**Chapitre 1 : Présentation générale**

**Introduction**

Dans ce chapitre nous procéderons à la présentation de l’organisme d’accueil. Par la suite, nous passerons à la problématique, suivie par la présentation du projet. Enfin, nous conclurons ce chapitre avec le choix méthodologique.

* 1. **Organisme d’accueil**

1. **Présentation de l’agence INSPIRE :**

INSPIRE est une agence créée en 2006 spécialisée dans le développement des sites web pour les différents médias ainsi que la conception des applications mobiles.

Les missions d’INSPIRE consistent à la réalisation, la gestion et l’accompagnement des projets web et mobiles ainsi qu’ils disposent d’un savoir-faire mise la disposition de sa cible.

INSPIRE dispose de 4 sites en Tunisie, en France, au Canda et à Singapour et entre 20 et 100 employés y travaillent.

1. **Organigramme de l’agence INSPIRE**

L’organigramme de la Figure 1 représente la hiérarchie générale d’INSPIRE :

Commerciaux

Président

Directeur de production

**Superviseur**

Directeur des projets

Chef de projet

Chef de projet

Chef de projet

Equipe Digital

Designers

Intégrateurs

Développeurs

Mobile

Développeurs

Web

Figure 2: Organigramme Inspire

* 1. **Problématique**

Quiz burger est un jeu télévisé décalé du XXIe siècle, c’est un jeu pour 2 joueurs ou 2 équipes de joueurs avec plusieurs questions auxquelles il faudra trouver une réponse rigolote, étonnante ou même sérieuse. La première équipe à totaliser 25 miam tentera le Burger de la Mort. Le problème est la révolution numérique dans laquelle nous vivons aujourd’hui est très orientée vers les nouvelles technologies de l’information et de la communication. Désormais, presque tout le monde peut accéder à l’outil internet, aux appareils mobiles et aux ordinateurs ce qui rend les jeux télévisés un peu ancestral par rapport au jeux mobiles, et vidéos, 3D.

* 1. **Présentation du projet**

Comme J’ai déjà expliqué dans la problématique mon projet a pour objectif de concevoir et développer l’application web du jeu quiz burger.

**Le principe du jeu est le suivant composé de deux phases :**

**•** Le premier joueur à cumuler **25 Miams** tente le **Burger** de la mort

**•** Déterminer un joueur qui commence et **attribuez les équipes** (Mayo ou Ketchup).

• Le premier joueur répond aux questions dans l’ordre :

**Phase 1**

1. **Nuggets** (1 Miam par bonne réponse)

- Le MdJ prend la 1ère carte Nuggets et pose la 1ere question, il pose ensuite la seconde question à l’équipe Ketchup et ainsi de suite (ne pas lire les infos entre parenthèses à haute voix).

- Lire les questions recto-verso de la carte. 3 questions par équipe

1. **Sel** **ou** **poivre** (1 Miam par bonne réponse : Epreuve de rapidité)

-Le MdJ prend la 1ère carte Sel ou Poivre du paquet et énonce le titre du questionnaire à tous les joueurs.

-Ne lire qu’une seule face de la carte.

- En cas de mauvaise réponse, les Miams sont attribués à l’autre équipe.

3.**Menus** (1 Miam/bonne réponse)

- Le MdJ prend une carte menu Bleue et une Rouge et propose à la première équipe les trois menus (2 menus recto-verso sur la carte bleue et un sur la rouge).

- Il pose ensuite la question sur le thème choisi.

- Puis il propose à l’autre équipe les deux menus restants et pose les questions du menu choisi.

4.**L’addition** (3 Miams/bonne réponse : Epreuve de rapidité)

- Le MdJ prend la 1ère carte Addition et énonce le titre du questionnaire à tous.

- Epreuve de rapidité, tous les joueurs participent en même temps.

- En cas de mauvaise réponse, les Miams sont attribués à l’autre équipe.

La première équipe a totalisé **25 Miams** est déclarée gagnante et tente le **Burger de la Mort.**

Si vous n’avez pas atteint le nombre de Miam, recommencez une partie jusqu’à réaliser **25 Miams**.

**Phase 2**

**4.Le Burger de la Mort**

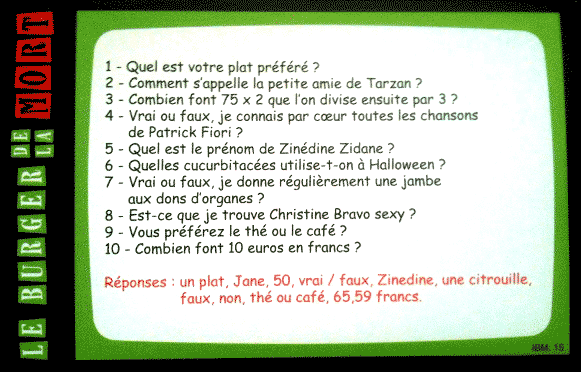
– L’équipe gagnante désigne son candidat pour tenter l’épreuve du Burger de la Mort.

Figure 3- Exemple des questions du jeux

•Le MdJ prend une carte **Burger de la Mort** et pose **10 questions** au joueur gagnant qui doit attendre la fin du questionnaire pour y répondre dans l’ordre.

•Il ne doit pas prendre de note, ni faire répéter les questions.

•Si le joueur réussit à donner 5 bonne réponses dans l’ordre, il gagne **le petit Burger de la Mort**

•Si le Joueur réussit à donner 10 bonnes réponses, il gagne **le Grand Burger de la Mort.**

Les objectif principaux visé par cette application Web est de développer la partie administrateur qui permet de :

* **Gérer** les joueurs
* **Paramétrer** le jeu
* **Remplir** la banque de question avec les réponses adéquates
* **Créer** les différentes phases ou niveaux (Levels) du jeu avec le design adéquat

Ainsi que le développement de la partie joueur qui permet de :

* **Consulter et accéder** au jeu
* **Gérer** son profil

Sans oublier la phase **d’authentification** pour les joueurs ainsi que pour l’administrateur.

* 1. **Processus de développement**

La gestion de projet est au cœur de la naissance des applications web. En effet, les méthodes appliquées vont participer directement au succès du projet.

Il existe plusieurs méthodes de développement logiciel construites sur UML comme la méthode : Unified Process (UP), Rational Unified Process, UP agile, Two Tracks Unified Process, eXtream Programming etc. Parmi ces méthodes notre choix est porté sur la méthode agile scrum.

**1.5.1 Justification du choix méthodologique**

Choisir la bonne méthodologie de gestion de projet est une étape indispensable pour réussir notre projet. Cette méthodologie doit valoriser les objectifs de notre projet et les points forts de l’équipe, aussi de finaliser le projet dans les délais de livraison.

