



MASTER 2 INFORMATIQUE : GÉNIE LOGICIEL

Rapport de stage Cirrusware

Professeur : Laurent Réveillère
Maitre de stage : Vincent Jaulin

Auteur :
Alexandre Erard

10 août 2021

1 Remerciment

2 Avant propos

Ce rapport s'inscrit dans le cadre du stage de fin d'études d'informatique de l'Université de Bordeaux, plus précisément dans la spécialité Génie Logiciel.

Au cours de mon cursus, j'ai pu m'intéresser à divers pan de l'informatique comme par exemple le développement d'application mobile native, l'intelligence artificielle, la gestion des bases de données et plus largement le Big Data, le développement web. Celui qui m'a le plus passionné est l'architecture logiciel, c'est à dire construire des applications scalable, maintenable tout en restant performante. Pour moi il prend tout son sens dans le développement d'application web qui ont pour obligation d'optimiser leur performances, ainsi que leur consommation de ressource tout en étant capable d'ajouter facilement de nouvelles fonctionnalités afin de s'adapter aux nouveaux besoins.

J'ai donc recherché un stage dans ce domaine et l'entreprise Cirrusware m'ont offert l'opportunité de participer à leur projet afin que je développe mes compétences de développeur full stack.

Table des matières

1	Remerciment	1
2	Avant propos	2
3	Contexte du stage	4
3.1	Cirrusware	4
3.2	La plateforme Send Up	4
4	Outils et technologie utilisées	5
4.1	Outils	5
4.1.1	Monday	5
4.1.2	Google Workspace	5
4.1.3	Git et GitKraken	6
4.1.4	PHPStorm	6
4.1.5	Postman	7
4.2	Technologies	9
4.2.1	Symfony	9
4.2.2	VueJS	9
5	Missions réalisées	12
5.1	Widget pour le studio Digital	12
5.1.1	Serializations des données des formulaires	12
5.1.2	Compréhension de l'API et implémentation des requêtes	13
5.1.3	Création des widgets	13
5.2	Reprise d'un projet XRM	14
5.2.1	Fonctionnalités	14
5.2.2	Analyse de l'existant	14
6	Conclusion	16
7	Anexes	17

3 Contexte du stage

3.1 Cirrusware

Cirrusware est une PME informatique française spécialisée dans le développement d'applications web ayant pour objectif de faciliter les communications numériques au sein des entreprises ainsi que de faciliter les gestions des relations entre tiers et contact.

Fonder en 2013, elle a su convaincre de grands groupes français tels que *MAÏSA-DOUR*, *OCEALIA*, *CARREFOUR*, ou encore *INTERCONTINENTAL - grand hotel de Bordeaux*. (Pour plus d'information, voir)

Aujourd'hui l'entreprise compte 13 employés dont 6 développeur 5 chefs de projet et 2 commerciaux.

Cirrusware développe deux type principaux de produits :

- La plateforme Send-Up [1]

- Le XRM.

3.2 La plateforme Send Up



Send-Up est le premier produit de Cirrusware et permet au client de créer, envoyé, analyser, et automatiser leur campagne de communication. Et cela au travers de divers outils présents sur la plateforme.

4 Outils et technologie utilisées

Lors de mon arrivé dans l'entreprise, j'ai du m'habituer à utiliser leurs outils de, gestion de projet, de communication, mais aussi de developement. Lors de nos projets d'études, là où on était une équipe de maximum 6 et on utilisait principalement des outils tels que Discord pour communiquer, VSCode pour developper, et GitHub Issue ou trello pour gérer le projet. Ici j'ai du m'habituer à utiliser des outils plus professionnel et complet afin de pouvoir travailler sur des projets de plus grande taille. Voici la liste exhaustive de ces outils.

4.1 Outils

4.1.1 Monday

Monday est un outil de gestion de projet simple qui permet de créer des tableaux et y ajouter des tâches. Chaque tâche peut etre attribué à un quelqu'un, possède un état (en cours, terminé, etc.) et une importance. De plus chaque tâche possède un espace "discussion" qui permet de faire état de l'avancement ou encore de noter certaines informations importantes.

