Suite aux développements faits pendant la session, certains des cas d'utilisation identifiés ont été complétés complètement et certains ont été complétés partiellement. Voici une mise au point sur l'état de leur avancement:

Cas d'utilisation	État
CU01 - Authentifier un utilisateur	Complété
CU02 - Créer un protocole d'instance de La Cour Supérieure du Québec	Complété
CU03 - Afficher les conflits de dates entre un protocole d'instance et un calendrier personnel	Complété
CU04 - Création d'un compte utilisateur	Complété
CU05 - Afficher les protocoles d'instances associés à un utilisateur	Complété
CU06 - Déconnecter un utilisateur	Complété
CU07 - Exporter un protocole d'instance dans un calendrier Outlook	Partiellement complété: Le système permet la création d'un fichier calendrier, mais l'affichage n'offre pas cette option dans l'interface utilisateur
CU08 - Accepter ou refuser une date	Partiellement complété: La logique d'affaire a été mise en place, mais il n'y a pas d'interface dans lequel les utilisateurs peuvent accepter ou refuser une date. De plus, il est impossible de voir l'état d'une date dans un calendrier de protocole d'instance.

Tableau 5 - État des cas d'utilisation

Améliorations futures

La prochaine équipe qui continue le développement de cette application devrait modifier le système de base de données actuel et n'utiliser qu'une seule base de données, soit DynamoDB. Également, la gestion de l'affichage des conflits n'est pas terminée et était beaucoup simplifiée au « back-end ». Il sera aussi nécessaire de finaliser l'intégration simplifiée des conflits et ensuite la bonifier pour mieux répondre au besoin serait nécessaire dans les versions futures. Principalement, l'ergonomie de l'interface et la facilité d'utilisation sera à améliorer.

De plus, la génération automatique des dates a été implémentée pour le protocole de La Cour Supérieure du Québec. Il devra être expérimenté par des utilisateurs pour voir son efficacité et son ergonomie. Il sera aussi nécessaire d'insérer les autres protocoles d'instance et les dates devraient être implémentées dans la génération d'un protocole d'instance.

Au niveau de la langue, il serait important de rajouter le protocole d'instance en anglais également et de traduire l'application en anglais et en français afin que si le procès a lieu uniquement en français ou en anglais, ils soient possibles de générer le document selon la langue voulue.

Finalement, une des fonctionnalités importantes non implémentées dans cette application serait d'imprimer le protocole d'instance, une fois que le protocole a été approuvé par toutes les parties, en format Word pour complétion par le demandeur et soumettre à la cour pour entamer le procès.

Conclusion

En conclusion, ce projet de fin d'études s'est bien déroulé. Quelques difficultés sont survenues au début, par exemple la compréhension du code source fait par l'équipe précédente ainsi que la compréhension du domaine d'affaires du droit. Dès que nous avons surmonté ces difficultés, nous avons pu avancer le prototype logiciel du projet et avons fait une première version fonctionnelle du prototype logiciel. De plus, l'équipe a bien travaillé ensemble et tout le monde mettait raisonnablement du temps sur l'avancement du projet.

Également, ce projet nous a permis d'approfondir nos connaissances et de développer avec des technologies récentes. Nous avons pu en apprendre plus sur les systèmes infonuagiques et comprendre mieux comment AWS fonctionnes. Aussi, nous avons pu développer le « front-end » de l'application avec un cadriciel qui est très populaire en 2019.

Finalement, nous sommes satisfaits du résultat et avons grandement pu améliorer le prototype d'application par rapport à sa version précédente.

Références

Justice Québec. (2019). [En ligne] Availableat: https://www.justice.gouv.qc.ca/centre-de-documentation/formulaires-et-modeles/systeme-judiciaire/protocoles-de-linstance-en-matiere-civile-et-familiale/

Letsencrypt.org. (2019). *Let'sEncrypt - Certificats SSL/TLS gratuits*. [En Ligne] Availableat: https://letsencrypt.org/fr/

microsoft.com. (2019). *Use the Outlook mail REST API - Microsoft Graph v1.0*. [En Ligne] Availableat: https://docs.microsoft.com/en-us/graph/api/resources/mail-api-overview?view=graph-rest-1.0

Full Calendar (2019) [En ligne] Disponible au: https://fullcalendar.io/

AngularMaterial Stepper (2019) [En ligne] Disponible au: https://material.angular.io/components/stepper/overview

angularjs.org. (2019). AngularJS. [En Ligne] Disponible au: https://docs.angularjs.org/guide

Fullcalendar.io. (2019). *Documentation*. [En Ligne] Disponible au: https://fullcalendar.io/docs

Amazon Web Services, Inc. (2019). *Amazon Cognito - Simple and Secure User Sign Up &Sign In* | *Amazon Web Services (AWS)*. [En Ligne] Disponible au: https://aws.amazon.com/cognito/

Amazon Web Services, Inc. (2019). *Cloud Object Storage* | *Store &Retrieve Data Anywhere* | *Amazon Simple Storage Service*. [En Ligne] Disponible au::https://aws.amazon.com/s3/

Spring.io. (2019). *SpringProjects*. [En ligne] Disponible au: https://spring.io/projects/spring-boot

Porter, B., Zyl, J. and Lamy, O. (2019). *Maven – Welcome to Apache Maven*. [En ligne] Maven.apache.org. Disponible au: https://maven.apache.org/

Swagger.io. (2019). *Swagger Tools* | *Swagger*. [En ligne] Disponible au: https://swagger.io/