**1.5.2 Démarche**

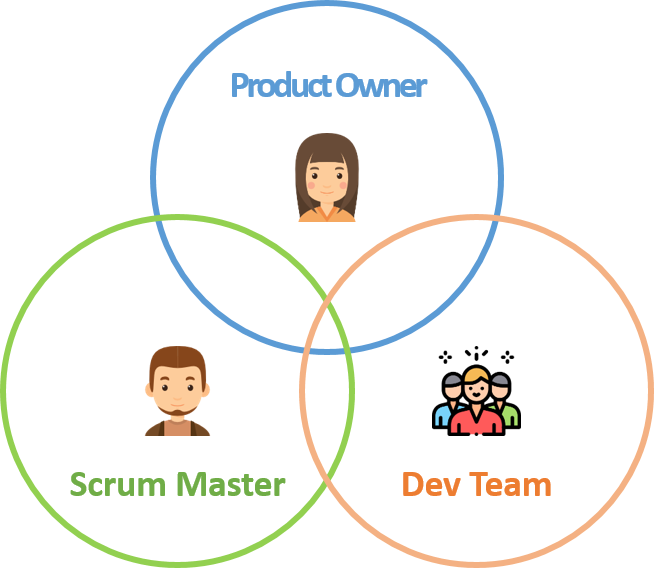
Le processus que nous avons adopté pour le développement de notre application est basé agile scrum. Il s’agit d’une méthodologie de gestion de projet, Scrum décrit un ensemble de réunions, d'outils et de rôles qui interagissent de concert pour aider les équipes à structurer leur travail et à le gérer. Nous avons utilisé « Jira Software » en tant qu’outil scrum afin d’organiser notre projet et le découper en sprint, prioriser et enfin valider les tâches.

Une image contenant texte, capture d’écran, portable, ordinateur

Description générée automatiquementLa figure se de sous montre clairement les différents projets de la société INSPIRE en cours de réalisation, y compris notre projet QUIZ Burger, en se basant sur la méthodologie Scrum à l’aide de l’outil « Jira Software ».

* 1. **Adaptation de la méthodologie SCRUM dans le projet**

La méthode de gestion SCRUM se base sur les développements itératifs à un rythme constant qui varie entre 2-4 semaines « sprints ». Parmi les avantages d’adaptation de la méthodologie Agile avec la méthode SCRUM, la diminution de la documentation au minimum pour gagner en productivité pour nous permettre la sauvegarde de l'historique des décisions prises sur le projet du client et d'effectuer facilement dans le futur des interventions sur la solution logicielle surtout que Scrum est qualifié comme simple, pragmatique et transparent.

**La méthode Scrum se compose de 3 principaux acteurs :**

* **Le Product Owner :** il est considéré comme le responsable du Product Backlog, son rôle est de définir les spécifications fonctionnelles, établit la priorité des fonctionnalités à développer ou corriger, coordonne l’implication des utilisateurs et des parties prenantes, ce qui le permet de considérer comme l’interface entre l’utilisateur, le Scrum Master et les équipes chargées du développement.
* **Le Scrum Master :** membre de l’équipe qui s’assure que les principes et les valeurs de Scrum sont respectés pour garantir la constante amélioration des performances et donc la réussite d'un projet agile. Il est considéré comme le garant de la méthode Scrum.
* **L’équipe opérationnelle :** c’est l’équipe Scrum qui va mettre en œuvre les solutions techniques, réaliser les développements. Tous les membres de l’équipe apportent leur savoir-faire pour accomplir les tâches. Elle doit travailler de façon incrémentale et livrer une partie du produit utilisable et testable à la fin de chaque sprint ou itération.

**On ne peut pas passer par la méthode Scrum sans connaître les termes suivants :**

* **Le produit backlog :** Un backlog produit contient les éléments qu’un client veut que l'équipe réalise sous forme d’une liste. Cette liste est composée par plusieurs fonctionnalités hiérarchisées à atteindre sous forme des brèves descriptions.

Le classement de ces fonctionnalités est fait selon la valeur ajoutée que cet élément apporte donc par ordre de priorité de réalisation.

Alors nous pouvons dire, que le Produit. Backlog Scrum est autorisé à mesurer, croître.et à évaluer que nous en apprenons plus sur le produit, ainsi que ses, clients.

**Un backlog Scrum typique comprend les différents types d’éléments suivants :**

• Fonctionnalités

• Bogues

• Travail, technique

• Acquisition, de connaissances

De loin, le principal moyen utilisé par une équipe Scrum, pour exprimer les fonctionnalités, du backlog, de produit consiste, en des user stories, des descriptions, courtes, et simples de la fonctionnalité, souhaitée, présentées du point de, vue de l'utilisateur.

**Le Backlog, du produit comprend les champs suivants :**

• **IDF** : un identifiant, de fonctionnalité.

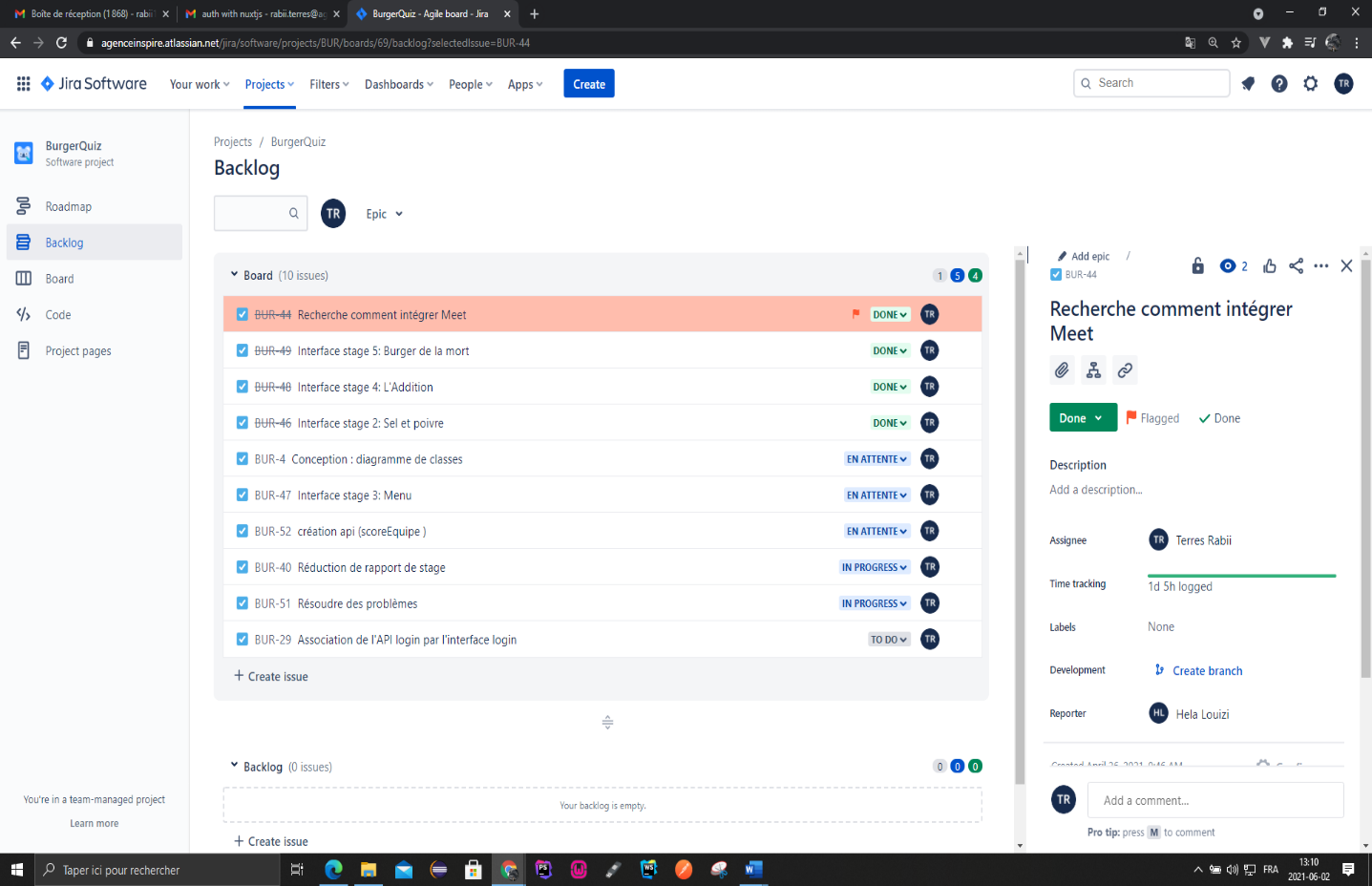
• **Fonctionnalité** : un ensemble de, user stories et qui construit, un service, observable de l’extérieur.