	Classement	Demandé le	Demandeur	Dev	Statut	Temps estimé	Terminé le
[Actions] - modale revoir les couleurs des boutons	★★★★★	juil. 26			MA. Fait		
[Drive] - Documents // Contrats et mandats	★★★★★	juil. 26			Clients		
[Timeline d'un client] - on peut voir que des actions liées avec l'utilisateur	★★★★★	juil. 26			1. A faire		
[Modale Email] - L'envoi d'email ne fonctionne pas	★★★★★	juil. 29			5. Push PROD		
[CSS] - Dans les champs de recherche faire apparaitre entièrement le mot "Rechercher"	★★★★★	juil. 26			4. Non validé	0,5h	
[Modale] - Agenda // Champ non obligatoire pour le message	★★★★★	juil. 26			3. Push Pre-PROD	0,5h	
[UDS] - Faire la géolocalisation	★★★★★	juil. 26			3. Push Pre-PROD	1h	
[Clients/Contacts] - sauvegarder la recherche + la page	★★★★★	juil. 26			3. Push Pre-PROD	1h	
[Fichier client] - Bloc drive // fichier n'apparait pas	★★★★★	juil. 26			2. En cours	2h	
[Fichier client] - Bloc drive // Suppression impossible	★★★★★	juil. 26			2. En cours	2h	
[CRM AVAL] - Voir le mode hors connexion	★★★★★	juil. 26			2. En cours	2h	
[Responsive] - A faire V0	★★★★★	juil. 26			1. A faire		
[Doc] - Information sur les droits	★★★★★	juil. 26			1. A faire		
[Modales] - quand on clique sur une action, faire apparaitre directement l'édition et pas l...	★★★★★	juil. 29			1. A faire	0,5h	
[Automation] - Revoir le process pour la géoloc	★★★★★	juil. 26			1. A faire	1h	
[Blason] - Fiche client // Header block paramétrage photo	★★★★★	juil. 26			1. A faire	1h	
[Abonné] - Enlever Groups // Vincent	★★★★★	juil. 26			1. A faire	0,5h	

FIGURE 1 – Exemple de tableau Monday

4.1.2 Google Workspace

Afin de faciliter la communication au sein des équipes, l'entreprise utilise Google Workspace afin de produire de la documentation (Cahier des charges, etc...), Gmail

et Chat afin d'échanger rapidement et Agenda pour planifier les rendez-vous. De plus, l'entreprise ayant des clients se trouvant dans toute la France, et tenant compte de la situation sanitaire actuelle l'applications Google Meet est utilisée comme application de visio conférence, pour les diverses réunions et rendez-vous projets .

4.1.3 Git et GitKraken

Git est un logiciel de gestion de version, il permet de créer des projets et de les versionner. Il permet également le partage du code ainsi que le développement sur plusieurs branches, afin de ne pas empiéter sur le travail des autres membres de l'équipe.

GitKraken lui est une interface graphique utilisant git. En effet git s'utilisant principalement par ligne de commande il est parfois difficile de l'utiliser. GitKraken permet de faciliter l'utilisation de git et permet de tout faire en passant par l'interface (pull, merge, se déplacer entre les branches, etc..).

//TODO Screen GitKraken

De plus il permet une totale intégration des git flow. C'est à dire créer facilement une branche par nouvelle feature, et simplement merge avec la branche de développement lorsque celle ci est terminée. Les git flow permettent également une meilleure gestion des HotFix et des release mais je ne les ai pas utilisé lors de mon stage.



FIGURE 2 – Déroulement d'un git flow

4.1.4 PHPStorm

PHPStorm créé par JETBRAIN et l'un des IDE les plus complet pour développement web. A l'aide de ses nombreux pluggins il permet une totale prise en charge des framework comme Symfony et VueJS. De plus il permet de facilement refactorer du code ou encore de créer de nouveaux snippets afin de faciliter les phases de

développement. Malgré ses nombreux points positifs il est possible qu'il reste un peu dur à prendre en main au début mais par chance au cours de mes études supérieures j'ai été amené à utiliser IntelliJ IDEA, ou encore Android studio qui sont deux IDE de JetBrains qui ont exactement la même structure.

//TODO : Screen PHPStorm

4.1.5 Postman

En ce moment l'entreprise ouvre son API de plus en plus à ces clients afin qu'il puissent eux même intégrer les différents services de la plateforme. Et c'est pour cette raison que l'API doit devenir de plus en plus robuste. Dans cette optique nous utilisons l'application Postman afin de facilement créer nos requêtes ainsi que des test associé. L'outil permet de faire hériter des tests à des sous collections ce qui nous permet de gagner du temps lorsque certains test sont génériques comme par exemple un temps de réponse inférieure à 200ms ou encore un statut de réponse différent de 500.

