

# Rapport ENI

Concepteur Développeur Informatique

**COMPERE Dylan**



## NOTE DE CONFIDENTIALITE

« Ce document comportant des informations internes à l'entreprise Design' Partner et à caractère confidentiel, sa consultation par d'autres personnes que le jury est soumise à l'autorisation de l'entreprise. »

# Rapport d'activité professionnelle

## SOMMAIRE

<b>Remerciement</b>	<b>3</b>
<b>Introduction</b>	<b>3</b>
<b>Présentation de l'entreprise</b>	<b>4</b>
Les services	4
Listes des principaux sites e-commerces de Design' Partner	4
Le poste	5
<b>Résumé du projet en anglais</b>	<b>6</b>
<b>Liste des compétences du référentiel qui sont couvertes par le projet</b>	<b>7</b>
<b>Expressions des besoins de l'application</b>	<b>8</b>
Contexte	8
Besoins	8
Exigence de l'entreprise pour ce projet	8
<b>Spécifications fonctionnelles</b>	<b>9</b>
Acteurs de l'application	9
Actions liés aux acteurs	10
Détails des fonctionnalités principales attendues au lancement du projet	13
<b>Spécifications techniques</b>	<b>23</b>
Environnement de travail	23
Architecture et détails techniques de l'application	24
Base de données	29
Front end et Polymer	31
<b>Réalisation</b>	<b>36</b>
<b>Gestion de projet</b>	<b>39</b>
<b>Conclusion</b>	<b>41</b>
<b>Glossaire</b>	<b>42</b>

## Remerciements :

Avant de commencer je voudrais remercier mon tuteur Brice Gaudin et mon employeur Maxence Corbrion pour m'avoir fait confiance et aidé tout au long de cette alternance. Je remercie aussi toute l'équipe de l'entreprise Design' Partner avec laquelle je travaille tous les jours.

## Introduction :

Je me présente, Dylan COMPÈRE 22 ans. Je suis actuellement développeur en contrat de professionnalisation dans l'entreprise Design' Partner à Vire. Je vais vous présenter mon parcours, de mon bac jusqu'à mon arrivée en formation au sein de L'ENI.

J'ai réalisé un Bac professionnel TEBEE (Technicien d'étude du bâtiment option A: Étude et économie de la construction), au Lycée Privé ingénieur Cachin à Cherbourg. Cette formation est axée sur l'acquisition de connaissances techniques et économiques dans tous les domaines de la construction (Métré, dessin assisté par ordinateur, ...). Elle m'a permis de connaître rapidement le monde de l'entreprise, étant donné que j'ai réalisé 22 semaines de stage sur 3 ans.

Pendant cette formation j'ai découvert ma passion pour l'informatique en général, mais avec une préférence pour le développement. C'est à la suite de ça que j'ai décidé de continuer mes études dans ce domaine.

En septembre 2013 je suis rentré au Lycée St Ursule à Caen, pour suivre le BTS SIO (Services informatiques aux organisations: option B solutions logicielles et applications métiers). L'objectif de ce BTS est de former au développement, et d'apprendre à adapter et à maintenir des solutions applicatives. Pendant cette formation j'ai pu apprendre les bases du développement dans différents langages (PHP, SQL, Java, HTML, CSS). Ce BTS m'a permis de réaliser 10 semaines de stage et d'obtenir un emploi saisonnier en juillet 2014 dans l'entreprise Mecatek, une agence de création graphique et de publicité tout support dont les sites web.

Après avoir obtenu mon diplôme en juillet 2015, j'avais pour souhait de trouver une poursuite d'étude en alternance pour ensuite avoir la possibilité de décrocher un CDI. J'ai donc décidé de rentrer à l'ENI, qui proposait une formation adaptée à mes attentes. Après avoir réussi les tests d'admission et trouver un employeur j'ai commencé la formation en septembre 2015.

# Rapport d'activité professionnelle

La suite de ce rapport sera divisée en 3 grandes parties. La première je présenterai l'entreprise et l'ensemble de son équipe et service. En deuxième partie je parlerai du projet en relation avec ce rapport et la dernière partie sera la conclusion sur le projet, le bilan sur les 2 ans d'alternance et mes projets professionnels pour le futur.

## Présentation de l'entreprise :

Design' Partner est une société créée depuis 1999 et située à Vire en Normandie. Elle embauche une vingtaine de salariés dont 4 développeurs. Le domaine d'activité principale de l'entreprise est le e-commerce, avec des sites web de vente de textiles vierges et imprimés. L'entreprise réalise aussi l'impression de visuel sur des mugs.

### **On retrouve différents services au sein de l'entreprise:**

- La production qui va s'occuper de la préparation et de l'impression des visuels du client, pour ensuite réaliser le marquage sur les différents textiles.
- Le service client qui s'occupe de la relation client. Il permet de dépanner, renseigner, et conseiller le client par téléphone et /ou par mail.
- Le service préparation qui va gérer la préparation des produits des commandes clients.
- Le service de maintenance qui va gérer la maintenance des différents systèmes et outil de travail de l'entreprise (presse à mugs, presse à tee-shirts, ...)
- Le service d'expédition qui va expédier les commandes clients.
- L'équipe de développement qui gère l'ensemble des sites de l'entreprise, mais aussi l'ensemble des interfaces utilisées par les différents services de l'entreprise. (Interface de préparation de commande, marquage des tee-shirts, réapprovisionnement de produit, ...).

### **Une listes des principaux site e-commerces de Design' Partner :**

- La maison de tee-shirt (<https://www.lamaisonduteeshirt.com>)
- Tee-shirt express (<https://www.tee-shirts-express.com>)
- Tee-shirt online (<https://www.tee-shirts-online.com>)
- Mug Express (<https://www.mug-express.com>)

## **Le poste :**

J'occupe actuellement un poste de développeur au sein de l'entreprise. Nous travaillons sur des technologies et langages web tels que du PHP, SQL, HTML, CSS, JavaScript avec des frameworks tels que jQuery, Polymer et Silex. Nous utilisons comme IDE PHPStorm avec Git comme gestionnaire de version. Le système de gestion de base de données (SGBD) sur lequel nous travaillons est mariaDB.

## Résumé du projet en anglais :

Design' Partner wants to transfer knowledge, share information, resources and facilitate communication between employees within the company. The goal is to create enterprise wiki. What an Enterprise Wiki ?

Enterprise wiki is an application meant to be used in a corporate (or organizational) context, especially to enhance internal knowledge sharing.

### ***Features of wikis specifically helpful to a corporation include:***

- Allows entering information, via quick-and-easy-to-create pages, containing links to other corporate information systems like people directories, CMS, applications, and thus to facilitate the buildup of knowledge bases.
- Avoids e-mail overload. Wikis allow all relevant information to be shared by people working on a given project. It is also very useful for the project manager to have all the communication stored in one place, which allows them to link the responsibility for every action taken to a particular team member.
- Organizes information. Wikis allow users to structure new and existing information. Like editable content, the structure of data is sometimes also editable by users.
- Access levels by rights and roles. Users can be denied access to view and/or edit given pages, depending upon their department or role within the organization.
- Knowledge management with comprehensive searches. This includes document and project management, as well as using a wiki as a knowledge repository useful during times of employee turnover, retirement and so on.

In this project all employees must be able to access and edit the resources (training, decision support, maintenance reports) depending on its situation in the company. The application must include rights management (read, write, change) according to users, and the application interface must be simple to use and responsive.

## Liste des compétences couvertes par le projet :

Le projet m'a permis de couvrir la majorité des compétences du référentiel. Je liste ci-dessous ces compétences avec une justification concernant le projet pour chacune. La justification détaillée de chacune des compétences ce fera tout au long de ce rapport.

### Développer des composants d'interface:

- Maquetter une application : Création diagramme de cas d'utilisation et mockup de l'application.
- Développer une interface utilisateur : Micro framework Silex, DAO Générique et documentation du code.
- Développer des composants d'accès aux données : Création model, dao et classe DAO Générique.
- Développer des pages web en lien avec une base de données : API Rest pour le back end faisant appel à la DAO et front end Polymer interagissant avec l'API.

### Développer la persistance des données:

- Concevoir une base de données : Création d'un Modèle conceptuel de données.
- Mettre en place une base de données : Création de la base de données avec PhpMyAdmin.
- Développer des composants dans le langage d'une base de données : Création d'une DAO générique.
- Utiliser l'anglais dans son activité professionnelle en informatique : Documentation et logiciel en anglais et développement en anglais.

### Développer une application n-tiers:

- Concevoir une application : Schéma de la structure de l'application, diagramme de cas d'utilisation.
- Collaborer à la gestion d'un projet informatique : Gestion projet agile avec utilisation de Scrum, Git + Gitlab pour gestion de code source.
- Développer des composants métier : Création du modèle objet de la base de données.
- Construire une application organisée en couches : Création de l'API avec un couche controller, model, service et DAO.
- Développer une application de mobilité numérique : Création d'une application à partir de Polymer, suivant les principes du Material Design, responsive avec l'utilisation de flexbox et micro framework CSS « Grillade ».
- Préparer et exécuter le déploiement d'une application : Utilisation des différents environnements Dev, Preprod, Prod.

## Expressions des besoins de l'application :

### **Contexte**

Avant de parler en détail des besoins et avantages attendus par l'entreprise à la suite de ce projet, je vais rappeler le contexte actuel. L'entreprise possède des ressources telles que des rapports de maintenances et des documents de formation en version papier et à différents endroits dans l'entreprise. Le partage de ces ressources entre le salarié n'est pas facile car elles ne sont pas centralisées, s'ajoute à ça la masse importante de document papier et de mails. De plus la formation de salariés à certains postes dans l'entreprise demande du temps et ne peut se faire seul.

### **Besoins**

Pour ces raisons l'entreprise souhaite créer un espace centralisé de partage de connaissances et de ressources dans le but d'avoir:

- Une base de connaissances et transfert de connaissances
- Une veille Collaborative sur les technologies ou la concurrence
- Une gestion de contenu et partage d'informations
- Une collaboration à distance (brainstorming, boîte à idées)
- Une communication interne (comptes rendus de réunions)
- Permettre l'autoformation du personnel à certains postes de l'entreprise

Les avantages attendus par l'entreprise à la suite de ce projet et de réduire la masse de courriels et de fichiers échangés, limiter la duplication involontaire de documents et d'informations, conserver un historique des connaissances et améliorer la collaboration sur les informations.

### **Exigence de l'entreprise pour ce projet**

L'espace de partages doit être centralisé et accessible depuis n'importe quel support. Chaque utilisateur aura des identifiants pour se connecter à cette espace. De plus l'interface devra être simple d'utilisation pour qu'elle soit utilisable par l'ensemble des personnes de l'entreprise et responsive afin qu'elle s'adapte à tous type de support (tablette, smartphone, ordinateur). Une partie administration devra être intégrée au projet pour gérer les documents et ressources proposées par chaque utilisateur de l'application. L'administrateur devra modérer / valider chaque documents ajoutés ou modifiés par un utilisateur. Chaque personne utilisant l'application devra avoir des droits sur celle-ci défini par l'administrateur. Droit en lecture, création et modification. Des droits pourront être ajoutés ou modifiés par la suite. Cette fonctionnalité devra être présente dans la partie administration du projet.