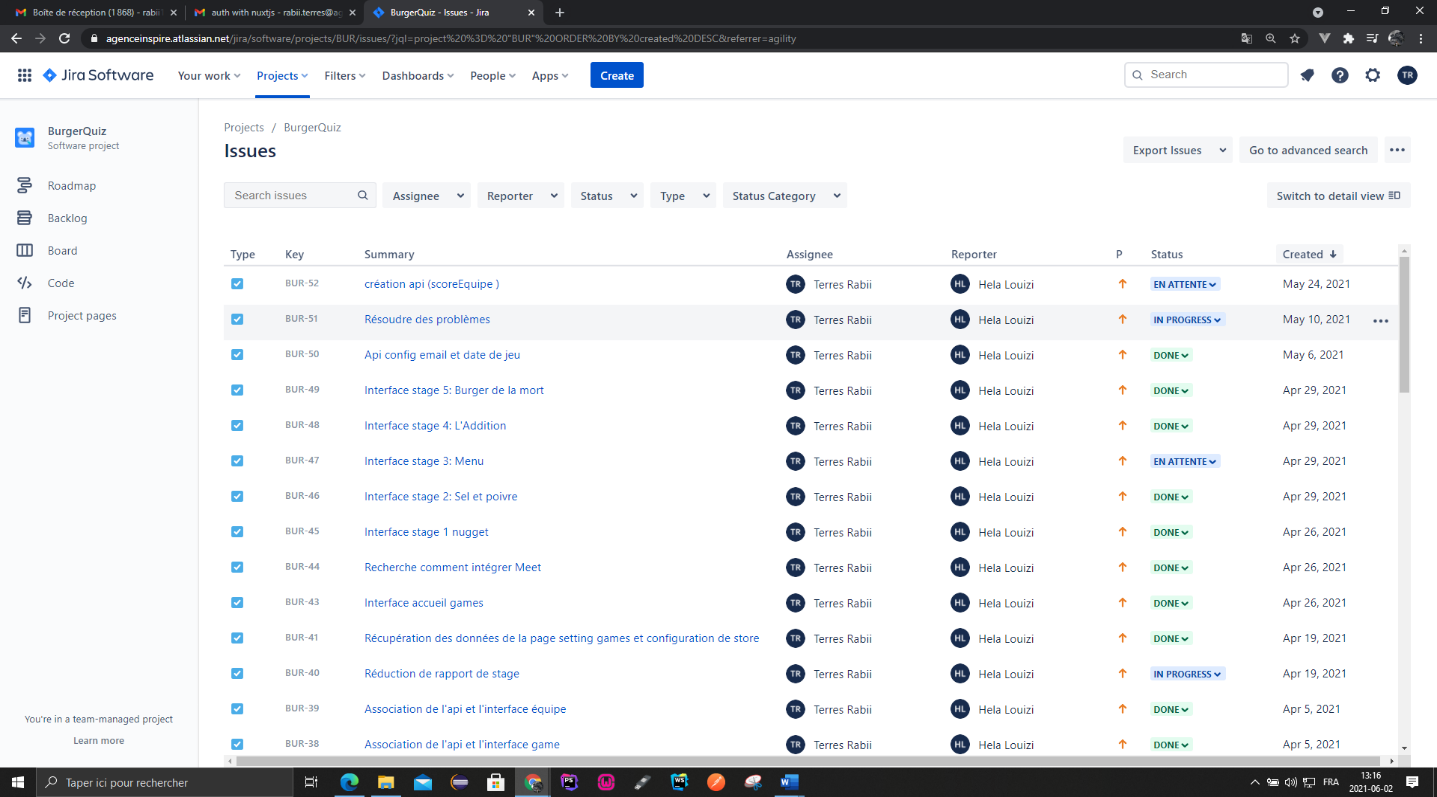
• **Histoire Utilisateur** « **User story** » : une phrase, décrivant la fonctionnalité désirée par le client.

• **ID-US** : un nombre unique, et auto-incrémenté, pour, chaque histoire utilisateur.

• **Business Value** (B.V) : la valeur, métier qui dirige la priorisation du développement des, histoires utilisateurs suivant les, attentes, et les besoins du client, allant de 0, à 100 (étant 100, le plus important).

• **Story point** (S.P) : Les Story Points, permettent de mesurer une tâche, en termes d’effort d’une équipe, concrète, au lieu de simplement, faire des estimations en jour-hommes, qui fluctuent selon les personnes. On utilise souvent, et on utilisera ici, la suite, de Fibonacci (1, 2, 3, 5, 8, 13, 21…) pour, décliner la quantité d’effort, à fournir.



Exemple backlog

Exemple liste tache Jira software

* **Le sprint backlog** : c’est un regroupement des éléments d’un document que l’équipe s’engage à livrer du début à la fin du sprint et qui sont présenter sous forme d’un tableau kanban (ou Scrum board).
* **User story :** c’est une brève description simple d'une fonctionnalité décrite généralement par un utilisateur ou un client du système.
* **La mêlée (Scrum) :** C’est une réunion d’avancement quotidienne durant le sprint pour faire un point sur ce qui a été fait et ce qu'il est prévu de faire.