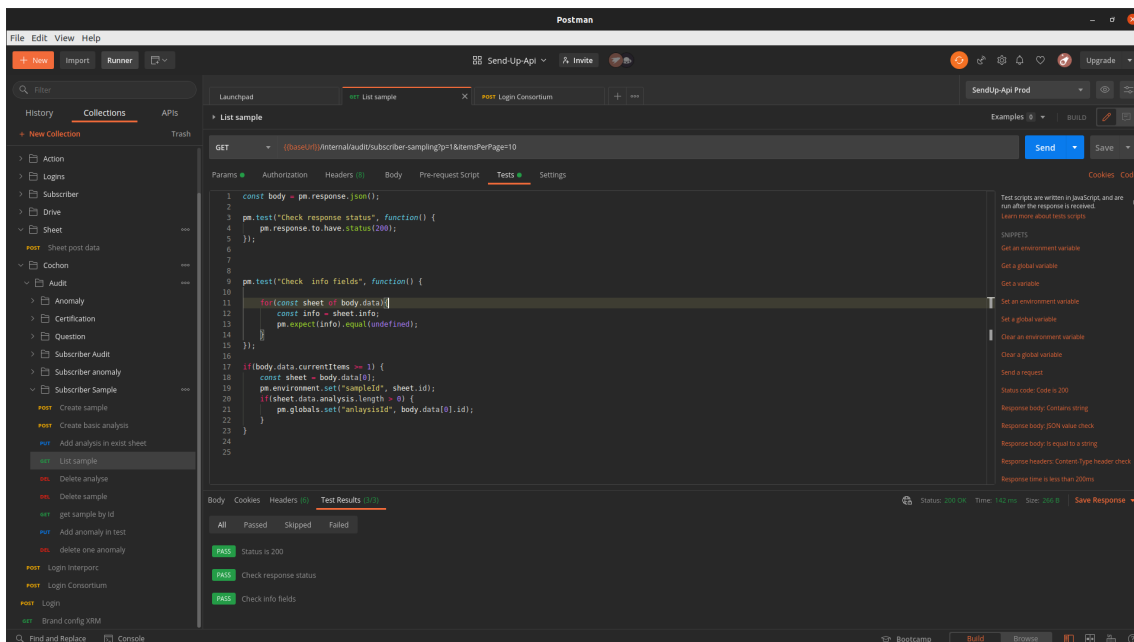


FIGURE 3 – Exemple de test Postman

De plus il intègre un *collection runner* qui permet d'agencer l'ordre des requêtes, ainsi que de les lancer x fois, à un intervalle de temps y .

