Ecole Polytechnique de l'Université François Rabelais de Tours
Département Informatique
64 avenue Jean Portalis
37200 Tours, France
Tél. +33 (0)2 47 36 14 14
polytech.univ-tours.fr

Projet Recherche & Développement

2019-2020

Projet VISIT : Web Application Administration des sites patrimoniaux de la région Centre

Tuteur académique
Gilles Venturini

Étudiant
Thomas Charlet (DI5)

7 avril 2020

Liste des intervenants

Nom	Email	Qualité
Thomas Charlet	thomas.charlet@univ-tours.fr	Étudiant DI5
Gilles Venturini	gilles.venturini@univ-tours.fr	Tuteur académique, Département Informatique

Avertissement

Ce document a été rédigé par Thomas Charlet susnommé l'auteur.

L'Ecole Polytechnique de l'Université François Rabelais de Tours est représentée par Gilles Venturini susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnait assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assument l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnait qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.

Pour citer ce document

Thomas Charlet, *Projet VISIT : Web Application Administration des sites patrimoniaux de la région Centre*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2019-2020.

```
@mastersthesis{
    author={Charlet, Thomas},
    title={Projet VISIT : Web Application Administration des sites patrimoniaux de la région
      Centre},
    type={Projet Recherche \& Développement},
    school={Ecole Polytechnique de l'Université François Rabelais de Tours},
    address={Tours, France},
    year={2019-2020}
}
```

Document mis en forme sous LaTeX
grâce à la classe de document polytech.cls V2.2.5 (2017/05/02)

Table des matières

Table des figures

1 Introduction

Ce document présente la conception d'un outil permettant de proposer à des touristes une visite dynamique et originale des sites patrimoniaux de la région Centre. Il devra permettre à des responsables de site de pouvoir administrer les informations sur les œuvres qu'ils possèdent . De plus, la mise en place d'une application mobile à destination des visiteurs aura pour objectif de redéfinir une visite touristique sous une vision nouvelle et originale.
Le besoin du projet a été émis par la Région Centre. Le maître d'ouvrage et maître d'œuvre est représenté par Monsieur Gilles Venturini. Ce rapport concerne essentiellement la partie Web Application du projet VISIT et vaut pour preuve et engagement du rédacteur à effectuer les tâches qui y sont spécifiées comme faisant partie de son domaine d'attribution.

2 Contexte de réalisation

1 Contexte

Dans le cadre d'un plan visant à faciliter l'accès aux informations du patrimoine national pour l'ensemble des citoyens, la région Centre a financé un projet de recherche et développement sur un outil permettant de mettre en place des visites touristiques interactives. Une commande portant sur une application répondant à cette problématique a donc été émise par la région via un appel pour ce projet de recherche d'intérêt régional.

2 Objectifs

Le sujet d'application touristique interactive a déjà été exploré à travers la mise en place d'applications permettant d'obtenir des informations sur des œuvres. Néanmoins lesdites informations ne sont accessibles que par l'intermédiaire de supports réels destinés à cet effet. En effet, nous tenons pour exemple la technologie QR-Code utilisée par quelques sites touristiques pour permettre aux visiteurs d'obtenir des informations sur l'œuvre qu'ils admirent à ce moment précis. L'idée principale est maintenant de pouvoir se passer de ces QR-codes et d'utiliser la reconnaissance d'image afin d'obtenir directement des informations.

L'objectif de ce projet est donc de faire en sorte qu'un touriste puisse obtenir des informations interactives sur son smartphone ou sa tablette lorsqu'il se trouve devant une œuvre d'art de la région. A partir de son appareil portable, ce visiteur pourra prendre en photo (« flasher ») l'objet d'intérêt (dans un château, un musée, une église, un lieu touristique etc.) et notre système pourra alors identifier de quel objet (ou sous- partie d'un objet) il s'agit avec des méthodes de reconnaissance d'images (et d'intelligence artificielle). En retour, l'utilisateur gagnerait sur son appareil mobile des informations multimédia sur l'objet, informations issues de bases de données de qualité (bases définies par des chercheurs dans le cadre du projet, plateforme de données d'Ipat, autres bases historiques). En outre, le système construit permettrait à tout responsable d'un site de patrimoine de la région d'ajouter des œuvres dans la plateforme afin d'augmenter la qualité des visites avec des outils numériques innovants. Ce projet s'articule avec le programme de l'ARD 2020 "Intelligence des Patrimoines"

Le système sera constitué d'une base de données regroupant les œuvres, leurs localisations ainsi que des informations d'authentification permettant l'ajout de nouvelles œuvres sur la plateforme pour les responsables du site.

Le développement de l'application s'orientera essentiellement autour de 2 aspects, à savoir :
• La création d'une Web App permettant d'administrer la plateforme et ainsi permettre l'ajout ou la mise à jour des œuvres présentes sur un site patrimonial,
• La création d'une Application Android utilisant la reconnaissance d'image pour identifier une œuvre et fournir à l'utilisateur des informations la concernant.

3 Hypothèses

Comme dans tout projet de cette envergure, des composantes peuvent amener à être modifier au cours du projet.

En effet, en prenant l'exemple du stockage des données, il est possible que le modèle adopté dans ce rapport puisse subir des modifications pour répondre à des problématiques apparues au cours de la conception. Il en est de même avec les autres modules qui composent le modèle global du projet sur lesquels nous reviendront plus tard dans ce document.

Certaines fonctionnalités étant plus difficiles à réaliser et/ou moins importantes que le cœur de l'application, il est possible que notre équipe ne réponde qu'à une partie des besoins en accord avec le chef de projet, au vu des délais de réalisation courts.

4 Bases méthodologiques

Avant de parler des bases méthodologiques adoptées pour ce projet voici la composition de l'équipe en charge de sa réalisation : (Web App et App Android)

- Gilles Venturini, Chef de Projet et Encadrant Web App
- Olivier Roussey, Assistant Ingénieur
- Barthelemy Serres, Encadrant App Android
- Thomas Charlet, Etudiant Ingénieur
- Timothe Sanouiller-tourne, Etudiant Ingénieur

Nous utilisons durant ce projet les outils suivants :

- Pour la communication au sein de l'équipe, le transfert et le travail en collaboration de fichiers de documentation : Discord, un espace Google Drive et essentiellement les mails
- Pour le suivi des tâches et l'évolution du projet : Trello, Project Professionnel
- Pour le versionning du projet : GitLab
- Pour la création, la gestion et l'administration de la base de données (du moins pour les protypes) : phpMyAdmin

La section suivante décrit davantage l'aspect bases méthodologiques adoptées pour la partie Web Application.

Pour ce qui est de la modélisation des fonctionnalités à réaliser, nous avons choisi d'utiliser le langage UML permettant ainsi de visualiser les interactions entre les différentes entités qui composent le système.

Dans le cas du développement d'une plateforme Web à destination des responsables de sites patrimoniaux, il a été décidé d'utiliser les différents langages liés à ce type d'application (PHP, HTML, CSS, JavaScript) .

De manière plus spécifique, nous utiliserons le Framework « Symfony PHP » avec « api-platform » pour créer une API REST destinée à permettre l'accès aux données stockées dans la base.

La création de l'interface d'administration des œuvres présentes sur un site patrimonial se fera à l'aide du Framework JavaScript « Angular ».

Les sources du logiciel seront versionnées grâce à Git et gérées depuis l'interface Web GitLab nous permettant ainsi de contrôler l'avancement du projet à travers l'évolution des différentes branches composant le dépôt Git. Ce dernier permettra également de mettre en place la documentation du projet ainsi que le suivi de bugs.

Pour le développement de ce projet, il est propice de mettre en place un feedback régulier et de pouvoir fournir des livrables fréquemment afin de contrôler les critères de Vérification et de Validation . Dans cette optique, nous allons mettre en place une méthode « agile » ou « pseudo-agile ». Pour rappel dans le cas précis de ce projet le feedback ne sera pas obtenu auprès du client direct (Région Centre) mais auprès de l'encadrant. Nos partenaires sont pour le moment "le musée des beaux arts de Tours" ainsi que "la cité royale de Loches". Des périodes de « Sprint » seront alors mis en place lesquelles auront pour objectifs la réalisation d'une ou plusieurs tâches précises et seront planifiées et notifiées dans le planning prévisionnel présent en annexe. Toutefois ces périodes de Sprint pourront être modifiées en accord avec l'encadrant dans le cas où une fonctionnalité nécessite davantage de temps ou de ressources pour pouvoir être développer.

On s'approchera donc d'une méthode Agile – Scrum avec des sprints à période flexible permettant ainsi de proposer une implémentation plus souple des fonctionnalités désirées.

5 Méthode de test et assurance de la qualité

Les méthodes de test et le plan d'assurance qualité sont décrits dans un document en annexe et complète les spécifications du projet.