Amazon. *Documentation AWS - Guides et références API*. [En ligne] Disponible au https://docs.aws.amazon.com

Annexe 1 - Cas d'utilisations

CU01 - Authentifier un utilisateur	
Description	Un utilisateur souhaite s'authentifier dans Protogest
Préconditions	L'utilisateur doit être sur la page d'accueil du site
Flux de base	1. L'utilisateur entre son courriel 2. L'utilisateur entre son mot de passe 3. L'utilisateur clique sur le bouton de connexion. 4. Le système valide l'information et redirige le fureteur web vers la page du calendrier d'évènements de l'utilisateur.
Post-conditions	L'utilisateur est authentifié dans le système

CU02 - Créer un protocole d'instance de La Cour Supérieure du Québec	
Description	L'utilisateur peut créer un protocole d'instance de La Cour Supérieure du Québec, remplir les des champs et le sauvegarder.
Préconditions	L'utilisateur doit être authentifié dans le système (CU01)
Flux de base	1. L'utilisateur clique sur l'onglet pour créer un protocole d'instance 2. L'utilisateur sélectionne le type de protocole pour la Cour Supérieure du Québec et clique sur le bouton «Suivant» 3. L'utilisateur ajoute le courriel d'un participant et clique sur le bouton «Suivant» 4. L'utilisateur choisit et remplit les sections qu'il désire et clique sur le bouton «Sauvegarder». 5. Le système valide les champs, sauvegarde l'information et envoie un courriel au participant entré par l'utilisateur
Post-conditions	Un protocole d'instance de la Cour Supérieure du Québec a été créé et liée à l'utilisateur et une notification a été envoyée au participant.

CU03 - Afficher les conflits de dates entre un protocole d'instance et un calendrier personnel	
Description	L'utilisateur peut importer son calendrier Outlook et voir les conflits présents entre les évènements de son calendrier et les évènements du protocole d'instance qu'il consulte.
Préconditions	L'utilisateur doit être authentifié dans le système (CU01) et être en train de consulter un protocole d'instance.
Flux de base	1. L'utilisateur clique sur «Synchroniser avec Outlook» 2. Le système redirige le fureteur web de l'utilisateur vers la page de connexion sur Outlook 3. L'utilisateur inscrit ses informations de connexion avec Outlook et clique sur «Connection» 4. L'utilisateur est redirigé dans Protogest sur le protocole qu'il consultait et le système a importé les dates du calendrier Outlook de l'utilisateur. 5. Le système présente à l'utilisateur les conflits d'horaires entre le calendrier Outlook et le calendrier des évènements du protocole d'instance
Post-conditions	L'utilisateur peut consulter les conflits d'horaires entre son calendrier Outlook et le calendrier du protocole d'instance qu'il consulte

CU04 - Création d'un compte utilisateur	
Description	L'utilisateur veut se créer un compte dans Protogest.
Préconditions	L'utilisateur doit être sur la page d'accueil du site
Flux de base	1. L'utilisateur clique sur le lien «Créer un compte» 2. Le système redirige l'utilisateur sur la page de création de compte 3. L'utilisateur entre son nom, son courriel et son mot de passe et clique sur le bouton «Créer un compte» 4. Le système valide que le courriel semble valide et que le mot de passe correspond aux critères de sécurité 5. Le système envoie un courriel de validation au courriel entré par l'utilisateur 6. L'utilisateur ouvre le courriel et clique sur le lien validant ainsi la création de son compte

Post-conditions	L'utilisateur peut maintenant se connecter dans l'application Protogest

CU05 - Afficher les protocoles d'instances associés à un utilisateur	
Description	L'utilisateur peut voir les protocoles d'instance qu'il a initié ainsi que ceux auquel il est associé.
Préconditions	L'utilisateur doit être connecté (CU01)
Flux de base	1. L'utilisateur clique sur «Mes protocoles» 2. Le système affiche la liste des protocoles créés par l'utilisateur ainsi que ceux sur lesquels il participe
Post-conditions	-

CU06 - Déconnecter un utilisateur	
Description	L'utilisateur peut se déconnecter du système.
Préconditions	L'utilisateur doit être connecté (CU01)
Flux de base	1. L'utilisateur clique sur le bouton «Déconnection»
Post-conditions	L'utilisateur n'a plus accès aux fonctionnalités de Protogest

CU07 - Exporter un protocole d'instance dans un calendrier Outlook	
Description	L'utilisateur peut exporter les dates d'un protocole d'instance dans un fichier de format calendrier qu'il peut importer dans son calendrier personnel.
Préconditions	L'utilisateur doit consulter un protocole
Flux de base	1. L'utilisateur clique sur le bouton «Exporter» 2. Le système génère un fichier du format «ics» 3. Le fureteur web télécharge le fichier ics
Post-conditions	L'utilisateur peut importer le fichier ics dans son calendrier Outlook

CU08 - Accepter ou refuser une date	
Description	L’utilisateur peut consulter un protocole d’instance et accepter et refuser les propositions de date.
Préconditions	L’utilisateur doit consulter un protocole
Flux de base	1. L’utilisateur clique un évènement à valider a. L’utilisateur accepte l’évènement b. L’utilisateur refuse l’évènement et propose une nouvelle date 2. Le système sauvegarde l’information et envoie une notification aux participants
Post-conditions	Le protocole d’instance est modifié et les participants ont reçu une notification indiquant que le protocole a changé

Annexe 2 - Architecture logicielle sur AWS