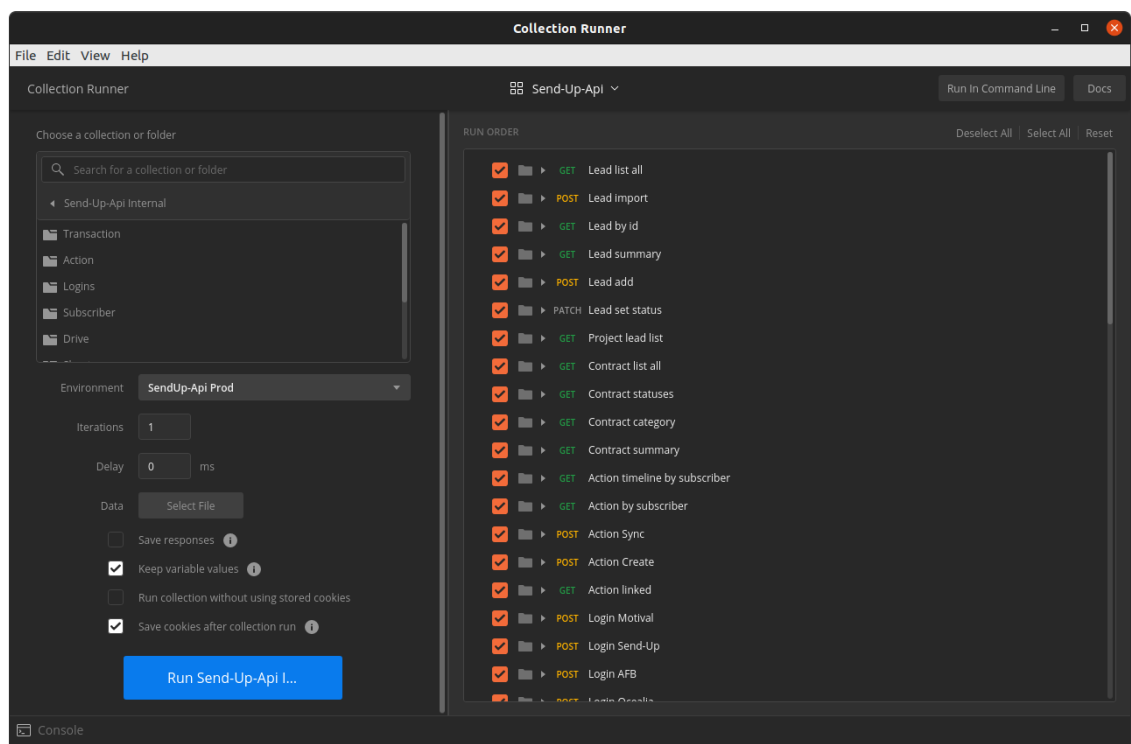


FIGURE 4 – Exemple de collection runner Postman

4.2 Technologies

Historiquement la plateforme Send-Up à été développée sur une stack Symfony 2.8/Jquery/Bootstrap. Aujourd'hui l'entreprise est en pleine transition technologique. Son objectif est de se passer de Jquery et Bootstrap au profit des nouvelles technologies du web et plus particulièrement du framework VueJS. Cette transition reste un défi pour l'entreprise car certains outils du site comme le *studio digital* doivent garder ce relicat afin de garder une rétro-compatibilité mais aussi afin de faciliter son usage auprès des clients. Voyons plus en détails les technologies choisies par l'entreprise à savoir Symfony et VueJS.

4.2.1 Symfony

Symfony [2] est un framework PHP français développée en 2005 par Fabien Potencier. Basé sur le pattern Modèle-Vue-Controller (MVC), il est actuellement en version 5.x.x Le gros avantage de Symfony sur les autres framework php du même type comme Laravel, ou CakePHP est sa documentation qui en plus d'être riches et également traduite en français. De plus, il possède une communauté très importante et réactive. En effet à ce jour il existe plus de X bundle sur la version 2.8. Et énormément de réponses sur les forums de développeur. Il intègre également des outils qui sont extrêmement pratiques dans la gestion des données comme par exemple **Doctrine** [3] un ORM (Object Relation Manager).

Un ORM permet de facilement gérer les données en transformant directement les résultats des requêtes en Entité php. Il intègre également un langage d'interrogation de la base de données nommé DQL. Ce langage rend plus permet de rendre plus lisible les requêtes et l'intégration de variable dans celle-ci. D'autres outils sont également présents comme l'injecteur de dépendance ou encore les outils de débbug disponibles en environnement de développement.

4.2.2 VueJS

Pour remplacer la partie Jquery de Send-Up le framework VueJs [4] a été choisi par l'équipe. Il s'agit d'un framework open-source développé à l'origine par Evan You en 2013. Ce framework facilite la création d'interface graphique et de Single Page Application (SPA). Pour présenter rapidement le framework voici un exemple de composant basique.

Un fichier .vue est composé de 3 éléments principaux.

- **template** : contient le code HTML de l'interface graphique.
- **script** : contient la logique et le comportement du composant.
- **style** : contient les styles CSS du composant. Le style peut être utilisé avec une props *scoped* afin de limiter les styles à l'intérieur du composant.

```
<template>
  <div>
    <h1>
      {{ helloworld }}
    </h1>
    <button @click="count">CLICK ME</button>
  </div>
</template>

<script>
export default {
  name: "HelloWorld",
  components: {},
  mixins: [],
  props: {
    msg: String,
  },
  data() {
    return {
      hello: "Hello",
      count: 0,
    };
  },
  computed: {
    helloworld() {
      return this.hello + " " + this.msg + "!";
    },
  },
  methods: {
    count: function () {
      this.count++;
    },
  },
  mounted() {
    console.log("HelloWorld component mounted");
  },
};
</script>

<style></style>
```