**1.6 .1 Les différents intervenants**

Durant un développement d’un projet avec la méthode Scrum il y a une interaction avec plusieurs intervenants se présente comme le montre la figure ci-dessous

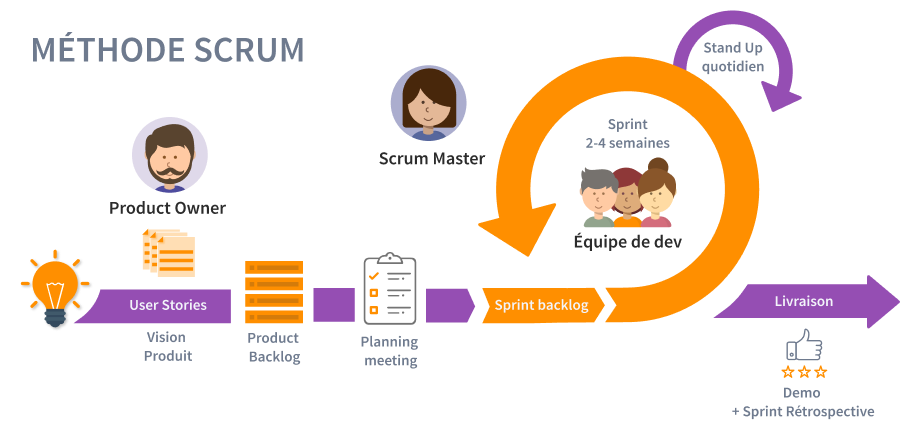


Figure 5 – les intervenants de la méthodologie Scrum

# **Le tableau 1 ci-dessous montre** **la présentation de l’équipe**

|  |  |  |
| --- | --- | --- |
| Rôle | Mission | Acteur |
| Propriétaire de produit (Product Owner) | •Proposées besoins fonctionnels à développer et valide le produit final à livrer | Inspire |
| Scrum Master | •Assure le bon déroulement du projet.  •Encourage l’équipe.  •Il est le relai entre le Product owner et l’équipe de développement. | Mehdi Louizi,  Hela Louizi |
| Équipe de développement | •Conception de l’architecture de la solution  • Mise en place des entités de la base de données  • Implémentation des Services Web  • Implémentation des interfaces utilisateurs  • Consommation et intégration des Service Web. | Rabii Terres |
| • Responsable technique  • Ingénieur système  •Lève les points bloquants | Majdi Allagui |

Tableau 1: Équipe du projet

**1.6 .2 Les sprints**

|  |  |  |
| --- | --- | --- |
| **Sprint** | **Description** | **Durée** |
| **Sprint 0 :** Analyse et  Spécification des besoins. | -Produire la version 0 du backlog du produit.  - Définir l’architecture globale de la solution et réalisation des diagrammes. - Création de l’environnement de travail.  - Choix et intégration du Template. | **4 semaines** |
| **Sprint 1 :** Gestion Partie administrateur. |  |  |
| **Sprint 2 :** Gestion Partie jeu. |  |  |

**Conclusion**

Dans ce premier chapitre nous avons mis le sujet dans son cadre général. Nous allons commencer un deuxième chapitre intitulé « Capture et spécification des besoins » dans lequel nous allons spécifier les exigences, identifier les acteurs et les cas d’utilisation à partir de ces dernières et finir par une description détaillée de quelques exigences majeures.

**Chapitre2 : Analyse et spécification des besoins**

**Introduction**

Dans ce second chapitre, nous exposerons dans un premier temps les exigences fonctionnelles de notre application, à savoir les besoins des utilisateurs accompagnés par des représentations graphiques (maquettes IHM) ainsi que les exigences non fonctionnelles. Dans un second temps, nous allons détailler la branche supérieure concernant la modélisation des cas d’utilisation après avoir identifié́ les acteurs.

**2.1. Vision du projet**

L’objectif fondamental de notre application est de migrer le jeux populaire QUIZ Burger de sa version télévisé vers une version Web afin du jouer à distance. Dans cette partie nous nous focaliserons sur les fonctionnalités de base de notre application.

**2.1.1. Exigences fonctionnelles**

L’application devra regrouper toutes les fonctionnalités nécessaires du jeu pour les deux parties à savoir la partie administrateur qui va paramétrer et préparer le jeu et la partie joueur c’est l’interface du jeu qui doit apparaitre aux joueurs.

Ci-dessous les interfaces du prototype générale « Löw fidelity wireframe » de l’administrateur et celui du joueur et nous allons détailler les autres interfaces dans les chapitres suivants.

Wireframe 1

**Figure n – Prototype de l’interface du tableau du bord de l’Administrateur**

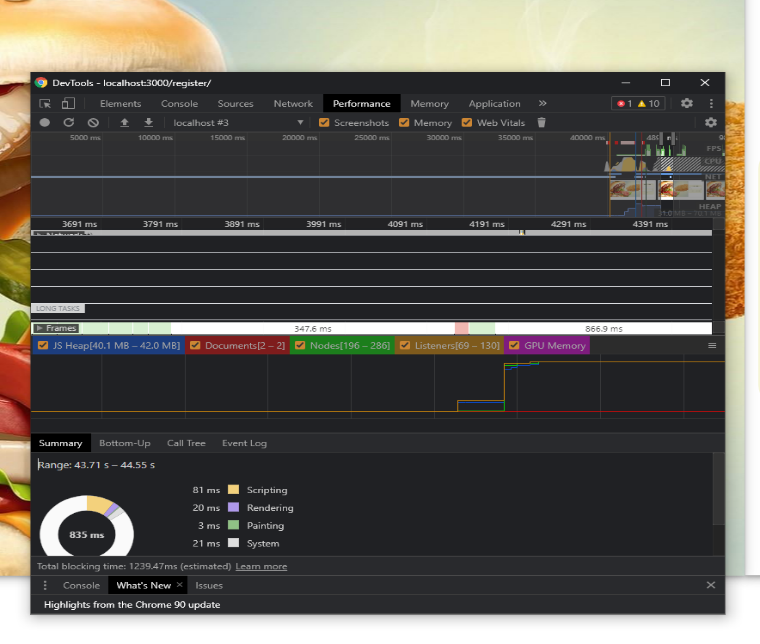
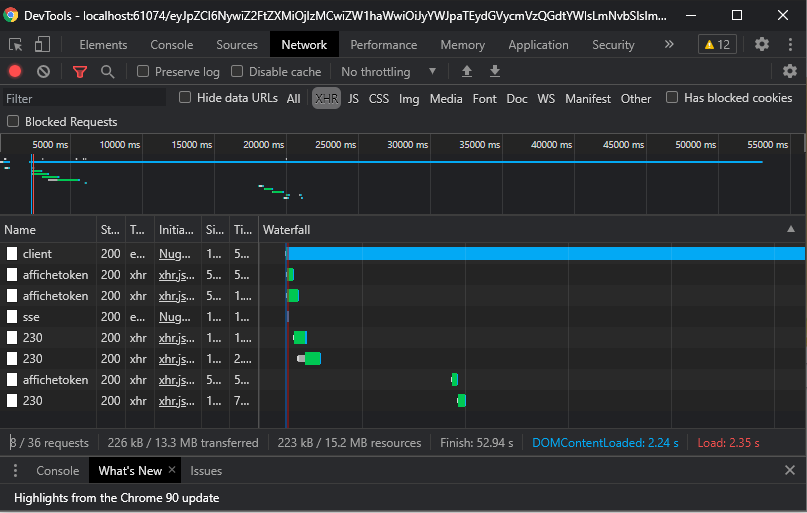
Wireframe 2