3 Description Générale

1 Environnement du projet

Comme il a été évoqué dans les parties précédentes, le sujet de création d'une application Web API Tourisme fait parti intégrante d'un projet comportant également la création d'une application Android tourisme à destination des visiteurs d'un site patrimonial. Ce projet VISIT s'étend à l'ensemble de la Région Centre et pourrait éventuellement voir étendre ses frontières au territoire national suivant les résultats obtenus.
Des recherches concernant le sujet ont auparavant été effectués par les encadrants du projet ou d'autres élèves ingénieurs des années passées. Il nous a cependant été demandé de prendre connaissance du sujet de façon personnelle sans nécessairement se baser sur l'existant afin d'élaborer de nouvelles idées et de soulever de nouvelles problématiques.
Le projet est séparé en 2 modules principaux à savoir la réalisation d'une Web App d'administration et d'une Android App qui évolueront en parallèle. Ces deux environnements distincts seront amenés à communiquer tout au long du projet. Dans le cadre de la réalisation du projet, un serveur a été mis à disposition par Polytech pour le stockage des données et l'hébergement du site.

2 Caractéristiques utilisateurs

Pour cette partie, nous allons revenir dans le cadre global du projet, à savoir la Web API et l'application Android. Etant donné que certaines fonctionnalités concernent le transfert de données de la base vers l'application Android, il est important de décrire l'intégralité des utilisateurs, ceux de l'application mobile compris. Etablir le profil des utilisateurs qui vont être amenés à utiliser notre application nous permettra d'établir une interface utilisateur la plus efficiente et la plus satisfaisante possible.

On peut alors différencier 3 types d'acteurs qui auront chacun des autorisations déférentes sur la plateforme web :

• Les Visiteurs : Ils n'ont pas accès aux fonctionnalités de modification du contenu de la plateforme et ont donc seulement un droit de consultation de certaines informations sur l'application mobile, lesquels seront définis au préalable. Les visiteurs seront amenés à se rendre occasionnellement sur la plateforme ou sur l'application dans le cadre de la planification de visite ou

de visite en cours d'exécution. (Droits associés : Consultation uniquement pour l'application mobile pour le moment)
• Les Responsables des sites patrimoniaux : Ils ont un accès complet aux œuvres et aux sites auxquels ils sont rattachés et peuvent donc effectuer diverses requêtes sur « leurs contenus ». Ces personnes seront destinées à se rendre plutôt régulièrement sur la plateforme dans le but de mettre à jour le contenu du site patrimonial dont ils ont la charge. (Droits associés : Ajout, Modification, Suppression, Consultation)
• Le Super-administreur : Personne définie comme ayant tous les droits de modification et d'administration de la plateforme. Cette personne contrôlera les accès à la plateforme en agissant comme un référant pour toute personne souhaitant accéder au site Web. Ce dernier est nécessaire pour donner les droits d'accès à un nouveau responsable de site. Ce profil utilisateur correspond à une personne ayant une approche familière de l'application puisqu'il sera amené à agir régulièrement sur la plateforme de par son statut de super administrateur. (Droits associés : Admin, accès aux logs)

Pour la suite du développement de l'application, on émettra l'hypothèse que les utilisateurs au statut de visiteur ou de responsable de site ne possèdent pas de compétences particulières en Informatique. Pour l'administrateur de la plateforme on supposera qu'il possède des compétences en Informatique qui lui permettent de comprendre et d'interagir de manière satisfaisante avec la plateforme. Il faudra donc adapter la complexité d'utilisation de l'interface ainsi que l'accès aux fonctionnalités demandées en conséquence.

3 Fonctionnalités du système

De même que nous sommes revenus dans le cadre global de l'application précédemment pour évoquer l'ensemble des utilisateurs potentiels de l'application nous allons cette fois ci nous recentrer sur la partie web application d'administration et plus précisément sur les fonctionnalités du système qui touche à ce module.

Pour rappel, il est important de distinguer les 3 différents acteurs pouvant interagir avec l'application d'administration web. Remarque : Ne sachant pas pour le moment si le visiteur pourra accéder à l'application avec uniquement des droits de consultation nous distinguerons par la suite ses interactions avec le système par le symbole (*).

Liste Fonctionnalités systèmes suivant les droits utilisateurs attribués sur la plateforme web :

• Visiteur (*) :

- Accès accueil du site et possibilité accès à la fenêtre « Log In » (*)

• Responsable de site :

- Log In - Log Out
- Consulter informations profil (nom, prénom, email, téléphone)
- Consulter informations du site en charge
- Consulter liste des auteurs concernant le *site en charge
- Consulter la liste des zones du site en charge
- Consulter la liste des œuvres du site en charge
- Gestion informations profil (informations personnelles + modification du mot de passe)
- Gestion des auteurs du site en charge (Ajout, Modification, Suppression)
- Gestion des zones du site en charge (Ajout, Modification, Suppression)
- Gestion des œuvres du site en charge (Ajout, Modification, Suppression)

*site en charge : un responsable de site est affecté à un unique site, il est donc en charge de sa gestion.

• Administrateur :

- Log In - Log Out
- Consulter informations profil (nom, prénom, email, téléphone, statut admin)
- Consulter informations de l'ensemble des sites
- Consulter informations de l'ensemble des auteurs
- Consulter informations de l'ensemble des zones d'un site
- Consulter informations de l'ensemble des œuvres
- Gestion des inscriptions de nouveau responsable de site (Gestion des droits)
- Gestion informations profil (informations personnelles + modification du mot de passe)
- Gestion de tous utilisateurs (Ajout, Modification, Suppression)
- Gestion de tous sites (Ajout, Modification, Suppression)
- Gestion de tous auteurs (Ajout, Modification, Suppression)
- Gestion de tous zones (Ajout, Modification, Suppression)
- Gestion de tous œuvres (Ajout, Modification, Suppression)

Nous reviendrons plus en détail sur chaque fonctionnalité ainsi que ses déclinaisons en sous fonctionnalités dans la suite du document. A suivre un diagramme résumant les cas d'utilisation globale de la Web Application.

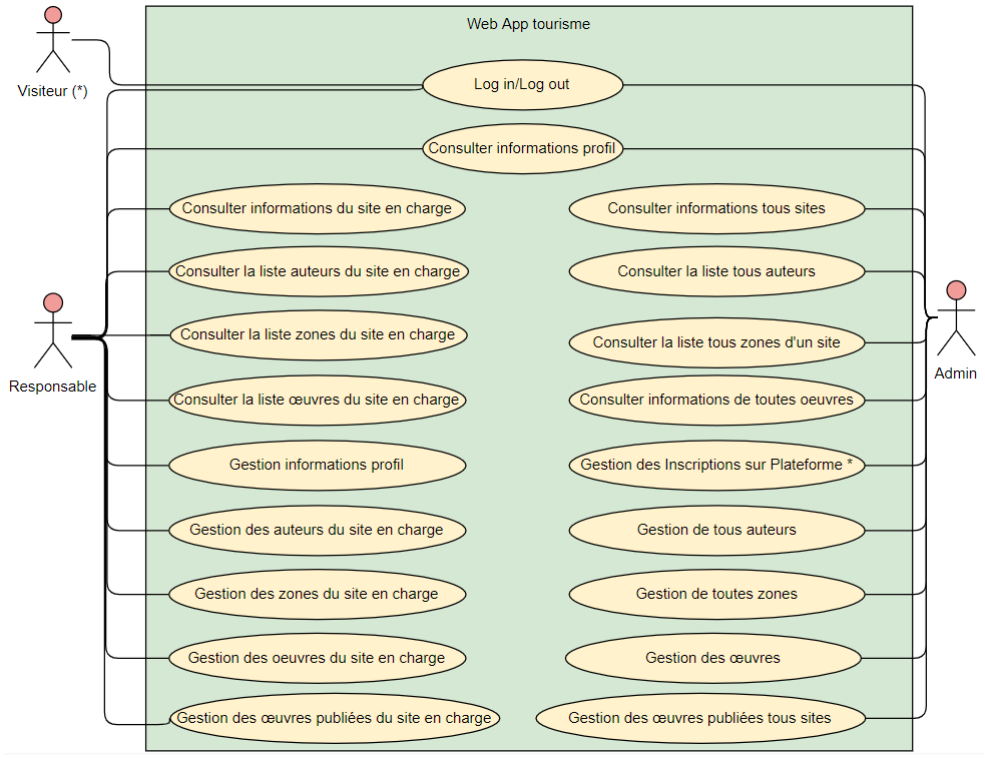


Figure 1 – *Digramme des cas d'utilisation*

Comme nous avons pu le voir dans le diagramme précédent, les principaux acteurs et objets constituant le système sont les suivants :
• Acteurs : Précédemment identifiés comme étant les Visiteurs, les Responsables de site ainsi que l'Administrateur de la plateforme.
• Object :

- Site : Correspond à un endroit géographique où se situe un regroupement d'œuvres,
- Zone : Permet le découpage d'un site en entités distinctes comportant un ensemble d'œuvres,
- Œuvre : Objet ou représentation artistique appartenant au patrimoine,
- Auteur : Personne ayant réalisé une œuvre présente sur un des sites recensés dans la base.