## Spécifications fonctionnelles :

Avant de continuer je spécifie quelque point important pour la suite du rapport. Etant donné que le projet est développé pour un usage interne à l'entreprise et utilisant une gestion de projet agile, il n'y a pas de cahier des charges ou de spécifications fonctionnelles complètes et précises concernant le projet. Celui-ci est développé sous forme d'itération, il est donc en perpétuelle évolution. Cependant il y a quand même un besoin de base défini au départ du projet, qui servira de support pour la suite du rapport. Autres informations à préciser on m'a laissé libre choix sur la charte graphique de l'application.

Je vais aborder les spécifications fonctionnelles en 3 grandes parties. Je vais d'abord parler des acteurs de l'application et de leurs rôles, ensuite je détaillerai les actions que chacun des utilisateurs pourra effectuer et je finirai par détailler les fonctionnalités principales attendues au lancement du projet. Cette partie permettra d'avoir une meilleure visibilité du projet.

### **Les acteurs**

Dans un premier temps je vais parler des différents acteurs de l'application, qui sont:

- Le service client
- La production
- Le service préparation
- Le service de maintenance
- L'équipe de développement
- Les administrateurs

Chaque acteur correspond aux différents services de l'entreprise. Je vais rappeler succinctement les actions des différents services.

Le service client: ce service s'occupe de la relation client. Il permet de dépanner, renseigner, et conseiller le client par téléphone et /ou par mail.

La production: c'est l'équipe qui va s'occuper de la préparation et de l'impression des visuels des clients, pour ensuite réaliser le marquage sur les différents textiles.

# Rapport d'activité professionnelle

Le service préparation: c'est le service qui va gérer la préparation des produits d'une commande, et qui va ensuite expédier cette commande.

Le service de maintenance: le service qui va gérer la maintenance des différents systèmes et outil de travail de l'entreprise (presse à mugs, presse à tee-shirts, ...)

L'équipe de développement: elle va réaliser la maintenance, la création, et la mise en place de nouvelles fonctionnalités sur les différents sites web de l'entreprise, mais aussi des interfaces en interne.

Chacun des acteurs aura des droits sur l'application. Ces droits pourront être édités et/ou modifier en cas de besoin par le ou les administrateurs. Ils devront permettre de restreindre l'accès à certaines fonctionnalités de l'application.

L'application devra avoir un accès sécurisé par un login et un mot de passe pour chaque utilisateur. Les identifiants de connexion seront géré par l'administrateur. Chaque utilisateur pourra se déconnecter à n'importe quel moment de l'application.

## **Les actions**

Je vais maintenant détailler, par acteurs, chacune des actions qu'il pourra réaliser sur l'application.

Chacun des acteurs aura accès à 3 fonctionnalités communes principales. Celle de pouvoir créer, modifier et consulter un document qui appartiendra à une catégorie. Cette dernière sera définie en fonction du service qui va créer le document.

**Exemple**: si le service client crée un document, alors il aura pour catégorie "Service client". Dans un autre cas si un document est créé dans le service maintenance alors il appartiendra à la catégorie "maintenance".

La consultation d'un document devra simplement permettre d'afficher le contenu du document avec les informations de celui-ci (date de modification, auteur, titre)

# Rapport d'activité professionnelle

La création et la modification d'un document fonctionnera de la même manière. Chaque utilisateur devra renseigner un titre et ajouter du contenu dans le document. L'édition de ce dernier devra posséder plusieurs outils:

- Insertion d'un titre interne au document
- Insertion d'un sous-titre
- Insertion d'un paragraphe
- Insertion d'une liste à puces numérotées ou non
- Insertion d'un extrait de code source avec coloration syntaxique
- Insertion d'image
- Insertion de vidéo
- Insertion d'une ligne horizontale
- Insertion d'un bloc d'information
- Insertion d'un bloc d'avertissement / d'alerte
- Changement de style du texte (gras, souligné, italique et surligner)
- Suppression de la mise en forme du texte
- Changement de l'alignement du texte (justifié, centré, gauche, droite)
- Annulation des dernières modifications des documents

Chaque utilisateur pourra enregistrer les modifications de son document à tout instant. Je vais maintenant aborder les actions spécifiques à chaque service.

## Service Client:

Le service client devra pouvoir accéder à une interface d'aide à la décision. Celle-ci va permettre de lister les questions les plus souvent posées par les clients, avec les réponses associées à celle-ci. Cet outil devra permettre aux opérateurs du service, de les aider dans les réponses à donner aux clients. Chacune des questions et des réponses pourront être éditée à tout moment par le service clients. Il pourra ajouter, modifier ou supprimer une question / réponses. L'édition d'une nouvelle question devra être validée au préalable par l'administrateur.

## L'équipe de développement:

L'équipe de développement devra pouvoir gérer la remontée de bugs et la proposition d'améliorations. Pour les améliorations ou les bugs, les développeurs devront pouvoir ajouter plusieurs infos:

- La date
- La description
- L'auteur
- La priorité
- Le Statuts (Réalisé, à faire, en cours)

# Rapport d'activité professionnelle

Chaque bug et / ou amélioration pourra être ajouté, édité, et supprimé à n'importe quel moment.

L'équipe de développement aura aussi un espace de gestion Scrum. Les développeurs pourront créer des sprints, des storys, des procédures et des comptes rendus de réunions. Ils pourront aussi afficher les statistiques de chacune des storys avec le score réalisé par chaque développeur dans un sprint.

## Les administrateurs:

Les administrateurs auront des droits et pourront gérer l'ensemble de l'application. Une des premières fonctionnalités de l'administrateur sera de pouvoir modérer tous les documents créés depuis l'application par les différents utilisateurs, mais aussi de modérer les questions et réponses du service clients. Pour la modération / validation des documents, l'administrateur pourra afficher le contenu du document et ensuite décider de le modérer ou non.

Un administrateur pourra aussi gérer les différentes catégories de documents liés à chaque acteur de l'application. Dans un premier temps il pourra seulement modifier le libellé de la catégorie. Des améliorations seront apportées à cette fonctionnalité par la suite, mais elles ne font pas partie des spécifications fonctionnelles de base du projet. La modification se fera via un formulaire après avoir sélectionné la catégorie à modifier dans une liste.

L'administrateur pourra aussi gérer les templates de document. Il pourra gérer un style par défaut de document en fonction des catégories de ce dernier.

La dernière fonctionnalité de l'administrateur sera de pouvoir gérer les droits et les rôles de chaque utilisateur de l'application. Chaque utilisateur appartiendra à un rôle et pour chaque rôle nous aurons des droits sur l'application. L'administrateur pourra modifier et créer des rôles et des droits.

Il sera nécessaire d'afficher tous les rôles, ensuite les droits associés à chacun et les utilisateurs appartenant aux rôles sélectionnés. L'administrateur devra aussi avoir des informations sur l'activité de chaque utilisateur au travers de l'application (document créés, modifiés, ...)

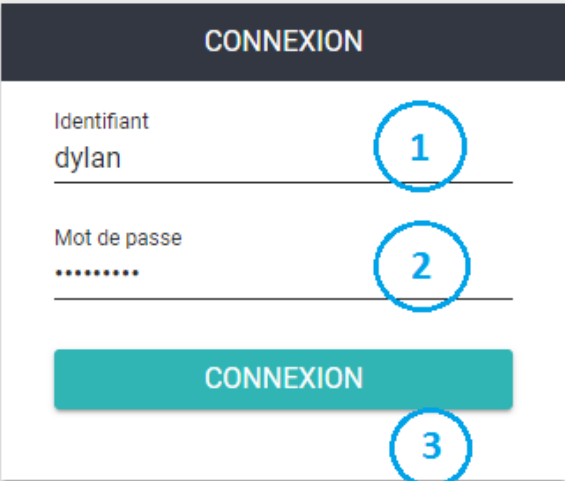
Pour la majorité des actions de l'utilisateur, l'application devra afficher une notification afin de savoir si l'action s'est bien déroulée ou si une erreur est survenue. Pour les actions les plus longues, un loader devra être affiché pour informer l'utilisateur d'une attente.

# Rapport d'activité professionnelle

## Détail des fonctionnalités principales attendues au lancement du projet

Pour les fonctionnalités attendues par écrans. Pour chacune d'entre elles je vais détailler leur comportement principal mais aussi les comportements en cas d'erreur sur l'application.

### Interface de login



The image shows a login form titled "CONNEXION" in a dark header. Below the header, there are two input fields: "Identifiant" with the value "dylan" and "Mot de passe" with masked characters ".....". Each input field is circled with a blue number: (1) for the identifier field and (2) for the password field. Below these fields is a teal button labeled "CONNEXION", which is circled with a blue number (3).

Je vais commencer par le premier écran, celui de la connexion de l'utilisateur ou de l'administrateur. Cet écran possède un formulaire étant composé de 2 champs. Un champ pour l'identifiant (1) et un champ pour le mot de passe (2). Nous retrouvons bien évidemment un bouton qui va permettre la validation de ce formulaire (3). Je vais maintenant détailler le comportement de cette interface.

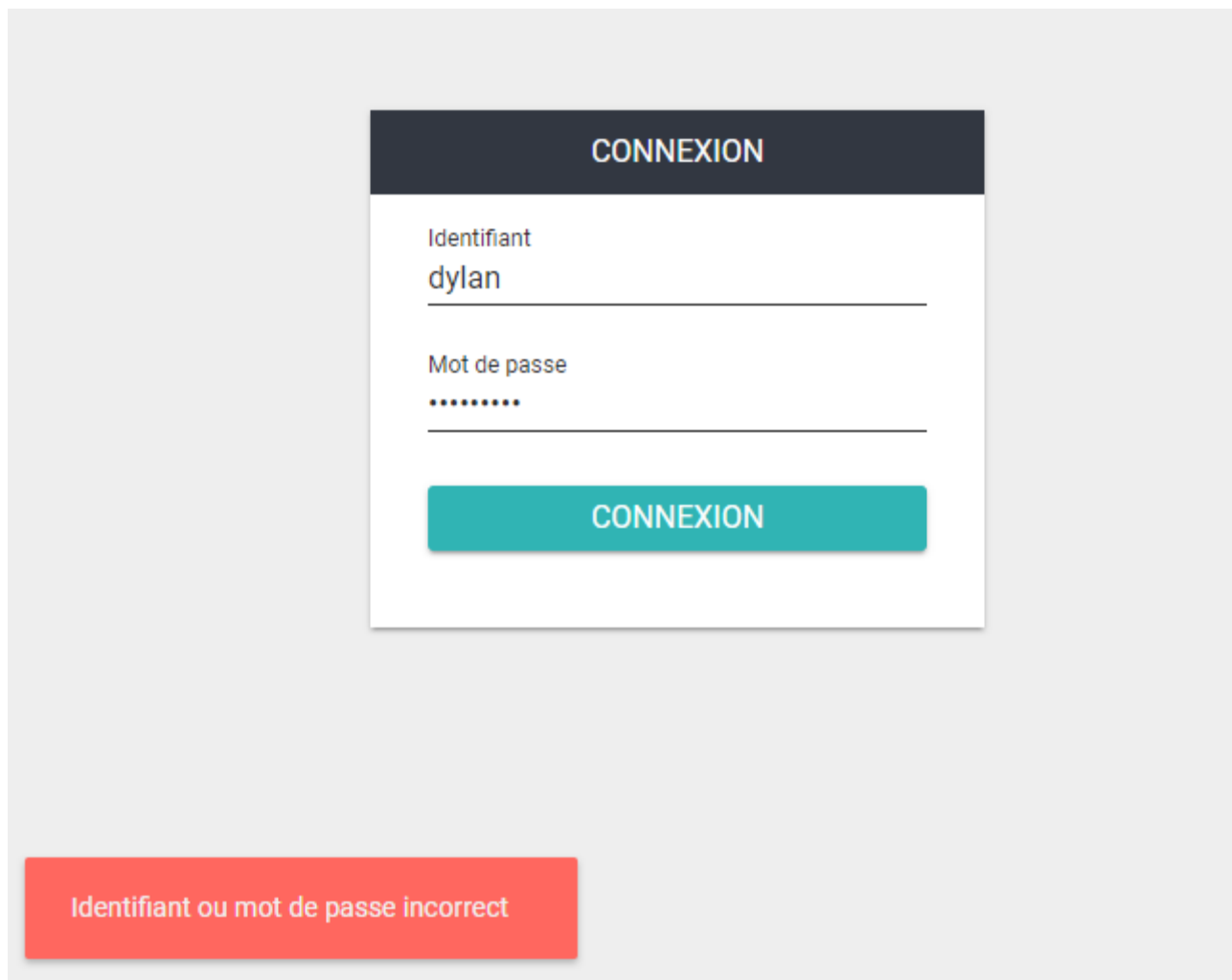
### Comportement principal:

Après avoir renseigné ces identifiants et valider le formulaire, nous vérifions si les informations de ce dernier sont valides. Dans ce cas l'utilisateur sera redirigé vers la page d'accueil de l'application.

# Rapport d'activité professionnelle

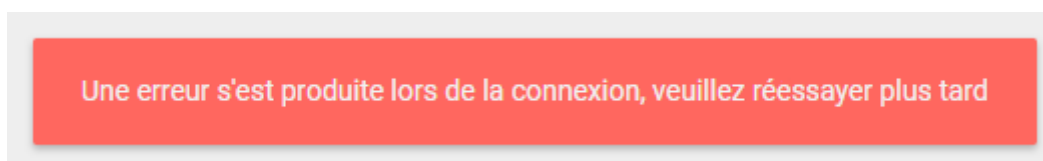
## Comportement en cas d'erreur des identifiants:

Dans le cas où l'utilisateur fait une erreur en renseignant son login et son mot de passe, alors nous affichons sur l'écran un "Toast", qu'on pourrait aussi appeler "Bulle de notification", lui informant que les identifiants sont incorrects. Voir capture d'un exemple ci-dessous.



## Comportement en cas d'erreur liée au serveur gérant la connexion:

Une erreur peut survenir sur le serveur. Dans le cas où celui-ci ne répond plus, alors nous affichons sur l'écran une "bulle de notification", lui informant que le serveur ne répond pas et qu'il doit réessayer plus tard. Voir capture d'un exemple ci-dessous.



# Rapport d'activité professionnelle

## Interface d'aide à la décision service client

Aide service client

Rechercher une question 1

Ceci est la question de test n°12 du service client. 2 + 15

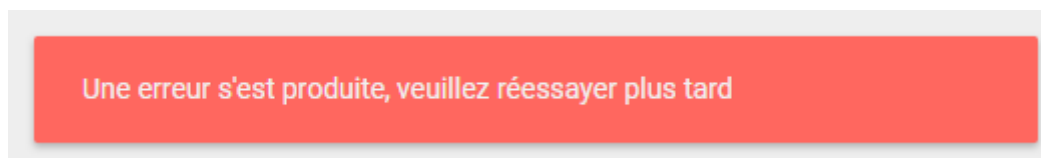
Ceci est la question de test n°2 du service client. 3  
Ceci est la réponse de la question n°2 du service client. + 10

Cette interface a pour but d'aider le service clients lors des appels, en proposant une liste de questions et réponse associées. Le personnel travaillant dans ce service pourra consulter la réponse à une question posée par le client, pour ensuite pouvoir lui répondre.

Chaque question est affichée sous forme d'un bloc (3), et lorsqu'on clique sur celui-ci, nous affichons la réponse associée en dessous de la question.

Sur chaque question listée, un bouton permettra, lorsqu'une question est demandée par le client, d'incrémenter son nombre d'utilisations pour ensuite permettre de faire des statistiques sur les questions les plus posées. Vous pouvez voir ce bouton sur la capture ci-dessus (2). Lorsque l'on clique dessus le score de la question est directement mis à jour et stocké dans la base de données. Sur cet écran il y aura aussi un champ de recherche qui permettra de filtrer la liste des questions (1). Le filtrage se fait dynamiquement lors de la saisie dans le champ recherche par l'utilisateur. Si aucune question n'est trouvée lors de la recherche, nous l'affichons à l'utilisateur.

Une erreur peut survenir lors de la récupération des questions. Dans ce cas nous informons l'utilisateur de l'erreur toujours avec une bulle de notification.



Chacune des questions et des réponses peuvent être gérées depuis une autre interface que je vais détailler maintenant.

# Rapport d'activité professionnelle

## Interface gestion des questions / réponses pour le service client

Gestion des questions réponses téléphones

1

Libellé de la question  
Qu'elle est le nom de l'entreprise ? 2

Réponse de la question  
Le nom de l'entreprise est Design' Partner. 3

AJOUTER UNE NOUVELLE QUESTION 4

Rechercher une question 5

7

Ceci est la question de test n°12 du service clientl.  
Ceci est la réponse de la question n°12 du service client. 6

Ceci est la question de test n°7 du service client sans réponse.  
Ceci est la réponse de la question n°7 du service client.

8

Ceci est la question de test n°3 du service client.  
Ceci est la réponse de la question n°3 du service client.

Ceci est la question de test n°6 du service client sans réponse.  
Ceci est la réponse de la question n°6 du service client.

Comme je l'ai dit précédemment cet écran va permettre de gérer les questions / réponses pour le service client. Nous allons retrouver sur cette interface un formulaire permettant d'ajouter ou de modifier une question et sa réponse (1), ainsi que la liste des questions / réponses.

Concernant le formulaire d'ajout et modification, il contient un champ pour renseigner la question (2) et un autre champ pour la réponse (3) (voir la capture ci-dessus). Lors de la validation du formulaire nous pouvons retrouver 3 comportements différents:



# Rapport d'activité professionnelle

Comportement initial les champs sont valides et le serveur répond:

La question / réponse est insérée dans la base de données et nous en informons l'utilisateur avec une notification.

A screenshot of a user interface showing a teal notification box with the text "La question a été ajoutée".

La question a été ajoutée

Un des champs n'est pas rempli et l'utilisateur valide le formulaire:

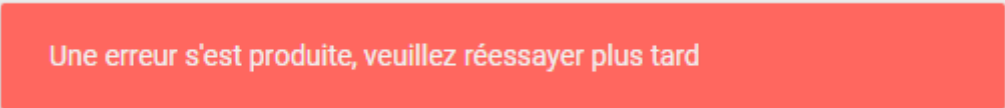
Dans ce cas aucune action n'est effectuée et nous notifions l'utilisateur que le formulaire est invalide et qu'il doit remplir tous les champs (voir la capture ci-dessous)

A screenshot of a user interface showing a red notification box with the text "La question et la réponse doivent être valides".

La question et la réponse doivent être valides

Le serveur ne répond pas l'ajout dans la base de données ne peut être effectué:

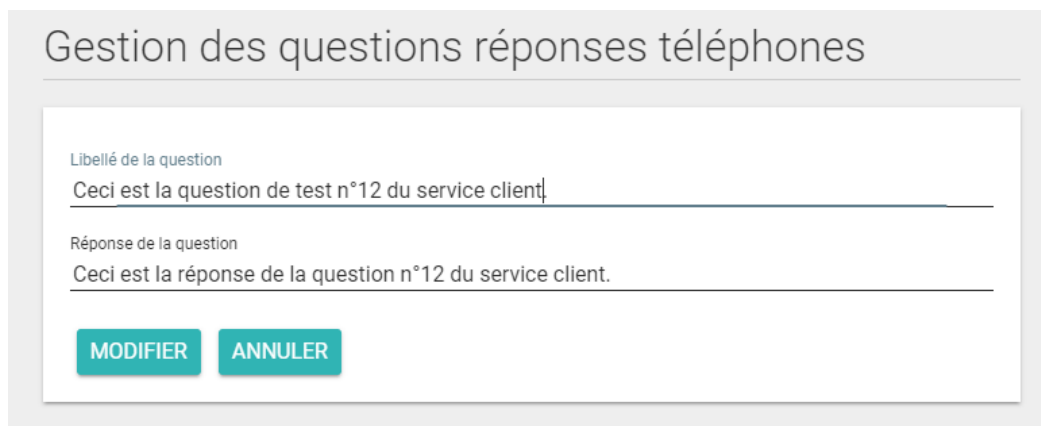
Dans ce cas informe l'utilisateur que le serveur ne répond pas et qu'il doit réessayer plus tard (voir la capture ci-dessous)

A screenshot of a user interface showing a red notification box with the text "Une erreur s'est produite, veuillez réessayer plus tard".

Une erreur s'est produite, veuillez réessayer plus tard

# Rapport d'activité professionnelle

Lorsqu'on clique sur une des questions de la liste, les champs du formulaire se remplissent avec les données de cette dernière. Le bouton de validation du formulaire change pour signaler une mise à jour. Ensuite le fonctionnement de la mise à jour d'une question est le même que celui de l'ajout. Voici une capture d'écran du formulaire de mise à jour d'une question:

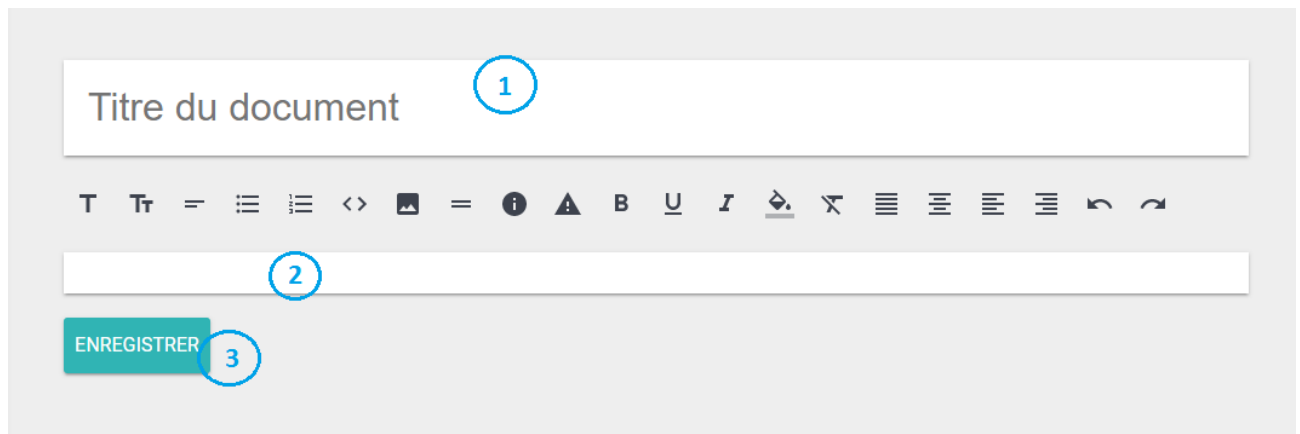


The screenshot shows a web interface titled "Gestion des questions réponses téléphones". It contains a form with two text input fields. The first field is labeled "Libellé de la question" and contains the text "Ceci est la question de test n°12 du service client". The second field is labeled "Réponse de la question" and contains the text "Ceci est la réponse de la question n°12 du service client.". Below the fields are two buttons: "MODIFIER" and "ANNULER".