**Figure n – Prototype de l’interface du tableau du bord du joueur**

**2.1.2. Exigences non fonctionnelles :**

Après avoir déterminé les besoins fonctionnels, nous présentons ci-dessous l’ensemble des besoins non fonctionnels qui représentent des exigences qui ne concernent pas spécifiquement le comportement du système, mais plutôt identifient des contraintes internes et externes du système. Les principaux besoins non fonctionnels de notre application se résument dans les points suivants :

* **Performance** **:** L'application doit optimiser les traitements pour avoir un temps de génération de schéma raisonnable. La lenteur d’une application web pourrait avoir un impact négatif sur l’UX (ou User Expérience). C’est pour cela que notre solution doit être avant tout performante pour répondre à toutes les exigences des usagers d'une manière rapide et efficace. Ici revient le choix technique de l’utilisation du Framework Vue Js que l’un de ces avantages est la réalisation des interfaces de type mono-page qui fonctionnent sans rechargement de la page web. En effet sur un mono-page après le chargement initial de la page, plus aucun code HTML n'est envoyé sur le réseau. Au lieu de cela, seules les données sont demandées au serveur, donc moins de temps et de bande passante que l'envoi constant de HTML.

La figure suivante prouve la rapidité de réponse du serveur (2.35 secondes) pour les statistiques :

Exemple : rapidité de réponse du serveur

* **Ergonomie** **:** En parallèle à l’UX, il y a l’UI (ou User Interface) qui est le lien entre l’humain et la machine. Donc la plateforme doit offrir une interface conviviale et ergonomique exploitable par l'utilisateur en envisageant toutes les interactions possibles à l'écran du support tenu.
* **Sécurité́ :** *Chaque* joueur ainsi que l’administrateur, pour accéder l’application, est obligé de s’identifier par un nom d’utilisateur et un mot de passe. Il ne pourra accéder qu’au pages qui lui sont permises par son profil ou les droits d’accès. Besoins de mot de passe, longueur, caractères spéciaux, expiration, politique de réutilisation, Donc pour garantir cette sécurité, nous avons utilisé les « JSON Web Token » ou JWT qui sont des jetons générés par un serveur lors de l’authentification d’un utilisateur sur une application Web, et qui sont ensuite transmis au client.
* **Efficacité :** La plateforme doit être fonctionnelle indépendamment de tous les évènements pouvant entourer l'utilisateur.

**2.2. Spécification des exigences**

Comme nous l’avons précisé dans le premier chapitre, notre application web, consiste à développer l’application web du jeu quiz burger.

Après avoir annoncé l’objectif de notre application, nous allons exposer les attentes de l’entreprise en spécifiant les besoins fonctionnels par une fiche de description textuelle.

**2.2.1. Identification des acteurs**

Après avoir identifié́ les exigences fonctionnelles de notre application, on peut déduire quels sont les acteurs qui vont interagir directement avec notre système. Ces acteurs vont accéder à l’application avec des privilèges bien définis.

**Les acteurs pour notre application sont les suivants** :

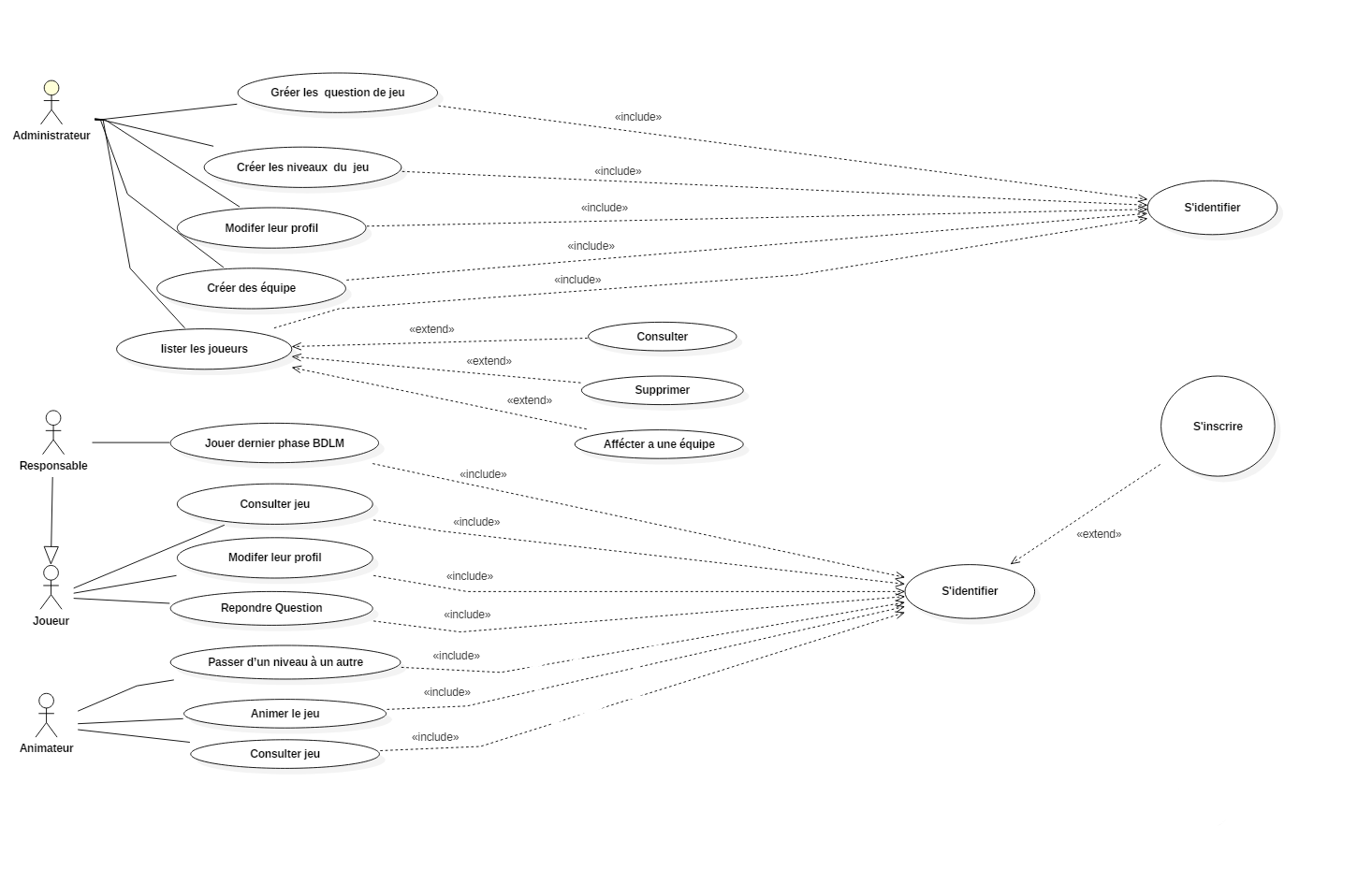
* **Administrateur :** c’est le fondé de pouvoir de l’application, c’estcelui qui a la responsabilité de paramétrer le jeu tel que la création des niveaux de jeu classés par catégories, créer les question quiz appropriés, classés par thème avec leurs réponses adéquates. Il a aussi la tâche de créer les équipes et affecter les joueurs à ces derniers.
* **Joueur**: qui peut être **un simple joueur** ou **Responsable** d’équipe de joueurs
* **Animateur :** est un utilisateur qui anime le jeu une fois paramétré par l’administrateur, il a donc le rôle de valider les questions dans une phase précise et permet de passer d’un niveau à un autre.