FIGURE 5 – Exemple de composant VueJS

La balise script est l'endroit où on va déclarer le composant et tout ce dont il a besoin pour fonctionner. On y retrouve donc les sous composant qu'il utilise dans son rendu, les paramètres (appelé *props*) que son parent peut lui passer, ici on attend un paramètre *msg* de type *String*. On retrouve également l'état du composant avec toutes les données utilisées, un compteur et la chaîne de caractère *hello*, les méthodes qui vont déterminer la logique, les propriétés calculées du composant. Les propriétés calculées sont appelables comme les variables indiquées dans *data*. Mais ne sont calculées qu'une seule fois à la création du composant puis lors de la modification de variable utilisée à l'intérieur. De plus le framework possède une catégorie d'objet nommé *Mixins* dont on parlera plus en détails plus tard.

Voici le rendu de ce composant appelé avec *msg = Friends* :

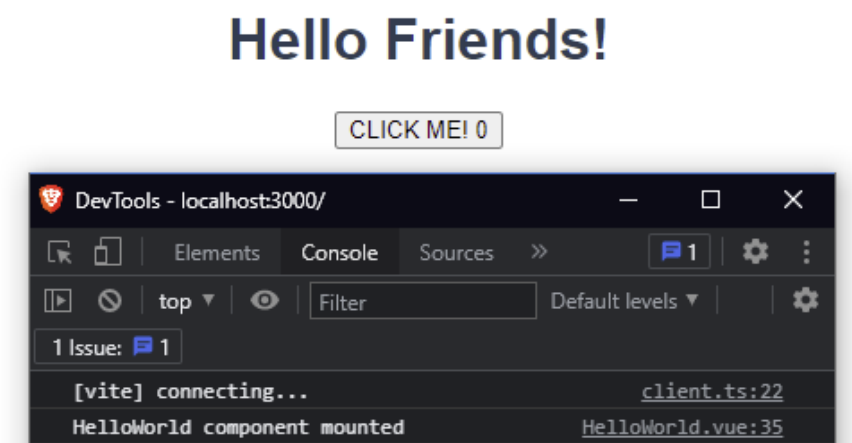


FIGURE 6 – Exemple de composant

On y retrouve également les divers événements du cycle de vie auquel on peut se brancher pour effectuer certains traitements spécifiques. Voici le schéma du cycle de vie d'un composant, pour plus de détails voir l'annexe 7

5 Missions réalisées

Au cours de mon stage j'ai pu intervenir sur différents projets allant d'un simple script permettant de dynamiser une page web d'un client à la reprise d'un projet complet. Ici nous allons détailler les deux principales missions de mon stage à savoir le développement de widget en vueJS pour le studio Digital et la finalisation des XRM du *Consortium du Jambon de Bayonne* et d'*Interporc Franche-Compté*.

5.1 Widget pour le studio Digital

Lors de mon arrivé dans l'entreprise, celle ci cherchait a valorisé ses formation en devenant Organisme de formation certifié **QUALIOPI**. QUALIOPI est une certification qui garanti des formations avec :

- des bons résultats
- des adaptations au besoins
- des formateurs qualifiés
- une bonne prise en compte des retours face au formations

Afin de correspondre à ces critères, nous avons besoin de construire un site web qui permettrait d'effectuer les questionnaires des différents modules de la formations, ainsi que le formulaire de satisfaction. Puis par la suite de pouvoir consulter les différentes statistiques des différents modules sous 3 vue différentes, formés, qui doit voir uniquement ses résultats, entreprise qui ne voit que les statistiques de ses employés formés, une visions globale qui permettra de voir toutes les statistiques des personnes formées.

La première étape réalisé afin de pouvoir réaliser ceci à était d'adapter l'enregistrement des formulaires du studio qui jusque là ne nous permet pas d'effectuer simplement des statistiques.

5.1.1 Serializations des données des formulaires

Afin de récolter les données des formulaire, le studio digital passe par un serializer qui va enregistrer dans un fichier json les valeurs des différents champs associé à leur id. Ce système présente plusieurs avantage comme la facilité et la rapidité d'implémentation. Malheureusement ceci ne nous permet pas de savoir si une réponse est, dans notre cas, correcte ou non. Ou encore pouvoir associé le label d'une question à son résultat.