Pour ce qui est de la liste nous affichons toutes les questions / réponses même celles qui ne sont pas modérées par l'administrateur (Voir la première capture en 8). Nous les différencions visuellement par leurs couleurs grises quand elles ne sont pas vérifiées par l'administrateur. Il y aura aussi un champ de recherche de questions, qui aura le même fonctionnement que celui de l'interface précédente (aide à la décision du service client). Nous pouvons retrouver ce champ sur la première capture en 5.

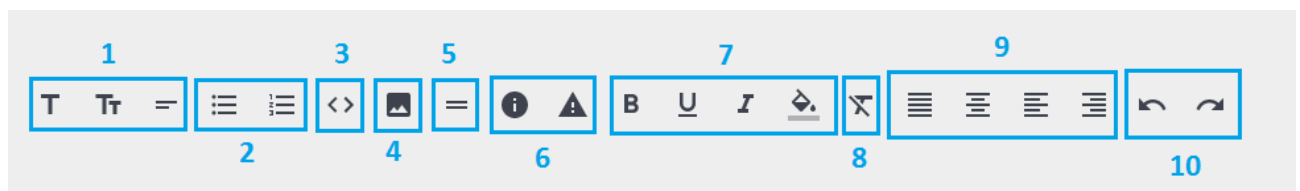
# Rapport d'activité professionnelle

## *Interface de création ou modification d'un document*



Cette interface va permettre de créer un nouveau document. Nous retrouverons un champ texte qui permettra de définir le titre du document (1). Nous aurons aussi le bloc éditable qui contiendra le contenu du document (2) et un bouton permettant d'enregistrer ce dernier. Pour enregistrer le document le titre de celui-ci est obligatoire. Dans le cas où l'utilisateur tentera d'enregistrer un document sans titre alors, une notification le préviendra que le titre est obligatoire.

Une barre d'outils sera présente en dessous du champ « titre » qui permettra de styliser le document. Je vais détailler l'ensemble de ces outils dans l'ordre. Je mettrai ensuite une capture d'écran des rendus des différents outils.



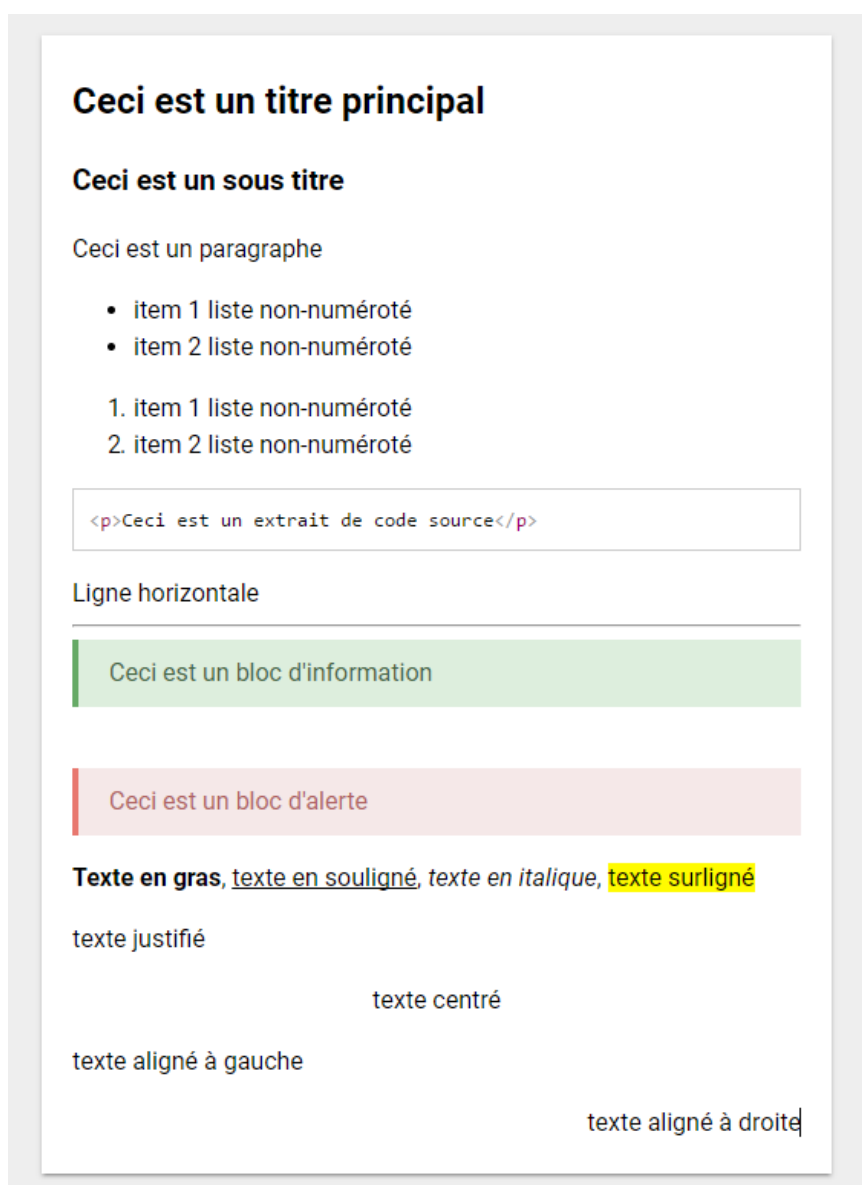
*Note: Le numéro des outils sur la capture ci-dessus, est en relation avec les numéros de la liste ci-dessous.*

1. La première icône permet de créer un titre principal, la seconde permet de créer un sous-titre et la dernière permet de créer un paragraphe.
2. On retrouve des outils permettant de créer des listes à puces numérotées ou non.
3. Insertion d'un extrait de code source dans le document, avec la colorisation syntaxique.
4. Insertion d'images.
5. Création d'une ligne horizontale

# Rapport d'activité professionnelle

6. Le premier outil permet d'insérer un bloc d'information, et le second un bloc d'avertissement.
7. Outils permettant de changer le style du texte. Dans l'ordre des icônes, mettre en gras le texte, souligné, mettre en italique et surligner le texte.
8. Permet de supprimer toute mise en forme de texte
9. Permet de changer l'alignement du texte (justifié, centré, gauche, droite)
10. Permet l'annulation des dernières modifications des documents

Rendu des différents outils:



**Ceci est un titre principal**

**Ceci est un sous titre**

Ceci est un paragraphe

- item 1 liste non-numéroté
- item 2 liste non-numéroté

1. item 1 liste non-numéroté
2. item 2 liste non-numéroté

```
<p>Ceci est un extrait de code source</p>
```

Ligne horizontale

Ceci est un bloc d'information

Ceci est un bloc d'alerte

**Texte en gras**, texte en souligné, *texte en italique*, texte surligné

texte justifié

texte centré

texte aligné à gauche

texte aligné à droite

# Rapport d'activité professionnelle

## Interface de sélection et d'affichage d'un document

Avant d'arriver sur l'affichage d'un document choisi, nous accédons d'abord à une liste permettant d'en sélectionner un pour afficher son contenu.

Accelerated Mobile Pages Advanced Concepts 2	Maintenance	Mis à jour le: 2017-04-20
Accelerated Mobile Pages Advanced Concepts 3	Service Client	Mis à jour le: 2017-04-20
Accelerated Mobile Pages Foundations	Développement	Mis à jour le: 2017-04-13
Android Things Weather Station	Autostore	Mis à jour le: 2017-04-11

Chaque document est affiché sous forme d'un bloc affichant son titre (1), sa catégorie (2) ainsi que la date de sa dernière mise à jour (3). En cliquant sur un de ces blocs nous serons redirigés vers l'interface permettant d'afficher le contenu complet du document. Voici l'aperçu de cette interface:

Accelerated Mobile Pages Foundations

Catégorie: Développement, Dernière mise à jour le 2017-04-13 par Bitc...

### 3. Run the sample page

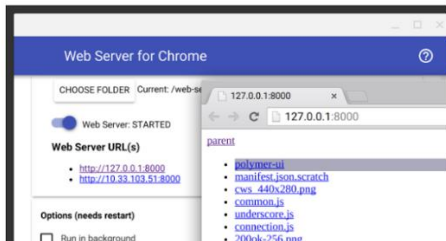
In order to test our sample page we need to access the files from a web server. There are several ways to create a temporary local web server for the purposes of testing. For this code lab we will provide instructions for 3 options available:

- **The Google Chrome app "Web Server for Chrome"** - This is the recommended approach as it is the simplest and most cross platform solution available. Note: this approach requires Google Chrome to be installed.
- **Firebase Hosting** - An alternative option if you're also interested in exploring our new static asset hosting platform "Firebase Hosting". SSL-enabled by default.
- **A local HTTP Python Server** - Requires access to the command-line.

**Note:** It is strongly recommended that HTTPS be used in production environments. HTTPS has several benefits beyond just security including SEO. You can read more about this topic [here](#).

#### Option #1: Web Server for Chrome

You can find the "Web Server for Chrome" app at [this link](#) on the Chrome Web Store.



# Rapport d'activité professionnelle

Sur cette interface nous pourrions y retrouver le titre du document (1), les informations concernant ce dernier. Auteur, date de modification et la catégorie à laquelle il appartient. Et nous retrouverons bien évidemment son contenu (3).


## Interface de modération d'un document

### Achieving Single Sign-on with AppAuth





*Categorie: Service Client, Dernière mise à jour le 2017-04-24 par Dylan*

#### 3. Run the sample app

First, let's see what the finished sample app looks like. With the code downloaded, the following instructions describe how to open the completed sample app in [Android Studio](#).

1. Open Android Studio
2. Open the `appauth-android_codelab_sso_managed` directory from the sample code folder (Select the *Open an existing Android Studio project* option on the welcome screen, or **File > Open**).
3. Plug in your Android device and click the  **Run** button. You should see the home screen of the sample app appear after a few seconds. Try signing-in to see how it works.

#### Frequently Asked Questions

-  [How do I install Android Studio?](#)
-  [How do I enable USB debugging?](#)
-  [Why doesn't Android Studio see my device?](#)
-  [Android error: Failed to install \\*.apk on device \\*: timeout?](#)

MODÉRER LE DOCUMENT

Cette interface permettra de valider le contenu d'un document et alors de rendre visible celui-ci par l'ensemble des utilisateurs de l'application. Sur cet écran nous retrouverons le contenu du document ainsi qu'un bouton qui permettra de valider / modérer le document. Une erreur peut survenir lorsque l'on clique sur "modérer le document" si le serveur ne répond pas dans ce cas l'utilisateur est informé via une notification.