Nous reviendrons plus précisément sur ces objets dans la partie expliquant la structure générale du système.

4 Structure générale du système

Cette partie du projet vise à mettre en place une interface d'administration Web à destination des responsables de sites patrimoniaux leur permettant ainsi de déposer des œuvres et autres informations dans la section les concernant sur la plateforme.

Nous utiliserons dans le cadre de ce projet une architecture 3 tiers. Cette dernière se matérialisera sous la forme de la création de modules les plus indépendants les uns des autres possible (Modèle/Vue/Controller). Le système sera alors composé d'une base de données, d'une API REST ainsi que de l'interface utilisateur.

- Base de données : permet le stockage des informations et correspond à la représentation physique du modèle de données,
- API REST : Système de communication permettant par l'intermédiaire de protocoles REST (HTTP) d'exposer les données contenues dans la base à destination de l'application ayant émis une requête.
- Interface : Partie visuelle de l'application et destinataire des données requêtées à la base à travers l'API.

Nous reviendrons plus en détail sur chacun de ces points dans la suite du document.

4 Etat de l'art

Ce projet se distingue en deux modules distincts à développer, à savoir une Web Application de gestion de sites et œuvres recensés dans la base de données ainsi qu'une application mobile à destination des visiteurs de ces sites patrimoniaux. La partie faisant l'objet de cet état de l'art correspond à la mise en place de la plateforme Web d'administration.

La conception d'application Web a rapidement évoluée au cours du temps. La manière de concevoir une plateforme web actuelle nous permet de mettre en place des sites dynamiques à travers des architectures système plus élaborées comme le « multitier ». Cette façon de penser se traduit par la séparation des modules destinés à la gestion des données, les processus applicatifs ainsi que les vues clients.

L'objectif du projet serait de fournir aux utilisateurs une interface d'administration efficace et satisfaisante utilisant le type d'architecture précédent permettant ainsi de faciliter la maintenance et la capacité à intégrer des évolutions dans la plateforme.

1 Stockage des données

L'ensemble du module destiné au stockage des données et API nécessaire pour la communication entre base de données et applications (Web plateforme, application Android) est hébergé sur un serveur Polytech.

1.1 Type de base de données

Il existe différentes manières de stocker les données dans une base nécessitant d'étudier les cas d'utilisation potentiels de l'application afin de définir le modèle à adopter et de correspondre au mieux les besoins clients initiaux.

Deux modèles ont rapidement été parcourus dans ce sens à savoir :
• Le modèle relationnel, le objets et associations sont matérialisés à l'aide de relations (tables) dans ce modèle.

- Modèle basé sur les données explicitant les relations entre les données
- Basé sur un modèle mathématique formel
- Le lien vers des programmes objets est plus difficile

• Le modèle objet, les données sont alors décrites comme dans la programmation orientée objet à savoir à l'aide de classe comportant des attributs et des méthodes. Ce modèle tente de résoudre plusieurs problématiques issues du maillage avec le modèle relationnel.

- Représente les objets du système et leur structure
- Permet de voir les associations, les compositions, les agrégations et l'héritage entre les objets
- Peut être utilisé pour implémenter directement une solution

Cependant le recours à un modèle objet de représentation des données demande une gestion manuelle des contraintes d'intégrité entre les objets ce qui ajoute une certaine complexité de développement. Il permet toutefois d'effectuer une navigation plus souple dans la base de données.

Afin de visualiser la représentation des données en mémoire, la réalisation d'un modèle relationnel utilisant ces dernières a été proposé comme nous pourrons le voir dans la partie « Analyse et conception ».

Finalement le choix d'un stockage par base de données « Objet » semble être une bonne solution.

2 Communication : Données – Processus applicatifs

2.1 Contexte

Après avoir pensé à un système permettant de stocker les données nécessaires à l'application, nous nous sommes intéressés à la manière d'accéder à celles-ci. La possibilité de mettre en place un processus de communication adapté à nos besoins et permettant l'échange de données entre les processus applicatifs et la base de données est primordial. La problématique consiste à trouver une solution actuelle permettant la gestion de multiples utilisateurs mobiles utilisant des applications multi-plateforme. La solution consistant à la mise en place d'une API nous a alors paru évidente, cette dernière offrant la possibilité d'exposer sur internet des services s'adaptant au canal utilisé.

Figure 1 – *Les plateformes web à travers le temps*
https://blog.octo.com/strategie-d-architecture-api

La tendance actuelle en matière de développement Web est de procéder de plus en plus à une ouverture du système d'information permettant ainsi d'externaliser les services à l'ensemble des acteurs externes. C'est dans cette optique notamment qu'a été pensé le concept d'Open API.

Remarque : Une API peut être définie d'Open API si celle-ci est en mesure de fournir une documentation interactive côté client comme côté serveur.

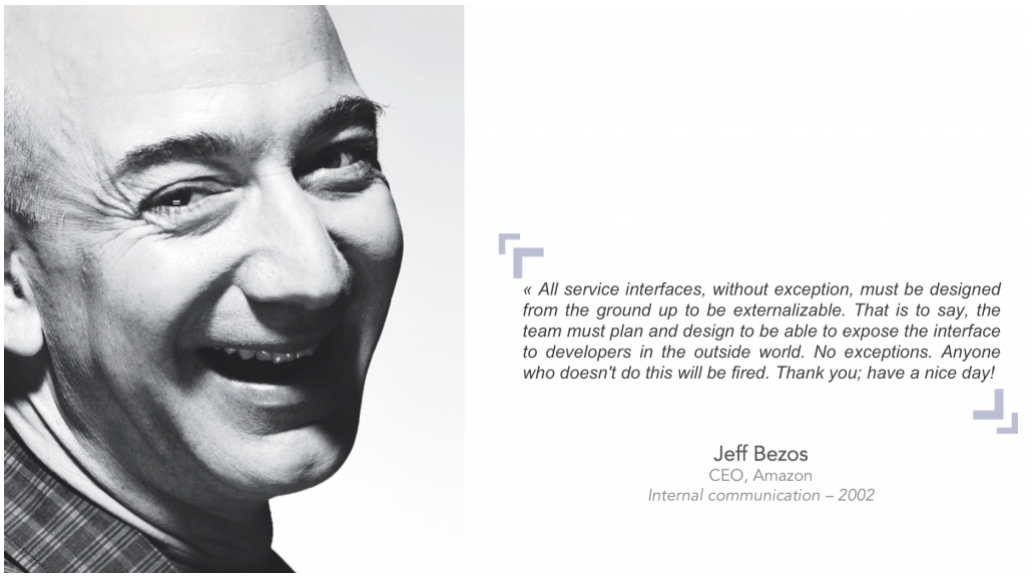


Figure 2 – *L'API de nos jours*
https://blog.octo.com/strategie-d-architecture-api

2.2 API et Architectures

Comme il l'a été présenté dans la partie précédente, il est de nos jours souhaité sinon imposé d'utiliser une API afin d'accéder à un système d'information. On peut définir le terme API, acronyme de Application Programming Interface comme étant un système permettant de mettre en place des processus de service inter-logiciel et d'exposer le système via des protocoles HTTP au travers d'une interface normalisée.

Le rôle d'une API est donc de proposer un ensemble de ressources à exposer à l'aide de services Web . Les architecture web mises en place au cours du temps s'orientaient principalement autour du REST en opposition constante des modèles architecturaux basés sur SOAP. En effet, l'architecture SOA (Service-Oriented Architecture dont fait partie SOAP) s'est vu peu à peu étendre vers un modèle considéré parfois comme plus léger que son prédécesseur à savoir l'architecture orientée Web (WOA). Cette manière de penser plus récente vise à optimiser les interactions navigateur et serveur en utilisant des technologies davantage orientées Web telles que REST

Figure 3 – *L'architecture WOA*
https://blog.octo.com/strategie-d-architecture-api

Par la suite nous sommes donc amenés à comparer ces deux styles d'architecture Web afin de déterminer celle qui est la plus adaptée à notre projet.