Pour atteindre le résultat souhaité j'ai du implémenter un nouveaux widget pour le studio digital en VueJS. Ce widget est un simple bouton de correction qui va devoir récupérer tout le contenu nécessaire du formulaire et les stocker dans des champs cachés afin de se servir du serializer. Le reste du formulaire étant coder en

JQuery/html, la seule solution était donc de parcourir le formulaire et de récupérer les informations dans le DOM afin de récupérer les informations nécessaires. À savoir les informations permettant le calcul de la note, diverses méta-données par rapport à la question comme le label, l'id du champ concerné, le résultat obtenu à la question, etc..

Dès que ces informations furent récupérées il ne restait plus qu'à établir les différentes requêtes nécessaires à la récupération des données en fonction du périmètre défini (formé, entreprise, admin)

5.1.2 Compréhension de l'API et implémentation des requêtes

Dans l'actuelle implémentation du serveur Send-Up, la partie gère les *Recordset* - Table MySQL comportant un champ info destiné à stocker du JSON - est gérée par une partie de l'API dont les routes pointant vers cette partie sont générées automatiquement à partir du nom de la méthode défini dans le contrôleur. Exemple la méthode `postRecordset` est appelée lors d'une requête POST sur `/Recordset`. De plus l'une des complications a été de comprendre le format des données à envoyer dans la requête. En cherchant plus profondément dans le code je me suis rendu compte que seuls les **FormData** étaient acceptés pour la requête existante. Une fois le fonctionnement compris il suffisait d'implémenter une requête à la base de données prenant en compte la portée dans laquelle on se trouvait afin de renvoyer uniquement les données nécessaires.

5.1.3 Création des widgets

Une fois les données récupérées et les requêtes créées il fallait que je crée 5 widgets différents qui permettent de :

- visualiser les résultats d'une personne formée pour chaque tentative.
- visualiser les réponses avec correction d'une personne formée.
- afficher les statistiques globales de la formation c'est à dire le nombre de bonnes réponses par rapport au nombre de personnes.
- afficher les statistiques globales pour le formulaire de satisfaction client.
- afficher les réponses d'une personne formée au questionnaire de satisfaction.

Après avoir développé certains d'entre eux, il s'avéra que certains traitements se répètent ou se ressemblent fortement. Or VueJS intègre ce qui appelle les *Mixins*, ces éléments possèdent les mêmes caractéristiques dans leur balise script qu'un composant VueJS classique. Mais ce dernier a la possibilité d'être intégré à l'intérieur d'un composant (ou autre Mixins). Chaque composant hérite donc des données, méthodes et autres propriétés calculées. Ceci empêche donc la redondance et facilite la maintenabilité. C'est donc pour cela, que j'ai créé un mixins spécial afin de faciliter la lisibilité et la maintenance du code.

5.2 Reprise d'un projet XRM

Suite au départ d'un développeur de l'équipe. On m'a placé sur la reprise de son projet qui consiste à créer un XRM permettant tous les points de contrôle concernant le *Consortium du Jambon de Bayonne* et *Interporc Franche-Comté*. Un XRM est un logiciel de gestion des relations tiers. C'est à dire des clients, fournisseur, revendeur etc... En plus de réaliser ceci le notre doit permettre de pouvoir de créer et de réaliser des audits, en fonction du type d'activité et du cahier des charges de la personne auditée, et de pouvoir saisir les données issues d'analyse laboratoire.

Toute la partie audit était réalisée par mon prédécesseur, il me reste donc à implémenter la partie Analyse et prélèvement.

5.2.1 Fonctionnalités

Tout d'abord passons en revue les fonctionnalités souhaitées par les clients. Ici nous avons 2 clients différents avec des besoins différents.

- En tant que Consortium, je souhaite pouvoir créer des fiches prélèvements. En fonction du cahier des charges, type d'activité de la personne auditée il est possible d'avoir plusieurs champs différents définis dans un fichier de configuration.
- En tant que Consortium, je souhaite pouvoir ajouter des analyses à un prélèvement. Chaque analyse possède un type dépendant des champs sélectionnés dans la fiche prélèvement. Ceci conditionnera les champs spécifiques de l'analyse.
- En tant que qu'Interporc, je souhaite pouvoir créer des fiches analyse. Les champs spécifiques dépendent uniquement du type de produit sélectionné : Saucisse, Lisier, Lactoserum, Sel.
- En tant qu'Interporc et Consortium, je souhaite pouvoir lever des écarts à chaque analyse si besoin est.
- En tant qu'Interporc et Consortium, je souhaite pouvoir téléverser les résultats des laboratoires et les associer à une analyse.