## Spécifications Techniques :

Après avoir parlé des spécifications fonctionnelles du projet, je vais passer aux spécifications techniques de celui-ci. Je vais aborder ce thème en plusieurs parties. La première sera orientée vers l'environnement de travail (IDE, Gestionnaire de version). La seconde partie parlera de l'architecture de l'application et des technologies utilisées pour réaliser le front et back du projet. Je parlerai aussi de la base de données en expliquant sa structure. Je finirai par aborder certaines spécifications techniques diverses concernant la configuration des serveurs pour le déploiement de l'application (Intégration continue, environnement de dev, prod et preprod).

### **L'environnement de travail**



Pour ce qui de l'environnement de travail j'ai utilisé comme IDE PhpStorm. C'est un éditeur pour PHP, HTML, CSS et JavaScript, édité par JetBrains et sortie en 2009. L'ensemble de l'équipe de développement au sein de l'entreprise Design' Partner utilise cette IDE. De plus il intègre et possède plusieurs fonctionnalités intéressantes :

- Une coloration syntaxique
- Affichage des erreurs à la volé
- Auto-complétion intelligente du code
- Ré-usinage du code.
- L'envoi des fichiers via FTP
- Un gestionnaire de version



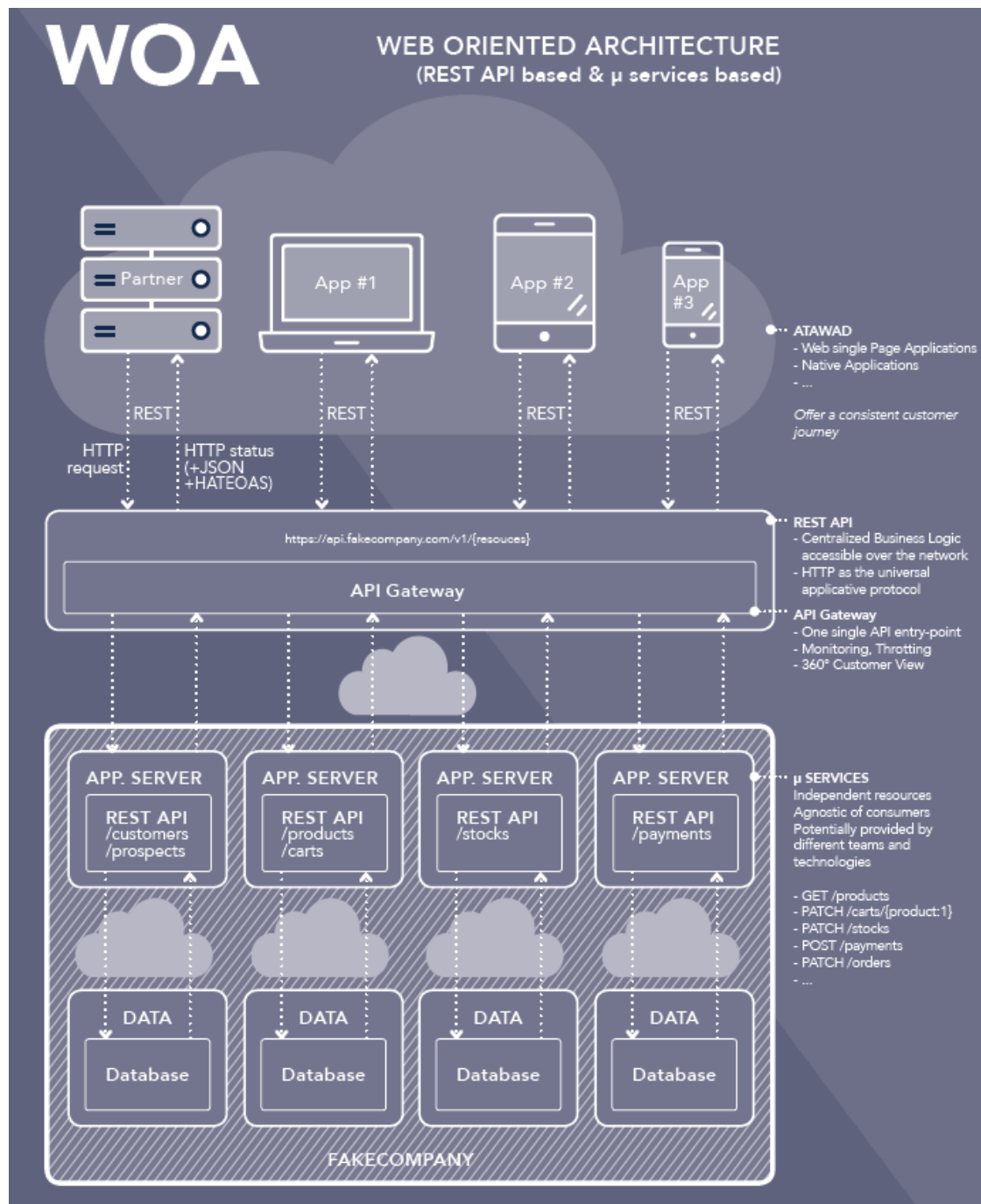
Pour la gestion de versions de mes fichiers j'ai utilisé Git. C'est un logiciel de gestion de versions décentralisé libre, créé par Linus Torvalds et sortie en 2005. L'ensemble du déploiement dans l'entreprise se fait avec Git et nous utilisons aussi Gitlab comme gestion de dépôts. Les fonctionnalités principales de Gitlab sont :

- Gérer des dépôts Git ainsi que les utilisateurs et leurs droits d'accès aux dépôts
- Authentification pouvant utiliser deux facteurs et la connexion à un annuaire LDAP
- Gérer l'accès par branche à un dépôt
- Effectuer des examens de code et renforcer la collaboration avec les demandes de fusion
- Chaque projet possède un outil de ticket et un wiki

# Rapport d'activité professionnelle

## Architecture et détails techniques de l'application

Je vais aborder en premier point parler de l'architecture de l'application. J'ai choisi de partir sur une architecture orientée web (WOA) en créant une API Rest. Cette architecture va permettre de séparer la partie front end du back end et de rendre totalement indépendants la partie client de la partie serveur. Voici un petit schéma permettant d'illustrer cette architecture :





# Rapport d'activité professionnelle

On retrouve 3 niveaux d'API dans notre cas nous serons sur le niveau 1. L'API est destinée à être consommée que par les applications développées au sein de l'entreprise. Voici un détail des 3 niveaux pour mieux comprendre.

- Niveau 1 « Private API » : l'API est destinée à être consommée par les applications développées au sein de l'entreprise.
- Niveau 2 « Partners API » : l'API est destinée à être consommée par les applications développées au sein de l'entreprise ou par des partenaires.
- Niveau 3 « Open API » : l'API est destinée à être consommée par tout type de développeurs.



La différence entre chaque niveau réside principalement dans la qualité du processus d'industrialisation de consommation des services qu'il convient de mettre en œuvre :

- au niveau 1, un développeur d'application est contraint de consommer l'API et peut en cas de difficultés directement contacter l'équipe.
- au niveau 3, le succès d'une API, qui peut être consommée par des milliers d'utilisateurs sur la toile, dépend de la facilité du processus d' enrôlement et des qualités intrinsèques de l'API et de sa documentation

Je vais maintenant parler brièvement du style d'architecture Rest, en évoquant les contraintes de celles-ci.

Une architecture REST doit respecter les contraintes suivantes :

- Client-serveur : les responsabilités sont séparées entre le client et le serveur. L'interface utilisateur est séparée de celle du stockage des données. Cela permet aux deux d'évoluer indépendamment.
- Sans état : chaque requête d'un client vers un serveur doit contenir toute l'information nécessaire pour permettre au serveur de comprendre la requête, sans avoir à dépendre d'un contexte conservé sur le serveur. Cela libère de nombreuses interactions entre le client et le serveur.

# Rapport d'activité professionnelle

- Mise en cache : toute réponse du serveur comprend des indications quant à la possibilité de mettre en cache cette réponse, comme sa fraîcheur, sa date de création, sa validité future.
- Une interface uniforme, cette contrainte agit selon quatre règles essentielles :
  - l'identification des ressources : chaque ressource est identifiée unitairement
  - la manipulation des ressources à travers des représentations : les ressources ont des représentations définies
  - un message auto-descriptif : les messages expliquent leur nature. Par exemple, si une représentation en HTML est encodée en UTF-8, le message contient l'information nécessaire pour dire que c'est le cas
  - hypermédia comme moteur d'état de l'application : chaque accès aux états suivants de l'application est décrit dans le message courant.
- Un système hiérarchisé par couches : les états de l'application sont identifiés par des ressources individuelles.

Après avoir parlé de l'architecture de l'application je vais maintenant détailler le back end. Je vais aussi aborder l'architecture de celui-ci, ensuite je parlerai des technologies et framework utilisé et je finirai par quelques exemples.

## Controller :

Le back end se compose de plusieurs couches. En premier nous retrouvons la couche « Controller », elle va permettre au front end d'interroger l'API afin de récupérer des données. Cette couche « Controller » respecte le design d'une API RESTful, en utilisant les bonnes méthodes HTTP en fonction des actions demandées:

- |          |                      |
|----------|----------------------|
| • GET    | renvoie des données  |
| • POST   | ajoute des données   |
| • PUT    | modifie des données  |
| • DELETE | supprime des données |

# Rapport d'activité professionnelle

Dans un cas concret nous aurons ceci :

- GET /users/            récupère les utilisateurs
- POST /users/        ajoute un utilisateur
- PUT /users/1        modifie l'utilisateur dont l'ID est 1
- DELETE /users/1    supprime l'utilisateur dont l'ID est 1

## Model :

La deuxième couche du back end est la couche « model » qui va contenir l'ensemble des données de l'application. L'ensemble des models correspond au schéma de la base de données dont je parlerai par suite. Un model correspond à une table de la base de données.

## Dao :

La troisième couche de l'application c'est la couche « DAO » elle va permettre d'interroger et de récupérer les données dans une base de données distante. Chaque model possède une DAO qui hérite d'une DAO générique que j'ai créé. Nous n'utiliserons pas d'ORM (Object Relational Mapping) dans ce projet.

## Service :

La couche « service » est la dernière du back end. Elle va servir de passerelle entre les « DAO » et les « Controllers », elle va permettre aux Controllers de ne pas interroger directement la DAO, c'est dans cette couche que sera déplacée la complexité du code. Ainsi les « Controllers » et la « DAO » seront très peu complexes ce qui facilitera la maintenance et la compréhension du code par l'équipe de développement.

Pour réaliser le back end j'ai utilisé comme langage de programmation PHP 7 avec le framework Silex. Silex est un micro framework basé sur « Symfony ». J'ai choisi ce framework car il est très modulable, léger et facile à mettre en place. De plus pour créer l'API j'avais seulement besoin des « Services Controllers » proposés par Silex.

Pour mieux comprendre, l'architecture et les technologies utilisées pour le back end voici un exemple des différentes couches. Pour chaque capture je détaillerai le fonctionnement.