**Le tableau 2 présente le Backlog de notre produit :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **IDF** | **Fonctionnalité** | **ID-US** | **User Story** | **B. V** | **S. P** |
| **F1** | **Gérer L’authentification** | **1.1** | En tant qu’administrateur, je souhaite m’authentifier à la plateforme via un courriel et un mot de passe. | **100** | **8** |
| **1.2** | En tant qu’un Joueur, je souhaite m’authentifier à la plateforme via un courriel et un mot de passe. |
| **F2** | **Gérer Utilisateurs** | **2.1** | En tant qu’administrateur, je souhaite lister, modifier rôle, afficher les détails et supprimer un utilisateur. | **100** | **21** |
| **2.2** | En tant qu’administrateur je souhaite consulter et modifier mes informations personnelles. |
| **2.3** | En tant qu’un Joueur ou Responsable ou Animateur je souhaite consulter et modifier mes informations personnelles. |
| **F3** | **Gérer Catégorie** | **3.1** | En tant qu’administrateur, je souhaite ajouter une catégorie  -Si le nombre des catégories inferieur **5**. | **60** | **3** |
| **3.2** | En tant qu’administrateur, je souhaite lister les catégories. |
| **3.3** | En tant qu’administrateur, je souhaite modifier une catégorie. |
| **3.4** | En tant qu’administrateur, je souhaite afficher les détails **Thème** appartient à une catégorie. |
| **3.5** | En tant qu’administrateur, je souhaite de supprimer une catégorie. |
| **F4** | **Gérer Thème** | **4.1** | En tant qu’administrateur, je souhaite ajouter un thème | **60** | **3** |
| **4.2** | En tant qu’administrateur, je souhaite lister les thèmes. |
| **4.3** | En tant qu’administrateur, je souhaite modifier un thème. |
| **4.4** | En tant qu’administrateur, je souhaite afficher les détails **Question** appartient à un thème. |
| **4.5** | En tant qu’administrateur, je souhaite de supprimer un thème. |
| **F5** | **Gérer Question** | **5.1** | En tant qu’administrateur, je souhaite ajouter une question | **40** | **3** |
| **5.2** | En tant qu’administrateur, je souhaite lister les questions. |
| **5.3** | En tant qu’administrateur, je souhaite modifier une question. |
| **5.4** | En tant qu’administrateur, je souhaite afficher les détails **Réponses** appartient à une question. |
| **5.5** | En tant qu’administrateur, je souhaite de supprimer une question. |
| **F6** | **Gérer Réponse** | **6.1** | En tant qu’administrateur, je souhaite ajouter une réponse | **40** | **5** |
| **6.2** | En tant qu’administrateur, je souhaite lister les réponses. |
| **6.3** | En tant qu’administrateur, je souhaite modifier une réponse |
| **6.4** | En tant qu’administrateur, je souhaite de supprimer une réponse. |
| **F7** | **Gérer les phases de jeu** | **7.1** | En tant qu’administrateur, je souhaite d’ajouter 2 équipes. | **40** | **3** |
| **7.2** | En tant qu’administrateur, je souhaite d’affecter un animateur de jeu. |
| **7.3** | En tant qu’administrateur, je souhaite d’affecter les questions des niveaux de jeu |
| **F8** | **Participer ou jeu** | **8.1** | En tant que joueur je souhaite participer dans meet call. | **60** | **8** |
| **8.2** | En tant que joueur je souhaite  Jouer tous les niveaux de jeu.  -Seulement la dernière phase. |
| **8.3** | En tant que responsable je souhaite de jouer toutes les phases de jeu.  - Si mon équipe gagne |
| **8.4** | En tant qu’animateur je souhaite consulter toutes les phases de jeu et de passer les questions. |

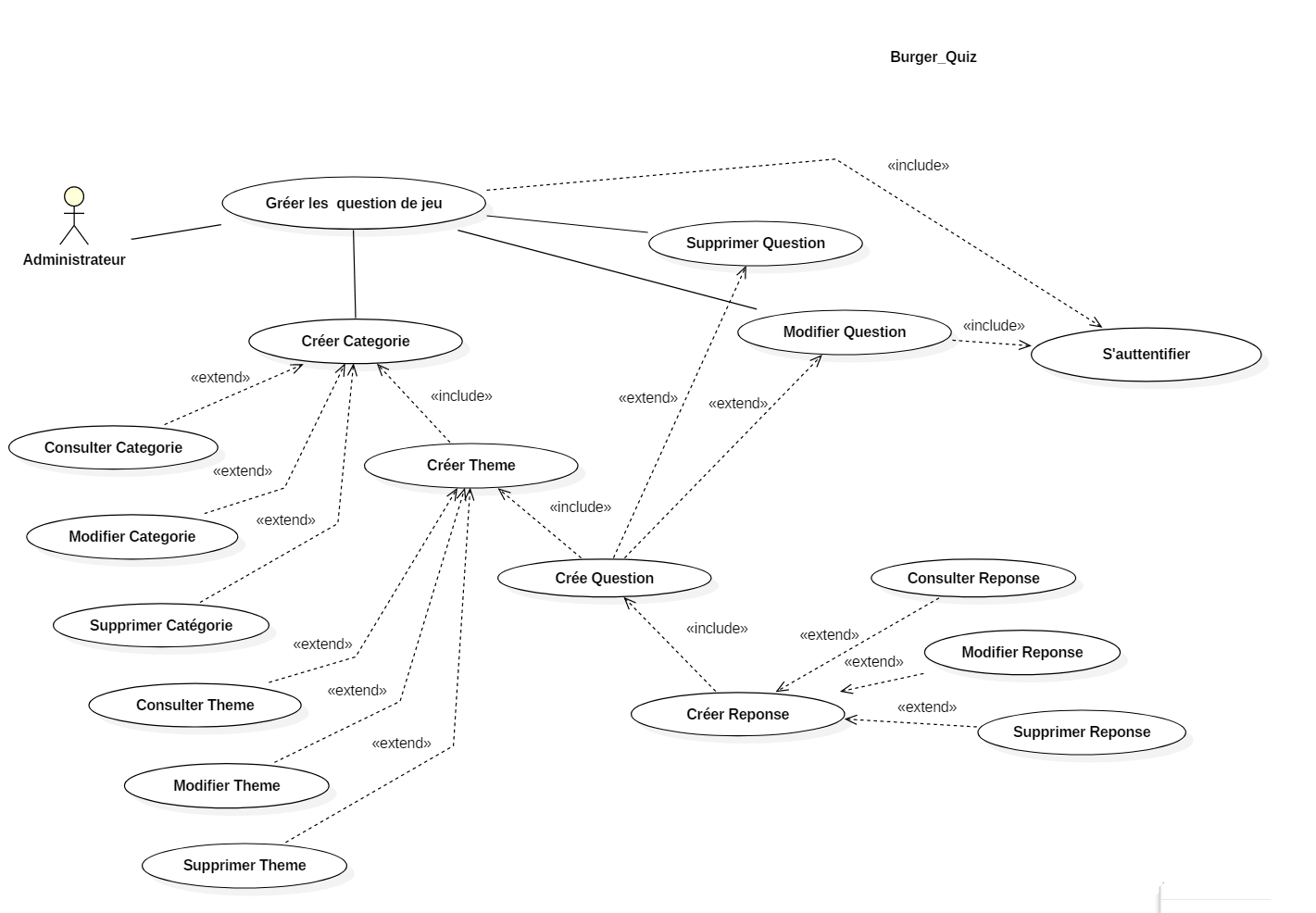
* + 1. **Identification des cas d’utilisation**