5.2.2 Analyse de l'existant

Après avoir discuté avec mon chef de projet, et mon maître de stage, on a déterminé qu'un audit et un prélèvement se ressemblent énormément. En effet chacun est relié à un auditeur/préleveur, une personne auditée, un ou plusieurs cahiers des charges et un ou plusieurs types d'activités.

Pour cela les audits sont stockés dans une table nommée *Recordset_app*. Cette classe possède un champ info au format JSON, comme la classe *Recordset* décrit précédemment. La grande différence est que cette dernière est gérée par une classe abstraite

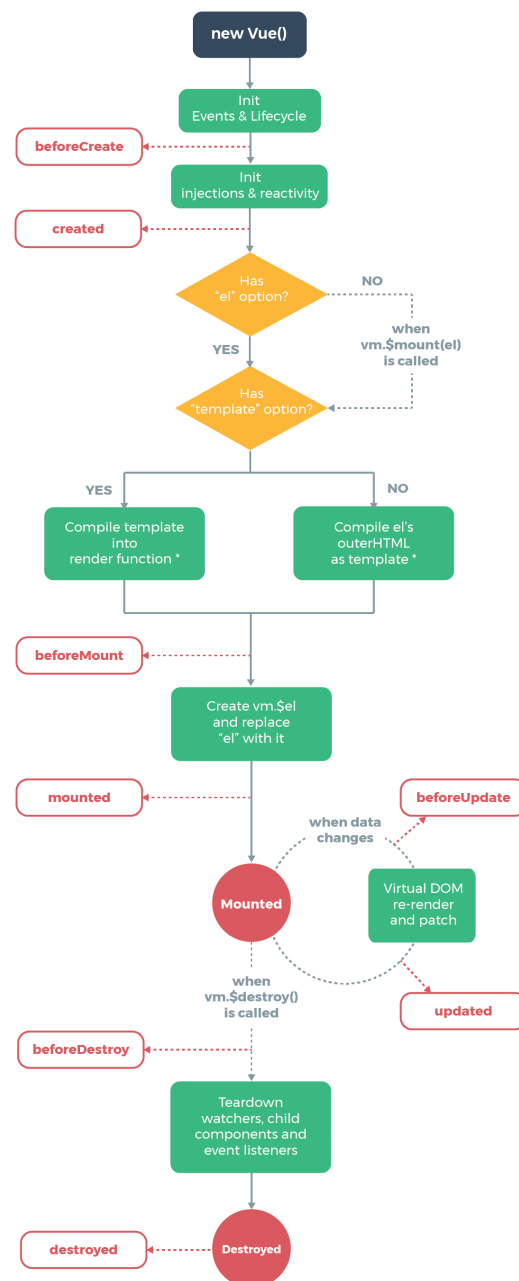
nommé *AbstractSheet*, qui définit que le champs info possède au minimum un champ builder, data et files. Chaque entité voulant être stocker dans la table *Recordset_app* doit être hérité de cette classe et donc respecter ce schéma. Pour différencier nos sous classe nous faisons appel une fonction de doctrine appelée le discriminant. Il permet d'associer un entier a une classe. A ce jour nous avons 2 classe héritant de *AbstractSheet* :

1. *Recordset_app* : qui correspond à toute les fiches standard.
2. *SubscriberAudit* : qui correspond au audit créer par nos subscriber.

Voilà pour l'existant au niveaux de l'api. Voyons maintenant comment à été construit les audit afin de pouvoir garder une cohérence dans l'application et de pouvoir réutiliser au maximum les composants déjà créer.

6 Conclusion

7 Annexes



* template compilation is performed ahead-of-time if using a build step, e.g. single-file components

FIGURE 7 – Schéma du cycle de vie - d'un composant vue [5]

Références

- [1] Cirrusware. Send-up. <http://www.send-up.net/>.
- [2] Symfony. Symfony. <http://symfony.com/doc/current/index.html>.
- [3] Doctrine Project. Doctrine. <https://www.doctrine-project.org/projects/doctrine-orm/en/2.9/index.html>.
- [4] Vue. Vue. <https://vuejs.org/>.
- [5] Vue. The vue instance. <https://vuejs.org/v2/guide/instance.html>.
- [6] Red Hat. What is a rest api? <https://www.redhat.com/fr/topics/api/what-is-a-rest-api>.