# Rapport d'activité professionnelle

```
1 <?php
2 use Symfony\Component\HttpFoundation\Request;
3
4 /**
5  * On récupère tous les utilisateurs
6  */
7 $app->get('/users', function (Request $request) use ($app) {
8     $statutCode = 200;
9     try {
10         $params = $request->query->all();
11         $userService = new UserService();
12         $data = $userService->getUsers($params);
13     } catch (Exception $e) {
14         $statutCode = 404;
15         $data = ['message' => $e->getMessage(), 'exception' => $e];
16     }
17     return $app->json($data, $statutCode);
18 });
```

Voici un modèle de « Controller » avec une route qui va permettre ici de récupérer tous les utilisateurs de l'application. Comme nous pouvons le voir dans le code nous utilisons bien la méthode « GET » pour récupérer un contenu, le nom de la route est très explicite « users » et pour finir nous renvoyons le contenu à l'utilisateur avec un statut 200, dans le cas d'une erreur un statut 404 avec le détail de cette dernière. Nous pouvons voir aussi que pour récupérer l'ensemble des utilisateurs nous passons par une couche service en lui passant les paramètres de la requête.

```
1 <?php
2
3 /**
4  * Class UserService
5  */
6 class UserService
7 {
8     /**
9      * UserService constructor.
10     */
11     public function __construct()
12     {
13         $this->userDAO = new UserDAO();
14     }
15
16     /**
17      * @param $params
18      *
19      * @return mixed
20     */
21     public function getUsers($params)
22     {
23         if (isset($params['login']) && isset($params['password'])) {
24             $data['user'] = current($this->userDAO->connection($params));
25         } else {
26             $data['users'] = $this->userDAO->findAll();
27         }
28         return $data;
29     }
30 }
```

# Rapport d'activité professionnelle

Cette couche service contient l'ensemble de la complexité du code et fais appel à la DAO qui elle va interroger la base de données pour y récupérer les utilisateurs.

```
106  /**
107   * @param array $condition
108   *
109   * @return array
110   *
111   * @throws Exception
112   *
113   * Requête permettant de récupérer tous le contenu d'une table
114   */
115  public function findAll($condition = [])
116  {
117      $result = [];
118      $sqlCondition = "";
119      $paramsCondition = [];
120      if (!empty($condition)) {
121          $sqlCondition = $this->generateConditionRequest($condition);
122          $paramsCondition = array_values($condition);
123      }
124      try {
125          if ($cnx = self::getConnection()) {
126              $sql = "SELECT..." . implode(',', $this->getColumns()) . " FROM..." . $this->getTable() . $sqlCondition;
127              $request = $cnx->prepare($sql);
128              if ($request->execute($paramsCondition)) {
129                  $result = $request->fetchAll(PDO::FETCH_ASSOC);
130              }
131          }
132      } catch (Exception $e) {
133          throw new Exception($e->getMessage());
134      }
135
136      return $result;
137  }
```

Voici une des méthodes de la classe générique DAO. Celle-ci va permettre de récupérer tous le contenu d'une table. Dans notre cas nous allons récupérer l'ensemble des utilisateurs. Le résultat sera retourné sous la forme d'une liste d'utilisateurs.

## **Base de données :**

Parlons maintenant de la base de données et de sa structure. Avant de créer la base de données sur le SGBD (Système de gestion de base de données) MariaDB, j'ai d'abord réalisé un MCD (Modèle conceptuel de données) sur MySQL Workbench. Je vais vous présenter le model en détaillant chacune des relations et des entités de celui-ci.

Petite présentation de MySQL Workbench, avant de détailler la structure de la base de données.

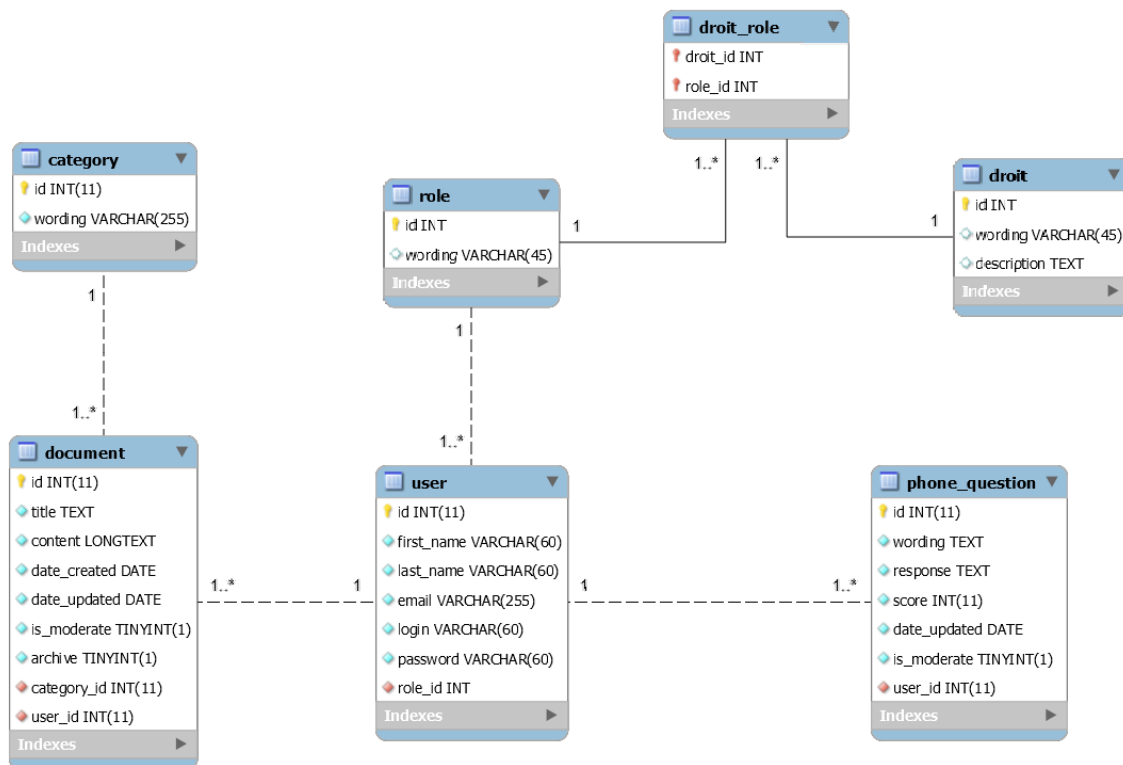


MySQL Workbench est un logiciel de gestion et d'administration de bases de données MySQL créé en 2004. Via une interface graphique, il permet, entre autres, de créer, modifier ou supprimer des tables, des comptes utilisateurs, et d'effectuer toutes les opérations inhérentes à la gestion d'une base de données. Il permet aussi de concevoir et de

# Rapport d'activité professionnelle

modéliser des bases de données, et donne la possibilité de générer ces dernières à partir du modèle et inversement.

Voici le modèle conceptuel de la base de données du projet :



L'entité « user » a comme propriétés first\_name, last\_name, email, login, password qui aura un hashage en SHA1 et la propriété role\_id qui permet de faire la relation avec l'entité « role ». Chaque utilisateur appartient à 1 et 1 seul rôle, et chaque rôle peut avoir plusieurs utilisateurs.

Si nous continuons les relations nous pouvons voir que l'entité « role » possède une propriété wording qui va permettre de nommer le rôle. La table « role » est associée à la table « droit » qui a pour propriété wording et description. D'après les relations nous voyons qu'un rôle peut avoir plusieurs droits et qu'un droit peut appartenir à plusieurs rôles.

Pour ce qui est de la relation « category », « document » et « user », la clé étrangère user\_id dans document permet d'identifier l'auteur du document. Nous pouvons retrouver dans l'entité « document » la propriété « archive » qui va permettre de passer un document en « archivé » sans le supprimer. Nous avons aussi le champ « is\_moderate » qui va permettre de modérer / valider un document par l'administrateur.

L'entité « phone\_question » va permettre de gérer les questions / réponses de l'outil d'aide à la décision pour le service client. Nous y retrouvons la clé étrangère « user\_id » qui va permettre d'identifier, comme pour « document », l'auteur des questions / réponses. Chaque question / réponse est relié 1 et 1 seul utilisateur, chaque utilisateur peut avoir 0 ou plusieurs questions / réponses.

Dans la couche model de l'application, nous aurons une classe qui va représenter chacune des entités de la base de données, qui servira au mapping des données lors des requêtes.

Pour sécuriser l'ensemble des requêtes nous utiliserons des requêtes préparées dans notre DAO pour éviter les injections SQL et l'extension PDO (PHP Data Objects) dans notre DAO ce qui permettra une meilleure portabilité du code en cas de migration de SGBD (Système de gestion de base de données). Pour information voici la liste des pilotes disponibles pour PDO :

- 4D
- CUBRID
- Firebird
- IBM DB2 (avec ou sans ODBC)
- Informix
- Microsoft SQL Server
- MySQL
- Oracle Database
- PostgreSQL
- SQLite

## **Le front end et Polymer**

J'ai détaillé l'ensemble des spécifications techniques pour le back end je vais maintenant aborder celle du front end. La partie front du projet a été réalisée en HTML, CSS, JavaScript à l'aide de web components. Pour bien comprendre la suite je vais rappeler ce que sont les web components et je détaillerai ensuite le framework Polymer avec lequel j'ai réalisé le front end.

### **Les web components :**

Les composants Web sont constitués de plusieurs technologies distinctes. Vous pouvez les voir comme des composants d'interface graphique réutilisables, qui ont été créés en utilisant des technologies web libres. Ils font partie du navigateur, et donc ne nécessitent pas de bibliothèque externe comme jQuery ou Dojo. Un Composant Web existant peut être utilisé sans l'écriture de code, en ajoutant simplement une

déclaration d'importation à une page HTML. Les Composants Web utilisent les nouvelles capacités standard de navigateur, ou celles en cours de développement.

Les composants Web sont constitués de quatre technologies (bien que chacune puisse être utilisée séparément) :

- Custom Elements: pour créer et enregistrer de nouveaux éléments HTML et les faire reconnaître par le navigateur.
- HTML Templates: squelette pour créer des éléments HTML instanciables.
- Shadow DOM: permet d'encapsuler le JavaScript et le CSS des éléments.
- HTML Imports: pour packager ses composants (CSS, JavaScript, etc.) et permettre leur intégration dans d'autres pages.

Polymer :



Polymer est une bibliothèque JavaScript open-source permettant de construire des applications web à partir de web components. La bibliothèque est développée par les développeurs et contributeurs Google sur GitHub. La bibliothèque est les composants sont tous basés sur les principes de conception de Google Material Design.

Polymer est utilisé par un certain nombre de services et de sites web Google, comme YouTube (Nouvelle Version), YouTube Gaming, les sites Web Google I/O, Google Play Music et Google Sites.