SOAP/RPC	REST
Modélisation par opération	Modélisation par ressources
Cherche à unifier le modèle de programmation distribué et local par l'intermédiaire d'un proxy	Modèle de programmation distribué explicite, et basé sur le WWW
Repose sur un toolkit pour être interprété	Mise sur une bonne expérience développeurs tous niveaux
Consommé que par des serveurs	Consommé par tout type de device, y compris les serveurs

Figure 4 – *Comparatif SOAP vs REST*
https://blog.octo.com/strategie-d-architecture-api

L'architecture orientée Web permet de mettre en place un système d'avantage ouvert et universelle que le modèle SOA qui lui implique l'utilisation de protocoles plus lourds en utilisant un proxy. Nos recherches dans le cadre de ce projet consistent également à proposer une solution s'inscrivant dans les tendances actuelles, critère important pour la maintenabilité et les évolutions futures de l'application.

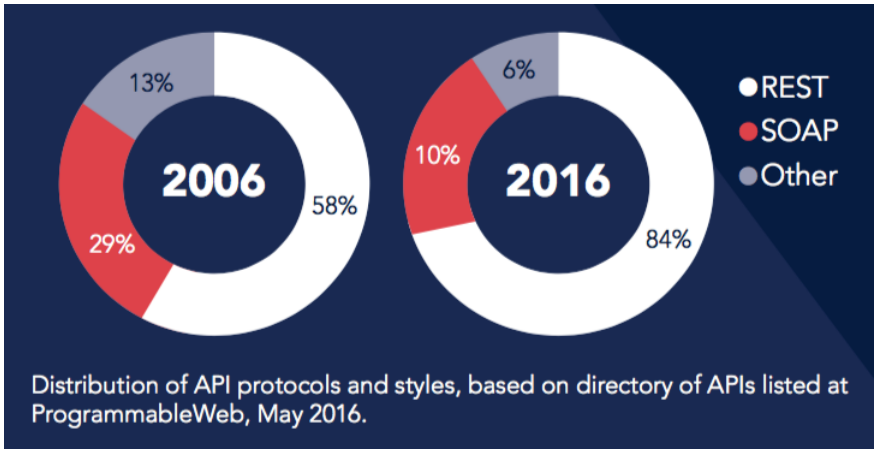


Figure 5 – *Comparatif SOAP vs REST*
Comparatiftendancesactuellesconcernantl\T1\textquoterightarchitectureweb

Nous pourrions davantage rentrer dans les détails concernant le descriptif de ces deux modèles, néanmoins par la simplicité de l'approche REST en matière de requêtes à effectuer, de réponses obtenues et de sa popularité actuelle, notre choix s'oriente vers ce style d'architecture.

2.3 Open API et choix de solution

Nous avons évoqué ce terme dans la partie précédente, ce principe consiste à proposer la création d'un système ouvert notamment via l'exposition de services universelles s'adaptant donc à l'usage que l'on souhaite en faire.

Figure 6 – *API Platform - Symfony*

Notre choix s'est donc arrêté sur api-platform qui est une Open Api utilisant un Framework REST et GraphQL afin de fournir les outils permettant de construire facilement un ensemble de services. Il faudra auparavant préparer l'environnement permettant la mise en place de cette API, cette dernière utilisant le squelette du Framework PHP-Symfony 4 afin de fonctionner.

Nous allons par la suite appliquer l'idée d'API First. C'est-à-dire commencer par construire notre API avant de construire les applications à destination des utilisateurs finaux. Cette manière de développer nous permettra d'appréhender les attentes utilisateurs en se mettant nous même à leur place.

3 Interface Web Administration

3.1 Règles interface Homme-Machine

En accord avec la politique transactionnelle adoptée concernant donc l'accès et le transfert des données pour ce projet, les échanges n'auront lieu que par l'intermédiaire de l'API REST mis en place. Dans cette partie nous allons nous intéresser à la manière de représenter les données à destination des utilisateurs de la plateforme d'administration Web.

Des recherches ont été effectuées dans ce sens afin de trouver les outils permettant de réaliser une interface web efficace et satisfaisante. En parallèle des recherches concernant les technologies à utiliser pour effectuer cette tâche il a fallu réfléchir à la manière de représenter les informations pour l'utilisateur.

En effet, comme il l'a été évoqué précédemment des recherches ont été effectuées au fil de temps permettant de fixer des règles visant à la bonne compréhension d'une interface homme-machine pour un utilisateur. Car si la partie back-end de l'application est très performante, il ne faut pas négliger la partie front-end qui reste le seul maillon de la chaine applicative avec lequel l'utilisateur agit physiquement. Une interface Homme-Machine doit être facile d'utilisation pour tous, sinon un maximum d'utilisateur. Dans cette optique, il est important de définir l'utilisabilité d'une interface, ce terme semblant être un pilier sur lequel on peut s'appuyer pour construire notre interface.
Selon Jakob Nielson, expert en ergonomie, l'utilisabilité peut se définir au travers de 5 critères qu'il faudra par la suite tenter de respecter dans notre conception de l'interface.

• La facilité d'apprentissage : C'est ce qui caractérise le comportement utilisateur lors de la première utilisation de l'application.
• La facilité d'appropriation : Après avoir utilisé plus d'une fois l'application, l'efficacité de l'interface se mesure à travers la capacité de l'utilisateur à mémoriser le comportement à adopter lors de la recherche d'informations ou l'accès aux fonctionnalités de l'application.
• La fiabilité : Notion correspondant à la mesure des erreurs effectuées par le système lors de son utilisation par un utilisateur.

• L'efficience : Correspond à la notion de facilité et donc de rapidité avec laquelle l'utilisateur parvient à réaliser les fonctionnalités souhaitées sur l'application.
• La satisfaction : Critère subjectif décrivant l'interface et étant propre au ressenti utilisateur

Nos recherches s'orienteront donc vers des outils nous permettant de mettre en place les fonctionnalités demandées tout en respectant les critères évoqués précédemment.

3.2 Technologies interface retenues

Figure 7 – *Evolution Technologies Web*
https://blog.octo.com/strategie-d-architecture-api

Un site web a recensé l'évolution des bibliothèques et Framework permettant de faire du web depuis une quinzaine d'années, ce qui constitue un bon point de départ et correspond à l'image ci-dessus.

Figure 8 – *Angular - AR Core FrameWork*

Après avoir étudié les techniques et architectures permettant de réaliser une interface « actuelle » respectant les critères d'utilisabilité, le Framework javascript Angular a été retenu pour la réalisation de la partie interface d'administration. Cette architecture est une réécriture complète d'AngularJS qui était un Framework développé par Google permettant de développer des pages web. Dans le cadre de ce projet nous utiliserons ce dernier sous sa version 8.

Pour rappel, du côté de l'application mobile, la technologie AR Core sera utilisée afin de concevoir l'aspect réalité augmentée. Ce SDK développé également par Google permettra de projeter des modèles 3D d'œuvres appartenant au patrimoine de la région Centre sur un appareil Android (version Android 7.0 Nougat ou plus), et ce indépendamment du mobile possédé.

3.3 Technologie de l'interface de gestion des annotations retenue

En relation avec les fonctionnalités à développer pour ce projet, il est nécessaire d'être en capacité de produire une interface de placement de balise et d'ancres sur une œuvre. Ce système permettra à un responsable de disposer des points caractéristiques sur une image AR. Ces derniers pourront alors contenir des informations, pour la majorité textuelle, permettant ainsi à l'utilisateur de pouvoir bénéficier d'explications complémentaires sur une œuvre. Des pistes de recherche nous ont été fournies dans ce sens, permettant de nous diriger vers des technologies telles que SVG et balises canvas.

Figure 9 – *SVG*

SVG ou Scalable Vector Graphics est un format de données permettant de décrire des graphiques vectoriels qui seraient dans notre cas l'emplacement des balises et ancres déposées sur une image. Canvas est un concept intégré dans le Framework Angular permettant le dessin d'environnement 2D (texte, image, autres objets) et le stockage en mémoire sous la forme de pixels.

Par ailleurs, Angular apparaît comme très efficace dans la manipulation du dom html5 avec notamment les balises canvas, semblant permettre de réaliser les tâches attribuées, nous avons pour le moment choisi cet outil afin de répondre aux besoins de placement d'annotations sur les images des œuvres. Cependant nous pourrons être amenés à revenir vers SVG si l'utilisation de balises canvas ne s'avérait pas suffisante.

5 Analyse et Conception

Dans l'optique de proposer une architecture modulaire permettant de faciliter la maintenance et l'apport d'évolution au projet, nous avons choisi le pattern MVC pour notre modèle. Nous avons pu mettre en place un tel modèle opérant une distinction entre les parties modèle, vue et controleur à l'aide du Framework PHP Symphony pour l'API REST. Un modèle davantage MVVM (Modèle-Vue-Vue-Modèle reposant sur la programmation évènementielle) sera utilisé avec javascript Angular 8 (ainsi que la librairie RxJs pour l'aspect programmation réactive et fonctionnelle) sur lequel nous reviendrons plus tard. L'analyse suivante s'est davantage portée sur les aspects modèle et Controller à mettre en place.