Les cas d’utilisation permettent d’exprimer les besoins des utilisateurs d’un système. Le diagramme des cas d’utilisation permet donc d’identifier les possibilités d’interaction entre le système et les acteurs. L’use case, qui présente l’ensemble des fonctionnalités offerte par l’application pour nos utilisateurs (Administrateur, Responsable, Joueur, Animateur), est donné par la figure suivante :



2.3. Spécification détaillée des exigences

2.3.1. Description textuelle des cas majeurs



2.3.2. Diagramme de séquence système des cas majeurs

**Conclusion**

**Chapitre 3 : Réalisation**

**Introduction**

Dans ce chapitre, pour commencer nous procèderons par la présentation de

L’environnement de travail. Ensuite nous présenterons l’architecture physique ainsi que

L’architecture logique et nous finirons par un jeu d’essai.

**3.1 Environnement de travail**

Dans cette partie nous allons présenter l’environnement logiciel, les Frameworks et les langages utilisés lors du développement de notre application web.

**3.1.1. Environnement logiciel**

Dans le cadre de préparation à ce projet, il nous a fallu maîtriser les outils auxquels on aura recours dans la phase de réalisation, à savoir.



**Jira** **Software** est un outil de gestion de projet agile qui prend en charge toute méthodologie agile, qu'il s'agisse d'un Scrum, d'un Kanban ou de votre propre création. Grâce à un seul outil, vous pouvez planifier, suivre et gérer tous vos projets de développement logiciel agile, des tableaux aux rapports agile. JIRA permet de suivre tout type d’unité de travail (qu’il s’agisse d’un problème, d’un bug, d’une user story, etc.) grâce à un workflow prédéfini [5]. Pour plus de détails sur le fonctionnement d’un workflow JIRA (voir annexe).

Ayant choisi d’adopter une méthodologie agile au cœur de notre projet, le fait d’élaborer une liste des tâches à effectuer sous forme de tickets bien détaillés, de décomposer le projet en itération, le suivi de l’avancement du travail ainsi de suite sur papier s’avère peu pratique surtout quand il est question d’effectuer des modifications, c’est aussi contre-productif et signe d’un travail mal ordonné. Du coup il nous a été primordial d’utiliser un tel outil ce qui nous a permis de rendre accessible en lecture et en écriture une seule liste des tâches et un seul workflow pour l’équipe entière où chaque membre est responsable du cycle de vie de la tâche bien décrite qui lui a été affectée.



**Bitbucket** est une solution d’hébergement des dépôts Git conçue pour les équipes professionnelles par Atlassian. Il vous donne un emplacement central pour gérer les dépôts git, collaborer sur votre code source et vous guider à travers le flux de développement.

Tout au long du développement de notre projet, notre base de code a dû subir des mises à jour quotidiennes de la part de tous les membres de l’équipe, il était donc essentiel d’avoir la possibilité d’annuler les modifications et de rendre le référentiel accessible à toute l’équipe. Du coup, l’utilisation d’un outil de versionnage était primordiale. Ensuite, nous avions le besoin de regrouper l’ensemble du code développé en local vers une base de code unique, hébergée quelque part et accessible à toute l’équipe en lecture et en écriture, nous avons donc eu le besoin d’utiliser un service d’hébergement de versions Git. Pour finir, vu qu’on a utilisé JIRA en tant qu’outil de gestion de projet agile et que Bitbucket dispose d’une intégration Jira puissante pour simplifier le suivi des tâches, notre choix s’était donc porté sur ce dernier.



**PHP** Storm est un IDE commerciale pour PHP vraiment très puissant. Par exemple, il ne fait que comprendre la syntaxe colorée d'un PHP, il comprend également Drupal, Symfony 2 et Zend. Il offre la possibilité d'effectuer une autocomplétions du code des fonctions PHP, des méthodes, des noms de variables. Il a une capacité d'analyse du code et retourne des avertissements pertinents. De plus, il supporte le PHP DOC, afin que vous puissiez écrire des commentaires et les exploités. Ainsi, il s'intègre avec les gestionnaires de versions comme Subversion, Git, Mercurial, Preforce, CSV et Team Fondation. Enfin, il supporte le terminal SSH pour les besoins plus particuliers. Il est uniquement disponible en version d'essai de 30 jours et il n'y a pas de version gratuite ou communautaires. Il est écrit en langage de programmation Java à l'aide du logiciel pour Java du nom de IntelliJ IDEA.

****

**Web Storm** est un IDE pour les langages Web (HTML, CSS et JavaScript), développé par l'entreprise JetBrains et basé sur la plateforme IntelliJ IDEA.

**Postman** : est un logiciel qui se focalise sur les tests des API. Il est devenu très populaire pour tester les Micro-services, notamment grâce à sa simplicité et ses fonctionnalités très spécialisées.



**StarUML** est un logiciel de modélisation UML, qui a été cédé comme open source par son éditeur, à la fin de son exploitation commerciale (qui visiblement continue ...), sous une licence modifiée de GNU GPL.