Polymer fournit un certain nombre de fonctionnalités sur les composants Web:

- Simplification de création d'éléments personnalisés
- La liaison de données directionnelle et bidirectionnelle
- Propriétés calculées
- Modèles conditionnels et répétitifs (dom-if, dom-repeat)
- Gestion des évènements
- Bibliothèque d'éléments (Polymer Element Catalog)

Aujourd'hui nous sommes à la version 2 de Polymer cependant j'ai utilisé pour ce projet la version 1, car Polymer 2 n'était pas encore sortie au moment du projet. Malgré leurs différences les 2 versions restent compatibles l'une et l'autre, les prochaines fonctionnalités du projet seront donc développées avec la version 2 de Polymer.



# Rapport d'activité professionnelle

Il existe aussi une interface en ligne de commande pour les projets Polymer appelé « Polymer CLI » qui comprend un pipeline de construction, un générateur de code pour créer des éléments et des applications, un linter permettant l'analyse statique du code source, un serveur de développement et un runner de test.

Pour mieux comprendre Polymer je vais parler de la création et de la structure d'un projet, puis celle d'un web component avec Polymer 1. La création d'un web component et d'un projet Polymer se fait à l'aide de Polymer CLI.

## Création et structure d'un projet Polymer :

Il faut d'abord créer un dossier et s'y placer à la racine :

```
mkdir my-app  
cd my-app
```

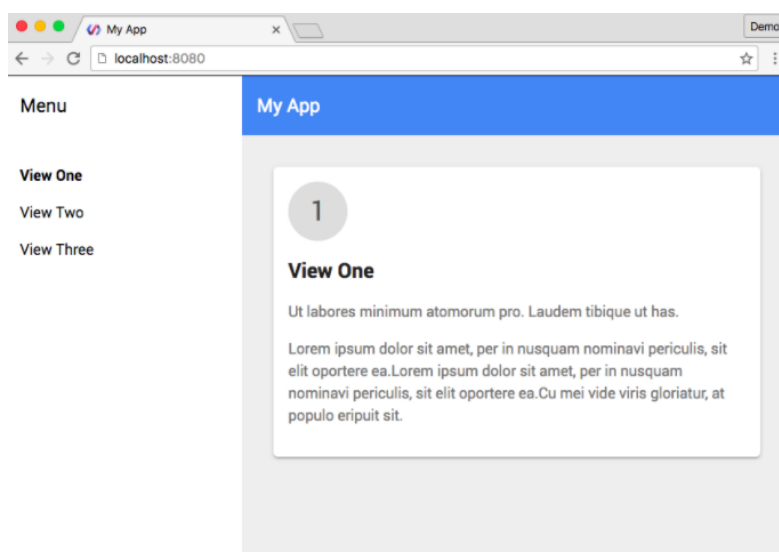
Ensuite nous initialisons le projet via la commande suivante :

```
polymer init starter-kit
```

Une fois le projet créé nous le lançons avec la commande suivante depuis le Polymer CLI :

```
polymer serve --open
```

La commande ouvre automatiquement un nouvel onglet avec le projet. Ici nous sommes partis du starter kit, nous obtenons donc l'affichage suivant :



# Rapport d'activité professionnelle

Voici la structure générée du projet :

```
/
|---index.html
|---src/
|---bower_components/
|---images/
|---test/
```

- **index.html** Est le principal point d'entrée de l'application
- **src/** Contient tous les éléments personnalisés du projet
- **bower\_components/** Contient tous les éléments personnalisés réutilisables et / ou les bibliothèques récupérer via Bower
- **images/** Contient toutes les images statiques du projet
- **test/** C'est le dossier qui contient les tests que l'on a défini pour les web components

## Création et structure d'un web component Polymer :

Comme pour le projet nous créons un dossier et nous nous positionnons à la racine de celui-ci :

```
mkdir my-el && cd my-el
```

Ensuite nous exécutons la commande suivante et nous sélectionnons « element » pour créer le web component :

```
polymer init
```

On entre ensuite le nom du web component, ensuite sa description et le web component est créé.

Voici la structure générée d'un composant web Polymer :

- **bower.json** Fichier de configuration de Bower
- **bower\_components/** Les dépendances du projet. Voir Gérer les dépendances
- **demo/index.html** Démo de l'élément
- **index.html** Documentation de l'élément générée automatiquement
- **my-el.html** Code source de l'élément
- **test/my-el\_test.html** Tests unitaires de l'élément

J'ai utilisé ces méthodes pour l'ensemble du front de l'application, je parlerai de l'ensemble des web components créés pour le projet, dans la partie réalisation du rapport. Aussi je détaillerai les principaux web components du projet durant l'oral, cela permettra de mieux comprendre le sujet et d'interagir avec le jury.

## **Autres spécifications techniques diverses :**

Pour la réalisation du projet nous utiliserons 3 environnements. Un environnement de « DEV » qui servira lors de la phase de développement du projet. Pour ce projet l'environnement de DEV sera un Wamp serveur, avec PHP 7.0.10, Apache 2.4.23, MySQL 5.7.14 et PhpMyAdmin 4.6.4.

Le deuxième environnement est celui de « PREPROD ». Il possède la même configuration que l'environnement de « PROD » à quelque exception près. Il permet de tester l'application et / ou les fonctionnalités avant le déploiement en production. Nous retrouvons sur cet environnement un serveur d'intégration continu « Gitlab CI ». Pour comprendre Gitlab CI voici une définition de l'intégration continue :

L'intégration continue est un ensemble de pratiques utilisées en développement pour vérifier, à chaque modification de code source, que le résultat des modifications ne produit pas de régression dans l'application développée. Le concept d'intégration continu se réfère généralement à la pratique de l'extreme programming. Le principal but de cette pratique est de détecter les problèmes d'intégration au plus tôt lors du développement. De plus, elle permet d'automatiser l'exécution des suites de tests et de voir l'évolution du développement de l'application.

Et en dernier nous avons l'environnement de PROD sur lequel nous déployons l'application pour les utilisateurs finaux. Nous retrouvons sur cet environnement de production, un serveur Apache, avec une base de données MariaDB et PHP.

## Réalisation :

La réalisation du projet s'est faite en plusieurs phases. La première a été la phase de réflexions du projet. Le projet étant réalisé pour un usage interne à l'entreprise, j'ai dans un premier temps regroupé les besoins et attentes du projet définis par Maxence Corbrion, chef de l'entreprise de Design' Partner. Pendant cette phase du projet j'ai aussi réfléchi aux outils et technologies que j'allais utiliser pour le projet, étant donné que je n'avais aucune contrainte sur le choix des langages. Après avoir fixé les éléments de base du projet je suis passé à la phase de conception de l'application en réalisant dans un premier temps des mockups sur papier, un modèle conceptuel de données (MCD) et un schéma de la structure de l'application.

Une fois l'étape de conception terminée, j'ai d'abord réalisé le back end du projet. J'ai commencé par créer la structure de l'application avec les différentes couches. La couche « Model », « Controller », « DAO » et la couche « Service ». J'ai ensuite développé au sein de la couche DAO une classe générique en PHP permettant d'effectuer des requêtes sur la base de données. À la suite de ça j'ai créé les models et les DAO correspondants à chaque entité de la base de données. Une fois la DAO et les models créés j'ai créé la base de données à l'aide de PhpMyAdmin.

Ayant les données et une DAO fonctionnel, je suis passé à la couche « Controller » du projet. Comme je n'appelle pas la DAO directement dans les controllers j'ai développé en parallèle des « Controller », les différentes classes de « Service ». Nous retrouvons dans l'application une classe service et un controller par entité. Pour les « Controller » j'ai développé l'ensemble des routes nécessaire au projet.

Voici un exemple du Controller Document, avec un commentaire qui définit l'utilité de chaque route. Pour chaque route, nous retrouvons une structure identique. Nous avons une gestion des exceptions avec les « try catch », nous retournons le Code HTTP associé à la méthode utilisées (GET, POST, PUT, DELETE, ...) en cas d'échec ou de réussite. Nous retournons aussi l'exception levée en cas d'échecs et les données récupérées en cas de succès.

# Rapport d'activité professionnelle

Voir la capture ci-dessous.

```
20 //On récupère un document précis
21 $app->get('/document/{id}', function ($id) use ($app) {
22     $statutCode = 200;
23     try {
24         $documentService = new DocumentService();
25         $data = $documentService->getDocumentById($id);
26     } catch (Exception $e) {
27         $statutCode = 404;
28         $data = ['message' => $e->getMessage(), 'exception' => $e];
29     }
30
31     return $app->json($data, $statutCode);
32 });
33 //On ajoute un document
34 $app->post('/document', function (Request $request) use ($app) {
35     $statutCode = 201;
36     try {
37         $documentService = new DocumentService();
38         $data = $documentService->addDocument($request->request->all());
39     } catch (Exception $e) {
40         $statutCode = 400;
41         $data = ['message' => $e->getMessage(), 'exception' => $e];
42     }
43
44     return $app->json($data, $statutCode);
45 });
46 //On modifie un document
47 $app->put('/document/{id}', function (Request $request, $id) use ($app) {
48     $statutCode = 200;
49     try {
50         $documentService = new DocumentService();
51         $data = $documentService->updateDocument($request->request->all(), $id);
52     } catch (Exception $e) {
53         $statutCode = 400;
54         $data = ['message' => $e->getMessage(), 'exception' => $e];
55     }
56
57     return $app->json($data, $statutCode);
58 });
59 //On supprime un document
60 $app->delete('/documents/{id}', function ($id) use ($app) {
61     $data = [];
62     $statutCode = 204;
63     try {
64         $documentDAO = new DocumentDAO();
65         $documentDAO->deleteById($id);
66     } catch (Exception $e) {
67         $statutCode = 400;
68         $data = ['message' => $e->getMessage(), 'exception' => $e];
69     }
70
71     return $app->json($data, $statutCode);
72 });
```

Tous au long du développement je testais les controllers avec CURL, pour vérifier que chaque route me renvoyer les bonnes informations.

Une fois le back end testé et opérationnel je suis passé au développement du front end. J'ai commencé par créer une application Polymer (Version 1). Après la création du squelette de l'application par le client Polymer (Polymer CLI), j'ai créé un thème CSS pour le template du projet à l'aide de Material Palette, un site permettant de créer des thèmes pour différents structure d'application (Android, Polymer, ...). Une fois le thème intégré j'ai commencé par crée un web component qui permet d'interroger de manière asynchrone le back end (l'API) et de gérer la relation entre le back et le front. Ce web component retourne le statut de la réponse et le résultat de celle-ci quand la requête est terminée. Le web component prend en paramètre un fichier de configuration qui pourra être différents selon qu'on soit dans l'environnement de PROD ou de DEV.

J'ai ensuite réalisé une des parties les plus importantes du projet, celle qui permet de créer un document. Vous pouvez retrouver le détail de cette partie dans les spécifications fonctionnelles du projet. Après cela j'ai réalisé l'ensemble des écrans de base qui sont définis dans les besoins et spécifications du projet.

Les fonctionnalités de base du projet sont développées, je suis actuellement dans une phase d'amélioration et d'ajout de fonctionnalités supplémentaires. Par la suite il y aura une phase de déploiement de l'application, tous d'abord en PREPROD pour réaliser les tests et en PROD une fois les tests validés.

## Gestion de projet :

Pour la gestion du projet, j'ai utilisé les méthodes agiles. Ce sont un ensemble de règles de pilotage et de réalisation de projets. Les méthodes agiles se veulent plus efficaces que les méthodes traditionnelles. Elles impliquent au maximum le demandeur (client) et permettent une grande réactivité à ses demandes.