1 Modèle

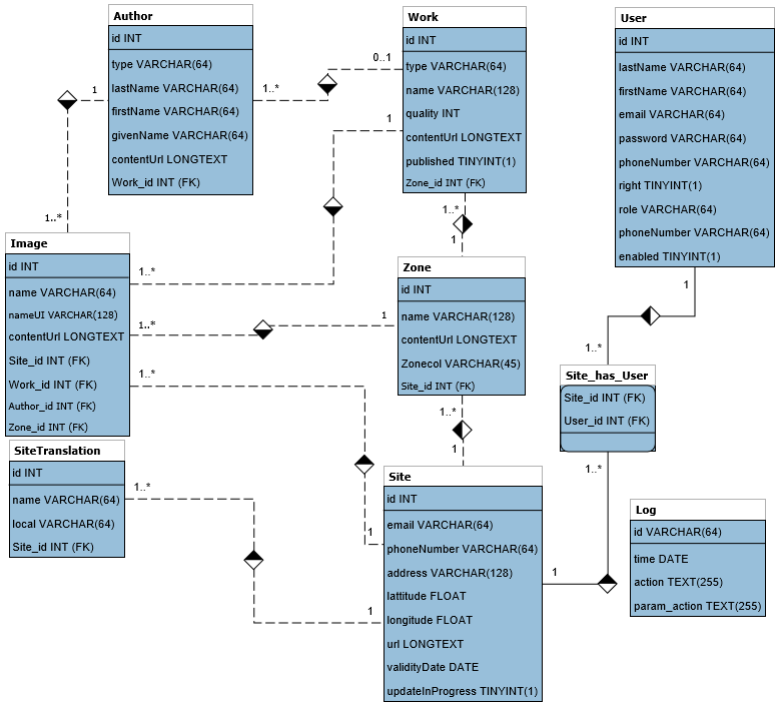


Figure 1 – *Premier modèle de base de données*

Dans une phase de prototypage et de recherche d'un modèle de base de données correspondant aux besoins du projet et répondant aux problématiques soulevées au cours des réunions, il a été demandé d'imaginer et de fournir une première version personnelle du modèle à adopter. Par soucis de clarté, mon choix s'est porté sur une représentation de la base de données sous forme de modèle relationnel. Le diagramme précédent constitue la première vision globale de la base de données permettant de stocker l'ensemble des éléments nécessaires au projet. Cette vision sera sans doute amenée à évoluer au cours du développement.

Dans la suite de cette section dévouée à la structure générale du système, nous allons revenir sur les principales composantes du système afin de détailler leurs rôles ainsi que leurs interactions.

1.1 Eléments constituants la base de données

Explorons donc davantage le prototype de base de données proposé afin d'expliciter le rôle des principales composantes dans la réponse aux besoins du projet. Découpons l'analyse par tables présentes dans le modèle.

• Site : Entité permettant de regrouper les informations essentielles décrivant un site patrimonial.

- Il comprend en autre l'ensemble des indicateurs géographique permettant de situer le site. Cet aspect permet de délimiter un périmètre autour du site afin d'anticiper l'absence de connexion sur un site touristique et le cas échéant proposer le chargement de données nécessaires à la visite avant de pénétrer dans ce périmètre.
- Une date de validité permet de savoir si le site est à jour sur la plateforme et s'il est toujours possible de le visiter.
- La possibilité d'inclure un lien menant vers le site internet officielle du lieu à visiter a également été pris en compte dans la conception.

• Zone : Entité permettant de découper un site en zones contenants un regroupement d'œuvres définies.

- Découpage permettant d'organiser la mise en place du processus de reconnaissance AR car permettant de situer le visiteur dans l'espace. (une œuvre appartenant à une zone).
- Cette distinction permettrait également de regrouper les œuvres sous la forme de « paquets » ayant comme dénominateur commun leur appartenance à un même site.

• Work : Entité correspondant au cœur du projet à savoir les œuvres,

- Comporte notamment un indice de qualité permettant de renseigner l'utilisateur de l'application sur le besoin de plus ou moins se mettre en face de l'œuvre et à bonne distance. Cet attribut aide également le responsable de site sur la nécessité de prendre une photo «correcte » de l'œuvre avant de la poster sur la plateforme.
- Chaque œuvre possède un statut qui est très important pour le fonctionnement de l'application globale à savoir l'attribut « published ». Cet aspect permet de différencier notamment les œuvres disponibles dans l'application sous sa forme « Debug » (Responsable) ou « Release ». (Visiteur)
- Chaque Œuvre présente a donc un auteur, et peut avoir un site et une zone attribués dans le cas où l'œuvre est consultable lors de la visite.

• Auteur : Personne ayant réalisé une œuvre présente sur un des sites recensés dans la base.

• Image : Entité permettant essentiellement de stocker les images des œuvres et de pouvoir regrouper les œuvres sous un endroit unique où elles sont situées géographiquement.

• User : Entité permettant la mise en place de statut utilisateur et donc le contrôle des droits qui leurs sont attribués

- Les attributs correspondant au mot de passe et email sont les éléments de connexion de chaque utilisateur à son compte avec possibilité de modifier son mot de passe pour chaque utilisateur connecté.
- Il a été laissé la possibilité à un utilisateur de pouvoir être responsable de plusieurs sites et à un site d'être géré par plusieurs utilisateurs.
- On suppose qu'il n'y a cependant qu'un super administrateur parmi les utilisateurs pour commencer

L'entité « Log » ne permet que le stockage des logs communiqués par l'application Android dans le cas où il s'avérait nécessaire d'effectuer ce transfert. De même, la mise en place de l'entité « SiteTranslation » dans le modèle correspond à une problématique anticipée concernant un statut particulier (« Traduction informations sites dans d'autres langues ») des sites qu'il est nécessaire de préciser.

1.2 Eléments relatifs à la plateforme

Cette partie recense davantage les éléments du système avec lesquels l'utilisateur sera directement amené à agir. En effet, selon ses droits, l'utilisateur de la Web App pourra manipuler certains éléments.

• Pour le responsable de site : La mise en place d'ancres et de balises sur une image permettant de créer des bulles textuelles comportant des informations sur l'œuvre. Mise en place de requêtes vers la base de données afin d'afficher les résultats fournis par les filtres de recherche.

• (*) Pour les visiteurs : La possibilité de consulter les sites, sinon les œuvres qui y sont présentes et d'appliquer des filtres de recherche sur le résultat obtenu

• (*) Pour les visiteurs : Uniquement dans le cas où l'idée de consultation pour les visiteurs est retenue et dans le cadre de la mise en place de filtres de recherches, il faudra effectuer des requêtes vers la base de données afin d'afficher les résultats de recherche.

(*) Si l'on donne un accès de consultation de toutes les œuvres, sites à un visiteur non connecté sur la plateforme ou sur l'application mobile à titre d'information (Aspect amené à évoluer au cours du projet). Pour le moment le visiteur n'a pas cette possibilité.

Le modèle de représentation « Objet » ayant été sélectionné le modèle présent dans cette section ne vaut que pour prototype aidant à la compréhension de l'architecture générale du système. Il correspond cependant à une représentation fidèle des caractéristiques de chaque entité ainsi que leurs interactions.

2 API

Nous utilisons donc api-plateforme, une Open pour créer notre système d'accès aux données stockées dans la base. A ces fins un prototype d'API REST se basant sur le modèle de base de données précédent a été réalisé.