**MySQL** est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde.



**Balsamiq Mockups** : est une maquette d'interface utilisateur graphique et une application de création de site filaire. Il permet au concepteur d'organiser des widgets prédéfinis à l'aide d'un éditeur WYSIWYG par glisser-déposé

**3.1.2. Environnement matériel**

**Machine de développement :**

Ordinateur Portable **HP Notebook 15** PC qui dispose de la configuration suivante :

* Processeur Intel Core i5-7200U 3.04GHZ
* RAM : 8 GO
* Système : Windows 10 professionnel 64 bit

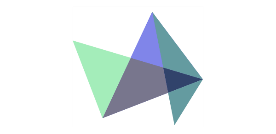
**3.1.3. Frameworks**

**Symfony** est un Framework PHP, un ensemble d’outils et de composants logiciels cohérents qui sert à créer la base de structures de développement, mais aussi le squelette d’une partie ou de la totalité d’un logiciel [3]. Dans notre application Symfony joue le rôle d’un fournisseur de service REST.



**Vue** est un Framework Javascript qui va vous permettre de développer des “Single Page Applications”, c’est à dire des applications qui sont faites avec du responsive javascript, des applications qui peuvent fonctionner sans recharger la page à chaque interaction (principalement avec des appels AJAX) qui permettent à l’utilisateur d’avoir une expérience utilisateur très agréable et très moderne. Dans notre système vuejs est la partie client qui consomme le web service fournit par Symfony.

**VuetifyJs** : est un Framework CSS crée avec vuejs qui dispose une belle collection de composants vuejs prêts à être utilisés (liste déroulante, checkbox…). Et nous facilite la création de nos interfaces.



**Highcharts** : est une bibliothèque JavaScript qui nous permet de créer des graphiques interactifs de nature dynamique ou statique. Cette librairie possède des caractéristiques qui font d’elle un outil indispensable pour la création de graphiques. Highcharts est simple d’utilisation, compatible tous navigateurs et responsive. C’est un outil modulable et interactif proposant différents types de graphiques basés sur une structure en HTML5.

**3.1.4. Langages**

**PHP** est un langage impératif orienté objet. PHP a permis de créer un grand nombre de sites web célèbres, comme Facebook et Wikipédia Il est considéré comme une des bases de la création de sites web dits dynamiques mais également des applications web.



**HTML** est le langage de balisage conçu pour représenter les pages web. C’est un langage permettant d’écrire de l’hypertexte, d’où son nom.

**Vue js** utilise la notion de composant et chaque composant possède une interface graphique sous forme d’une Template HTML, ce-si donc justifie notre utilisation du langage HTML.



**CSS** C’est un langage utilisé pour décrire des styles réutilisables pour la présentation de documents écrits dans un langage de balisage. Son concept a été créé par Hakon Wium Lie en 1994. En décembre 1996, le W3C a fait de CSS un cahier des charges.

Qui permet aux développeurs Web de modifier la mise en page et l'apparence de leurs pages Web.

Par exemple, CSS peut être utilisé pour modifier la police utilisée dans certains éléments HTML, ainsi que sa taille et sa couleur. Un seul fichier CSS peut être lié à plusieurs pages, ce qui permet au développeur de modifier l'apparence de toutes les pages en même temps.

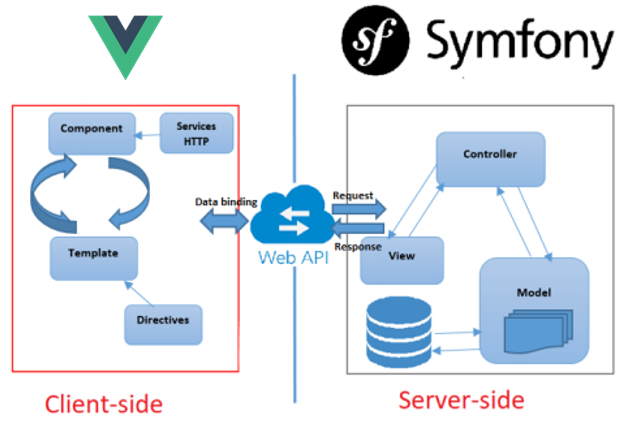


**JavaScript** est un langage de programmation qui permet de créer du contenu mis à jour de façon dynamique, de contrôler le contenu multimédia, d'animer des images, et tout ce à quoi on peut penser.

**JavaScript Object Notation** (**JSON**) est un format de données textuelles dérivé de la notation des objets du langage JavaScript. Il permet de représenter de l'information structurée comme le permet XML par exemple.).

**3.2 Architecture adoptée**

**3.2.1Architecture physique**



***3.2.2Architecture logique***

Dans le domaine informatique, l’architecture logique décrit les composants abstraits (souvent les services et leurs clients) et les interactions entre eux. Dans ce contexte nous allons commencer par présenter le modèle MVC.

***3*.2.2.1 Architecture logique MVC** **(Model View Controller)**

**MVC** signifie Modèle, Vue et Contrôleur. MVC sépare les applications en trois composants : modèle, vue et contrôleur.

* **Modèle**

Le modèle représente la forme des données et la logique métier. Il maintient les données de l'application. Les objets de modèle récupèrent et stockent l'état du modèle dans une base de données. Le modèle est une donnée et une logique métier.

* **View**

View ou Vue est une interface utilisateur. Afficher les données d'affichage en utilisant le modèle pour l'utilisateur et leur permet également de modifier les données. La vue est une interface utilisateur.

* **Contrôleur**

Le contrôleur gère la demande de l'utilisateur. En règle générale, l'utilisateur interagit avec la Vue, ce qui déclenche à son tour la demande d'URL appropriée. Cette demande sera gérée par un contrôleur. Le contrôleur rend la vue appropriée avec les données du modèle en réponse. Le contrôleur est un gestionnaire de demandes.

**3.3. Déroulement des Sprints**

**3.3.1. Sprint1** : …………………………………………………………… (indiquer le nom du sprint)

**3.3.1.1. Conception détaillée du sprint 1**

**3.3.1.2. Développement du sprint 1**

**3.3.1.3. Tests relatifs au sprint 1**

**3.3.2. Sprint 2** : …………………………………………………………… (indiquer le nom du sprint)

**3.3.2.1. Conception détaillée du sprint 2**

Dans ce paragraphe on insère le diagramme de cas d’utilisation Dans ce paragraphe on explique en détail les cas d’utilisation et on insère les diagrammes d’activités des cas les plus pertinents Dans ce paragraphe on décrit l’architecture adoptée : principe et avantages Page 13 sur 14

**3.3.2.2. Développement du sprint 2**

**3.3.2.3 Tests relatifs au sprint 2**

**3.3.3. Sprint 3:…………………………………………………………….. (Indiquer le nom du sprint)**

**3.3.3.1. Conception détaillée du sprint 3**

**3.3.3.2. Développement du sprint 3**

**3.3.3.3. Tests relatifs au sprint 3**

**3.4. Chronogramme Conclus**

**Conclusion**