Les méthodes agiles reposent sur un cycle de développement itératif, incrémental et adaptatif. Elles doivent respecter quatre valeurs fondamentales déclinées en douze principes desquels découlent une base de pratiques, soit communes, soit complémentaires.

### Les méthodes agiles prônent 4 valeurs fondamentales :

- Individus et interactions plutôt que processus et outils
- Fonctionnalités opérationnelles plutôt que documentation exhaustive
- Collaboration avec le client plutôt que contractualisation des relations
- Acceptation du changement plutôt que conformité aux plans

### Les Douze principes généraux des méthodes agiles :

- Satisfaire le client en priorité. Des versions fonctionnelles du logiciel doivent être régulièrement livrées au client, de manière rapprochée dans le temps.
- Les développeurs doivent bien accueillir et de manière réactive des demandes d'évolution même s'ils sont déjà bien avancés dans le développement. L'objectif est de permettre l'acquisition d'un réel avantage concurrentiel pour le client.
- Les développeurs doivent réduire les cycles de développement et livrer des versions opérationnelles aux clients dans un laps de temps le plus court possible, entre 2 semaines et 2 mois.
- Assurer une coopération permanente entre le client et l'équipe projet. Les utilisateurs métier (clients du logiciel) et les développeurs doivent travailler ensemble quotidiennement.
- Construire des projets autour d'individus motivés, donnez-leur l'environnement de travail et le support dont ils ont besoin et surtout faites-leur confiance pour que le travail soit fait.

# Rapport d'activité professionnelle

- Privilégier la conversation en face à face. La méthode la plus efficace pour communiquer des informations à une équipe projet.
- Mesurer l'avancement du projet en termes de fonctionnalités de l'application. Le bon fonctionnement du logiciel est la première mesure de progression.
- Faire avancer le projet à un rythme soutenable et constant. Les développeurs et les sponsors du projet doivent pouvoir maintenir indéfiniment un rythme de travail régulier.
- Porter une attention continue à l'excellence technique et à la conception
- Faire simple. Eliminer le travail inutile et superflu. Il faut savoir répondre au besoin exprimé par le client de manière informatiquement simple, avec un développement qui peut facilement évoluer.
- Responsabiliser les équipes. Les responsabilités ne sont pas la possession d'un chef de projet mais sont partagées par chaque membre de l'équipe agile.
- A intervalles réguliers, l'équipe agile s'interroge sur la manière d'améliorer encore son efficacité, puis règle et ajuste son comportement en conséquence.

Dans le cadre du projet, j'ai réalisé 3 réunions, pendant la phase de développement, pour montrer l'avancement du projet au client, qui dans notre cas interne à l'entreprise. Ces réunions ont permis d'avoir un retour sur l'application pour ensuite l'améliorer et effectuer les changements demandés par le client. Le projet étant en cours de développement d'autres revues sont déjà prévu.

Le développement de l'application ne suit aucun planning défini au départ du projet. Nous réalisons le projet par itération, en développant que des morceaux fonctionnels de l'application. Le projet est donc en constante évolution et le client peut suivre celle-ci et nous faire part de son point de vue et / ou des changements qu'il souhaite apporter. Etant donné que j'ai commencé le développement de l'application seul, les premières itérations ont été faites tous les mois. Elles seront réduites car dans l'entreprise nous utilisons la méthode agile SCRUM, avec une longueur de sprint de 2 semaines. L'amélioration et l'ajout de fonctionnalité se feront avec l'équipe de développement et des storys seront intégrées au planning SCRUM de l'équipe. Dès que l'on aura une première version stable de l'application nous déploierons celle-ci sur l'environnement de PREPROD pour la tester, puis sur l'environnement de PROD. En cas de bugs remontés par les utilisateurs de l'application nous allons ajouter des tâches de dette techniques ou de nouvelle story si les modifications à apporter au projet sont trop importantes.



## Conclusion :

Ce projet a été et continu d'être très intéressant sur plusieurs points. Il m'a permis d'accroître mes connaissances sur le framework Polymer mais aussi sur la structure, le fonctionnement et les bonnes pratiques de mise en place d'une API Rest. Aussi ce projet m'a permis de réaliser des activités que je ne réalise pas habituellement dans l'entreprise telle que le maquettage, création de diagrammes et la gestion de projet. Cela m'a permis d'avoir une vision plus large sur la création et le déroulement d'un projet.

Malgré ces points positifs que m'a apportés le projet, je me remets en question sur certains points du projet. Par exemple ce qui auraient pu être mieux c'est de faire du TDD (test-driven development) ou en français développement piloté par les tests. Cela m'aurait certainement permis de gagner du temps sur la partie back end du projet et d'avoir une couverture de code optimal. Autre point que je n'ai pas réalisé dans le projet, c'est la mise en place d'OAuth 2 pour sécuriser l'API. Cependant je travaillerai prochainement sur ce point, étant donné que l'application est toujours en développement et sera en constante évolution.

Je vais maintenant passer à la conclusion sur les 2 années d'alternance. Elles m'ont dans un premier temps permis d'accroître mes connaissances dans la conception et le développement d'applications grâce au partage de connaissances et aux retours d'expérience des formateurs. Un des plus gros points positif des 2 années d'alternance, c'est l'entrée dans le monde professionnel. En effet j'ai la chance d'avoir décroché un CDI dès la fin de mon alternance et je voudrais remercier une fois de plus Maxence Corbrion mon employeur pour cette offre.

Pour terminer ce rapport je vais parler de mes projets pour le futur, qui sont d'accroître mes connaissances en développement et gestion de projet, d'apporter ma contribution à l'entreprise Design' Partner grâce à mon CDI et d'apprendre de nouvelles technologies telles que Angular 4 et NodeJS.

## Glossaire :

**API** : (Applications Programming Interface) interface de programmation composé d'un ensemble normalisé de classes, de méthodes ou de fonctions qui permet de se « brancher » sur une application pour échanger des données.

**Rest** : (Representational State Transfer) Le protocole REST constitue un style architectural et un mode de communication fréquemment utilisé dans le développement de services Web.

**Scrum** : Méthode agile dédiée à la « gestion de projet ».

**Méthode Agile** : Approche itérative et collaborative, capable de prendre en compte les besoins initiaux du client et ceux liés aux évolutions.

**TDD** : (test-driven development) ou en français développement piloté par les tests est une technique de développement de logiciel qui préconise d'écrire les tests unitaires avant d'écrire le code source d'un logiciel.

**MySQL** : Système de gestion de bases de données relationnelles (SGBDR)

**SGBD** : (Système de Gestion de Bases de Données) logiciel qui stocke des données de façon organisées et cohérentes.

**SQL** : (Structured Query Language) en français langage de requête structurée, c'est un langage informatique normalisé servant à exploiter des bases de données relationnelles.

**PHP** : (Hypertext Preprocessor) langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP.

**CSS** : (Cascading Style Sheet) ou feuille de style en cascade en français, il permet d'insérer des styles sur un code HTML ou XHTML.

**HTML** : (HyperText Markup Language) format de données conçu pour représenter les pages web. C'est un langage de balisage permettant d'écrire de l'hypertexte, d'où son nom.

**Polymer** : Bibliothèque JavaScript open-source permettant de construire des applications web en utilisant des composants web.

# Rapport d'activité professionnelle

**Web Component** : Composants web en français, permettent aux développeurs de créer des balises HTML personnalisées et réutilisables.

**Shadow DOM** : Permet d'encapsuler du JavaScript et du CSS au sein d'un Web Component.

**PDO** : (PHP Data Objects) extension définissant l'interface pour accéder à une base de données avec PHP.

**Silex** : Micro framework web écrit en PHP et basé sur Symfony, Twig (template engine) et Doctrine (abstraction de base de données).

**HTTP** : (HyperText Transfer Protocol) protocole de transfert hypertexte en français, c'est un protocole de communication client-serveur développé pour le World Wide Web. HTTPS (avec S pour secured, soit « sécurisé ») est la variante du HTTP sécurisée par l'usage des protocoles SSL ou TLS.

**OAuth** : Protocole libre qui permet d'autoriser un site web, un logiciel ou une application (dite « consommateur ») à utiliser l'API sécurisée d'un autre site web (dit « fournisseur ») pour le compte d'un utilisateur.

**MCD** : (Modèle conceptuel de données) modèle de données ou diagramme pour des descriptions de haut niveau de modèles conceptuels de données.

**Mockup** : Un mockup représente un squelette d'une interface utilisateur. Ce squelette est en noir et blanc et à principalement pour but de montrer sur papier (ou sur ordinateur) où seront placés les éléments sur l'interface et comment fonctionnera l'interaction entre eux (bouton d'action, zone de texte, élément interactif ...).

**ORM** : (Object Relational Mapping) technique de programmation informatique qui crée l'illusion d'une base de données orientée objet à partir d'une base de données relationnelle en définissant des correspondances entre cette base de données et les objets du langage utilisé.

## Annexes






## MOCKUP ECRAN DE CONNEXION

The mockup consists of three main parts:


- Browser Window:** A grey header bar at the top contains navigation icons (back, forward, home, search, close) and a URL bar with the text "http://". Below the icons, the text "A Web Page" is displayed.
- Central Login Form:** A white rectangular box in the center contains three input fields. The top field is labeled "Identifiant", the middle field is labeled "Mot de passe", and the bottom field is labeled "Connexion".
- Bottom Navigation Bar:** A grey bar at the bottom contains a "Connexion" button and a "Mot de passe" label.

Additional elements include a yellow sticky note on the left side with the text "Infobulle qui affichera les notification concernant la connexion" and a red tab on the right side.

## MOCKUP ECRAN DE GESTION QUESTIONS / REPONSES SERVICE CLIENT




http://



A Web Page

Service Client



Accueil

Service Client

Préparation

Production

Maintenance

Développement

Administration

Ajouter question / réponse

Question

Réponse

Ajouter

Liste des questions / réponses

Que signifie CSS ?  
CSS signifie Cascading Style Sheets et sert à styler des éléments HTML.

Que signifie HTML ?  
HTML signifie HyperText Markup Language.

Que signifie PHP ?  
PHP signifie Hypertext Preprocessor.

## ECRAN DE GESTION QUESTIONS / REPONSES SERVICE CLIENT

Design' Partner

Service Client

Accueil

Service Client

Préparation

Production

Maintenance

Développement

Administration

Rechercher une question

Ceci est la question de test n°12 du service client.  
Ceci est la réponse de la question n°12 du service client.

Ceci est la question de test n°7 du service client sans réponse.  
Ceci est la réponse de la question n°7 du service client.

Ceci est la question de test n°3 du service client.  
Ceci est la réponse de la question n°3 du service client.

Ceci est la question de test n°6 du service client sans réponse.  
Ceci est la réponse de la question n°6 du service client.

Ceci est la question de test n°3 du service client.  
Ceci est la réponse de la question n°3 du service client.

Gestion des questions réponses téléphones

Libellé de la question

Réponse de la question

AJOUTER UNE NOUVELLE QUESTION

3

4

Dylan Compière  
Développeur

48



## Modèle conceptuel de données