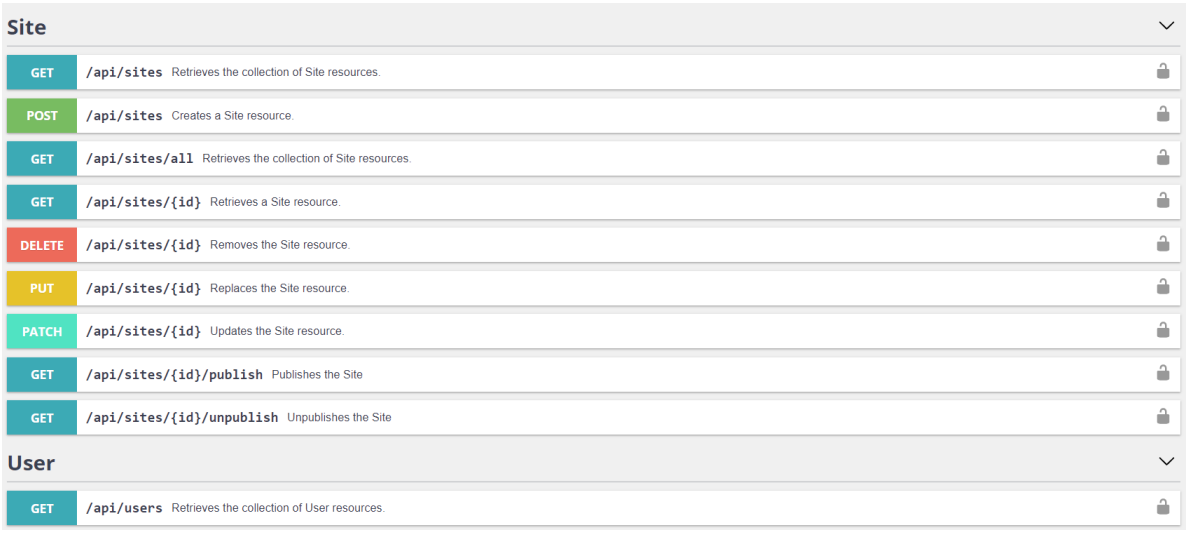


Figure 2 – *Premier prototype API REST (sous api-platform)*

Ce prototype a été réalisé avec pour attente de tester les canaux de communications inter-système et réponses obtenues après requêtages. Il a également permis d'établir une liste complète des requêtes à réaliser pour répondre à l'ensemble des besoins clients.

3 Interface

Afin de visualiser l'accès aux fonctionnalités et d'appréhender l'utilisation de l'application par un utilisateur extérieur, des mockups représentants l'interface d'administration des sites patrimoniaux ont été réalisés. Ces derniers sont davantage décrits dans la section relative aux « Interfaces Homme/Machine » présente en annexe de ce document.

4 Modifications apportées

Les modifications apportées lors du semestre 10 du côté analyse concernent essentiellement le modèle de données qui a bien évolué au cours du développement. Pour pouvoir représenter de nouveau ce modèle via un schéma de base de données nous avons repris le modèle relationnel précédent et l'avons modifié pour aboutir au schéma actuel de la base de données.

Les nouvelles entités incluses dans le dernier modèle en plus de celles présentes dans la première version sont les suivantes :

- SiteState : Entité permettant d'enregistrer les différentielles entre version d'un même site patrimonial permettant de mettre à jour les packages à destination de l'application mobile.
- MediaObject : Entité permettant essentiellement de stocker les images des œuvres et de pouvoir regrouper les œuvres sous un endroit unique où elles sont situées géographiquement.
- Annotation : Entité permettant de créer et d'administrer des annotations pouvant être intégrées à des oeuvres d'un site patrimonial par l'intermédiaire de la table de correspondance « annotated_image ».

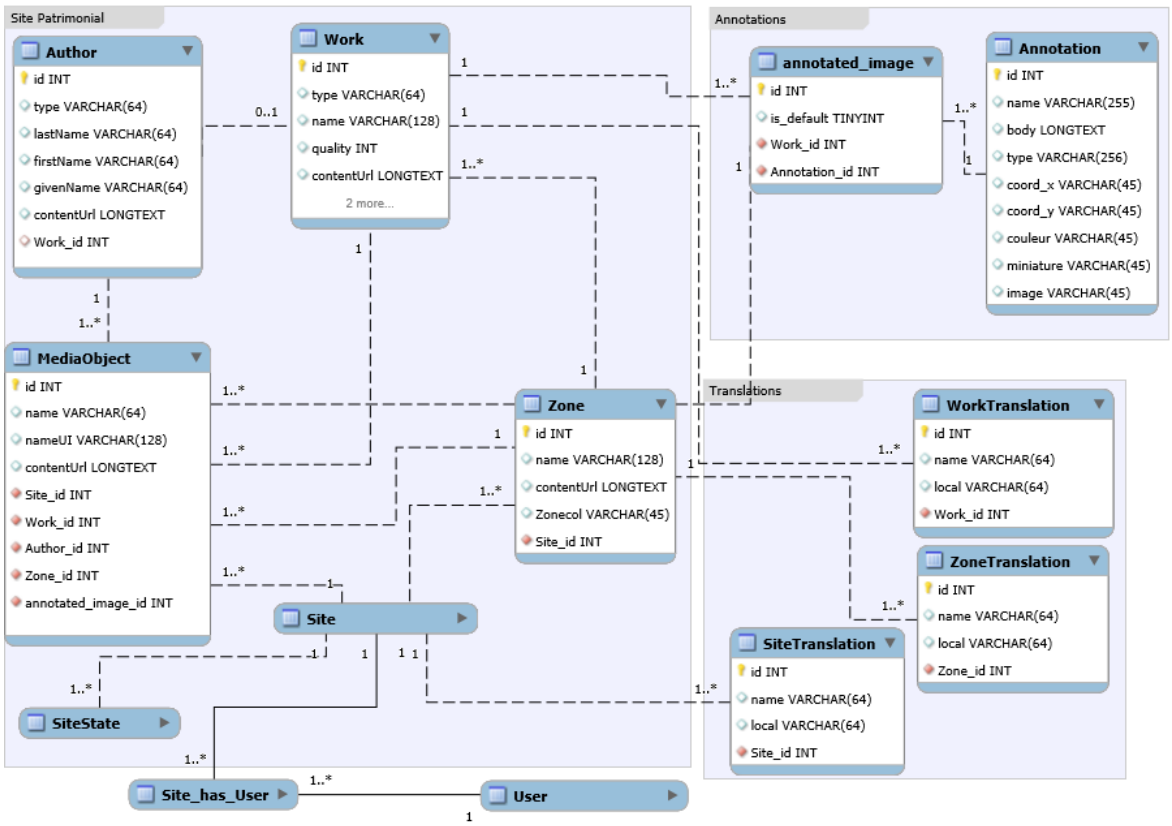


Figure 3 – *Modèle de données final de la base de données*

En plus de la mise en place de l'entité « SiteTranslation », nous avons rajouté « Zone Translation » et « WorkTranslation » dans le modèle correspondant à la même problématique anticipée évoquée dans la partie « 5.1.1. Eléments constituants la base de données » à savoir un statut particulier (« Traduction informations sites, zones et œuvres dans d'autres langues ») des sites qu'il est nécessaire de préciser.

Nous nous sommes conformés au plan de départ concernant l'API Rest en PHP Symphony et l'interface utilisateur réalisée à l'aide du Framework Angular.

6 Mise en oeuvre

1 Choix techniques et outils

L'objectif est de décrire l'ensemble des choix techniques et outils ayant été adoptés pour ce projet permettant par exemple de reprendre le projet dans l'objectif d'apport d'améliorations et d'évolutions ou d'assurer son maintien.

Comme nous avons pu l'évoquer plutôt dans ce rapport nous avons choisi d'implémenter une Web API Rest à l'aide du Framework PHP Symfony et « api-platform ». Nous avons utilisé pour l'interface utilisateur le Framework JavaScript Angular permettant de produire un code modulaire. Le choix concernant l'IDE utilisé pour la phase de développement des solutions m'a été laissé. J'ai pour ma part utilisé « Visual Code » aussi bien pour le développement back-end que front-end.

Le détail sur les versions des frameworks et langages utilisés ainsi qu'un guide d'installation sont présents en annexe de ce document dans la partie « Cahier de développeur ».

2 Implémentation des solutions

Cette partie rapporte l'ensemble des solutions ayant pu être implémentées pendant la phase de développement. Elle fait également part des limites rencontrées, des risques identifiés dans la partie analyse qui sont finalement devenus des problèmes que l'on a rencontrés pendant le développement.

2.1 Architecture du code

Commençons donc par la partie architecture de code que nous avons proposé pour l'implémentation de notre partie du projet. L'architecture back-end mise en place pour la création de serveur API Rest suit la norme proposée par le frameworks PHP Symfony et l'architecture front utilise l'architecture par défaut proposée par le framework Angular. Ces choix nous ont permis de proposer à tout développeur en connaissance de frameworks utilisés la possibilité d'interagir facilement avec les différents composants mis en place aussi bien pour la partie back-end que front-end. L'architecture mise en place propose donc une grande modularité des différentes

composantes et facilite la lisibilité du code. Le détail concernant ces architectures de codage est présent en annexe dans le cahier du développeur à la section « 11.3 » et « 11.4 ».

2.2 Solutions implémentées

Pour revenir sur ce qui a été réalisé au cours de cette phase de développement, l'implémentation des solutions était divisée en 4 sprints consistants respectivement à mettre en place les points suivants :

- Création du package de données AR à destination de l'application mobile,
- Création de l'interface de gestion des annotations sur une œuvre et tâches additionnelles,
- Un plan de tests et contrôle qualité de code (surtout fonctionnelle) pour faciliter la reprise du projet,
- La rédaction du guide utilisateur, cahier de développeur et rapport de projet ;

Nous allons donc revenir point par point sur ce qui a pu être mené à bien concernant chacun de ces sprints dans la suite de cette partie.

2.2.1 Création des packages de données pour l'application mobile

L'objectif de cette tâche était de préparer coté serveur back, des packages de données compressés contenants chacun des fichiers propres à chaque site patrimonial (détail sur l'architecture de ces packages à venir) et devant être mis à disposition de l'application mobile via des services de l'API Rest. Cette tâche était prioritaire sur l'ensemble de celles m'ayant été attribuées et a même fini par occuper la quasi-totalité de la phase de développement pour ma part.

Revenons tout d'abord sur l'architecture de ces packages et leur localisation :

Chaque site patrimonial possède dans la version finale du projet deux archives à savoir une pour le mode « Debug » (Ensemble du descriptif du site : zones, œuvres, images . . . à destination des responsables de site) et une pour le mode « Release » (Seulement les éléments disponibles pour visiter (ex : œuvres publiées)) de l'application.

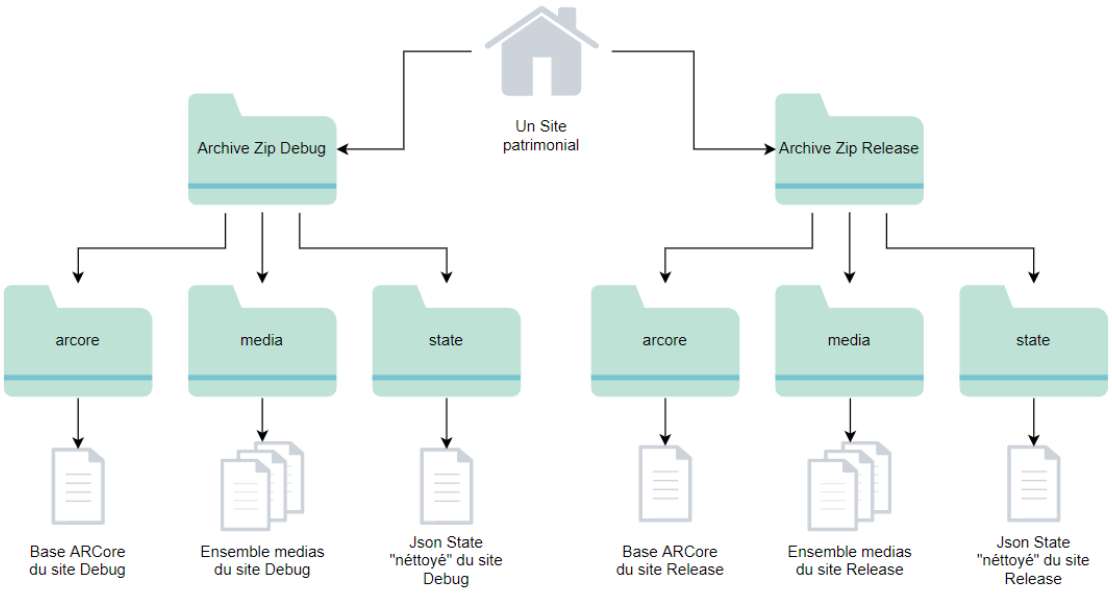


Figure 1 – *Architecture des packages de données*

Chaque archive possède donc une fois décompressée 3 sous-dossiers :

- Arcore : Contient la base de données ARCore (Images de réalité augmentée indexées) correspondant au site patrimonial dans l'état correspondant au mode de lancement spécifié pour cette archive et au fichier Json dans le dossier State.
- Media : Contient l'ensemble de médias donc vidéos, audios ou encore photos autre que les images en réalité augmentée qui caractérisent le site patrimonial dans l'état correspondant au fichier Json contenu dans le dossier state et le mode spécifié pour cette archive (pour rappel debug ou release)
- State : Contient uniquement le Json State « nettoyé » (c'est-à-dire le Json state caractérisant l'état du site contenu dans le dossier « public/state/ » du serveur ôté de toute information inutile et respectant les contraintes fixés par l'équipe de développement, contraintes sur lesquelles nous allons revenir par la suite.)

Concernant l'implémentation maintenant, il a d'abord fallu dans un premier temps déterminer comment créer une commande serveur avec PHP Symfony. La documentation PHP Symfony a rapidement répondu à mes interrogations sur ce sujet et une première commande simple en guise de prise en main de cette doctrine a été effectuée. (https://symfony.com/doc/current/console.html)

Le code permettant de déclarer la commande ainsi que son comportement est présent dans la classe php située à l'emplacement « src/Command/CreateBundleCommand.php ». C'est dans ce fichier que la commande « app :create-package » est déclarée et que son comportement selon le paramètre qui la suit est indiqué. Le détail permettant de lancer la commande de création des packages de données des bases de données ARCore est présent en annexe de ce document dans le cahier de développeur à la section « Architecture du code de création de la commande de package .

Le fichier agissant comme contrôleur entre la commande de création de package et l'accès aux différentes informations de l'application correspond à « src/Service/PackageService.php ». C'est dans la fonction « getImagesFromSiteState(KernelInterface $kernel, string $jsonSiteState, string $versionStateSite) » qu'a été créé la « machine » permettant de créer les packages de données suivant les différentes contraintes établies par l'équipe de développement sur leurs contenus.

Voici donc les contraintes devant respecter les packages de données à destination de l'application mobile :

- Les bases AR ne contiennent que des images de qualité supérieure ou égale à 75,
- La base AR contient l'ensemble des images RA que l'œuvre soit à l'état published true ou false,
- La base AR ne contient que les images RA pour une œuvre à l'état published true,
- Le dossier media contient les images du site, des zones si site publié, des annotations des creativeWorks publiés (+ simplification),
- Le dossier media contient images du site, des zones si site publié, des annotations des creativeWorks publiés (+ simplification),
- Le dossier debug contient le jsonState simplifié :
  - ☐ Liste administrators site supprimée,
  - ☐ Conservation de l'attribut validity date string ,
  - ☐ Image descriptive du site ->enlever name, isImage et quality,
  - ☐ Image descriptive d'une zone -> enlever name, isImage et quality,
  - ☐ Image descriptive d'une œuvre -> enlever name, isImage et quality,
  - ☐ Zones qui ne comportent pas d'annotations à enlever,
  - ☐ Œuvres pas d'annotations à enlever également,
  - ☐ Dans les œuvres : -> retirer le champ media, pas besoin de l'image annotée par

default (isDefault=true), enlever champ isDefault pour images annotées)

- Le dossier release contient le jsonState simplifié :
  - ☐ Liste administrators site supprimée,
  - ☐ Conservation de l'attribut validity date string ,
  - ☐ Image descriptive du site ->enlever name, isImage et quality,
  - ☐ Image descriptive d'une zone -> enlever name, isImage et quality,
  - ☐ Image descriptive d'une œuvre -> enlever name, isImage et quality,
  - ☐ Zones qui ne comportent pas d'annotations à enlever,
  - ☐ Œuvres pas d'annotations à enlever également,
  - ☐ Dans les œuvres : -> retirer le champ media, pas besoin de l'image annotée par default (isDefault=true), enlever champ isDefault pour images annotées)

Une fois la commande exécutée, les packages sont présents dans le dossier « public/package », leur nom normé de la manière suivante : « idSite_mode_versionBaseAR ».

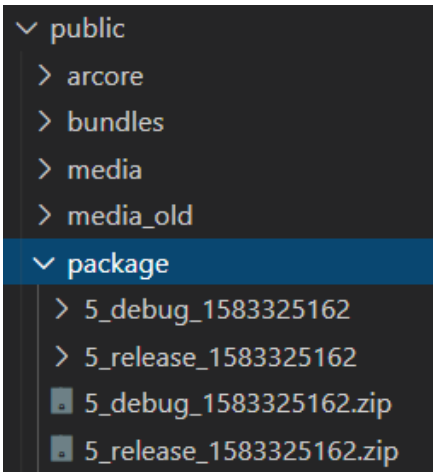


Figure 2 – *Localisation des packages sur le serveur*

La complexité de cette tâche résidait principalement dans le fait que le contenu de l'archive à fournir a beaucoup changé au cours du temps et des réunions entre membres de l'équipe de développement qui avaient lieu régulièrement. Le risque de ne pas pouvoir finir la tâche de création de l'interface d'administration des annotations a donc rapidement été évoqué ce qui a permis de convenir avec l'encadrant du nouvel impératif de la phase de développement à savoir finir la tâche de création de package au détriment de la partie interface devant être implémentée.

2.2.2 Création de l'interface de gestion des annotations sur une oeuvre

Je suis tout de même parvenu à effectuer la création de l'entité permettant de stocker les annotations dans la base de données et à mettre cette nouvelle entité en interaction avec celles déjà mis en place par le passé. Il est possible de voir cette intégration au modèle dans la figure représentant le modèle de données final dans la section « 5.4 ». Des services assurant la gestion de la nouvelle entité depuis l'API Rest ont également été créés me permettant de participer à la conception de cette interface de gestion d'annotation. Je n'ai néanmoins pas vu aboutir cette gestion d'annotations d'une œuvre. En effet, la création de l'interface utilisateur dédiée à cette partie n'a pas pu être implémentée dû aux délais de réalisation courts et à une re priorisation des tâches à effectuer d'ici la fin de l'année en accord avec l'encadrant.

```php
class Annotation
{
    /**
     * @ORM\Id()
     * @ORM\GeneratedValue()
     * @ORM\Column(type="integer")
     * @Groups({"annotation_read"})
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=255)
     * @Groups({"annotation_read"})
     */
    private $name;

    /**
     * @ORM\Column(type="text", nullable=true)
     * @Groups({"annotation_read"})
```

Figure 3 – *Extrait de l'implémentation de l'entité annotation*

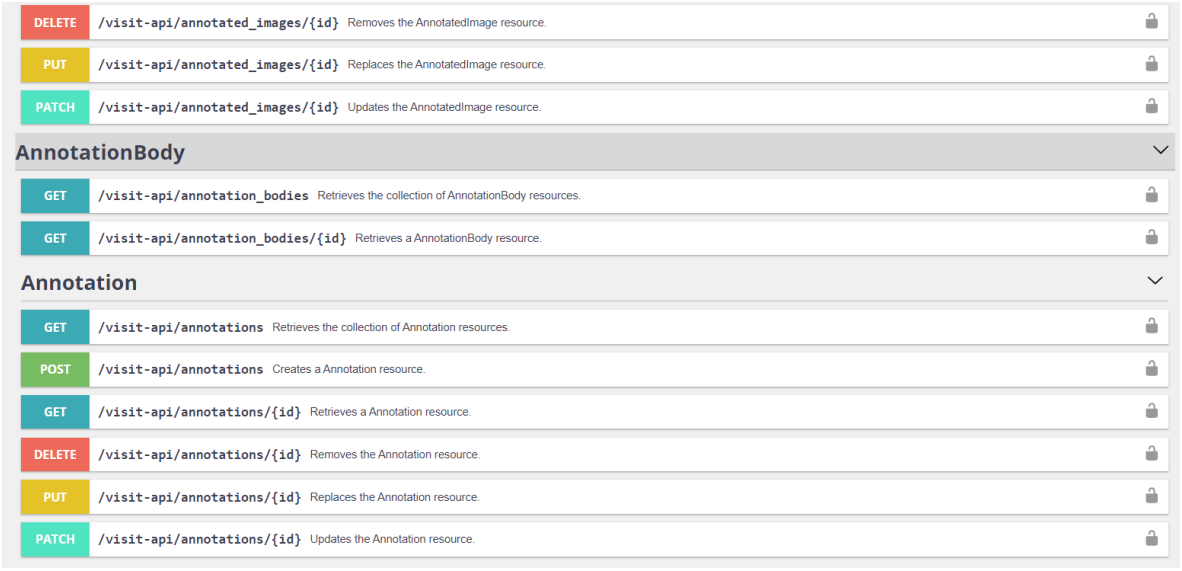


Figure 4 – *Services API Annotation implémentés*

Pour information, il était pendant la partie analyse du projet question d'un choix à effectuer concernant la technologie à utiliser pour le placement des balises et ancres représentant les annotations dans l'interface correspondant. Deux solutions avaient été retenues à savoir l'utilisation de SVG ou de Canvas. Finalement la solution Canvas avait été retenue. Nous avons néanmoins effectué un retour en arrière après de premières implémentations tests de la solution utilisant SVG, qui finalement correspond mieux à ce que nous souhaitons mettre en place pour ce projet.

2.2.3 Plan de tests et mesures de qualité de code

Les tests effectués ont principalement été fonctionnels. Après avoir suivi le plan de tests qui est disponible en annexe de ce document sous la forme d'un cahier de recette dans la partie « Tests et Qualité de Codage », section « cahier de recette »., nous avons pu vérifier si l'ensemble des contraintes caractérisant la création de package étaient remplies. Ces tests permettaient par la même occasion de contrôler le bon fonctionnement de l'interface utilisateur concernant les fonctionnalités déjà implémentées. Nous avons alors pu valider les solutions mis en place, le détail est présent dans la partie « Tests et Qualité de Codage » de ce document.

2.2.4 Rédaction des documents d'accompagnement

Enfin un guide utilisateur (section 14) a été fourni au même titre que ce rapport. La mise en place du projet ainsi que son historique sont expliqués dans ce rapport qui contient également une documentation développeur (section 11) par la même occasion.

2.2.5 Fonctionnalités globales de l'interface d'administration des sites patrimoniaux

L'interface utilisateur constituant le lien entre l'utilisateur et les données qui sont stockées sur notre site web est à ce jour dans une version fonctionnelle grâce au travail d'un autre membre de l'équipe de développement (Olivier Roussey).

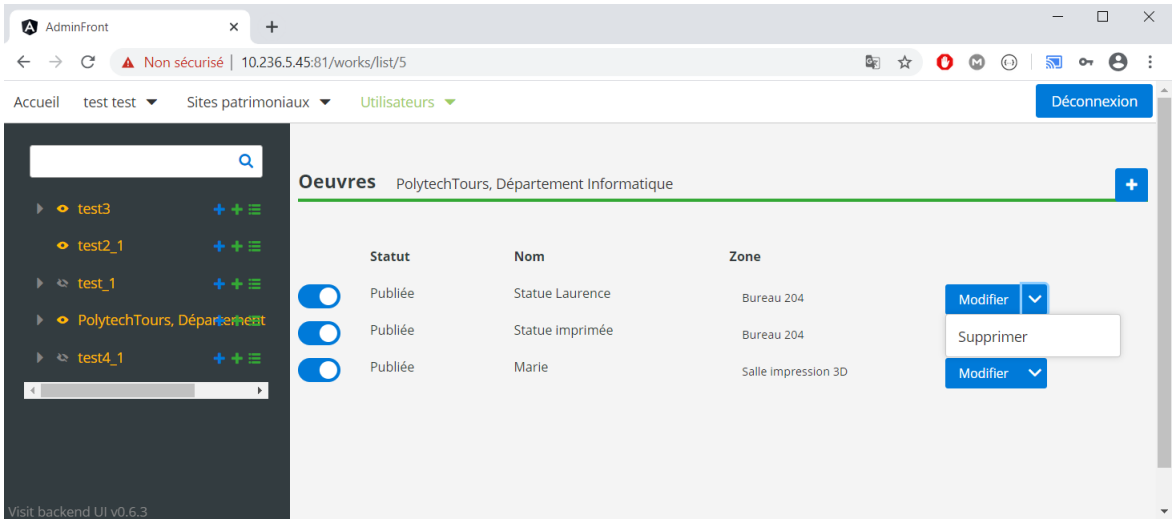


Figure 5 – *Interface Utilisateur dans sa version actuelle*

Le guide utilisateur évoqué dans la section précédente a été réalisé à partir de cette version. Les documentations développeur ainsi que le présent rapport ont été rédigés en connaissance des personnes reprenant par la suite le fruit de mon travail.

Les points constituants les 4 tâches principales répertoriées dans le Trello sinon dans le planning prévisionnel ont été remplies sinon revues pour rentrer dans les délais de réalisation. Leurs résultats sont notamment visibles dans les 2 sous-parties décrivant les deux tâches principales (« Création des packages de données pour l'application mobile » et « Création de l'interface de gestion des annotations sur une œuvre »), un guide utilisateur (section 14) ou le rapport de tests (section Tests et Qualité de Codage).

3 Analyse des résultats

Concernant les résultats obtenus suite à la phase de développement, ces derniers ont été testés via principalement des contrôles fonctionnels réalisés lors de plusieurs lancements de l'application et exécutions de la commande de création de package. Ces différents résultats sont notamment visibles à travers les captures et explications qui composent le guide utilisateur. Les mesures de qualités et évaluation de performance sont consultable dans la section Tests et Qualité de Codage » en annexe de ce rapport.

7 Conclusion

Les délais de réalisation de ce projet sont fixés par la durée de l'année universitaire. Le projet en lui-même est découpé en 2 phases distinctes matérialisées par les deux semestres qui composent une année. La répartition du projet en termes de rendu suit donc le modèle suivant :

• Semestre 9 : Recherches de solutions et spécifications du projet,

• Semestre 10 : Mise en œuvre et développement de solutions, réalisation de tests, rédaction de documents utilisateurs et livraison de la solution.

1 Bilan du Semestre 9

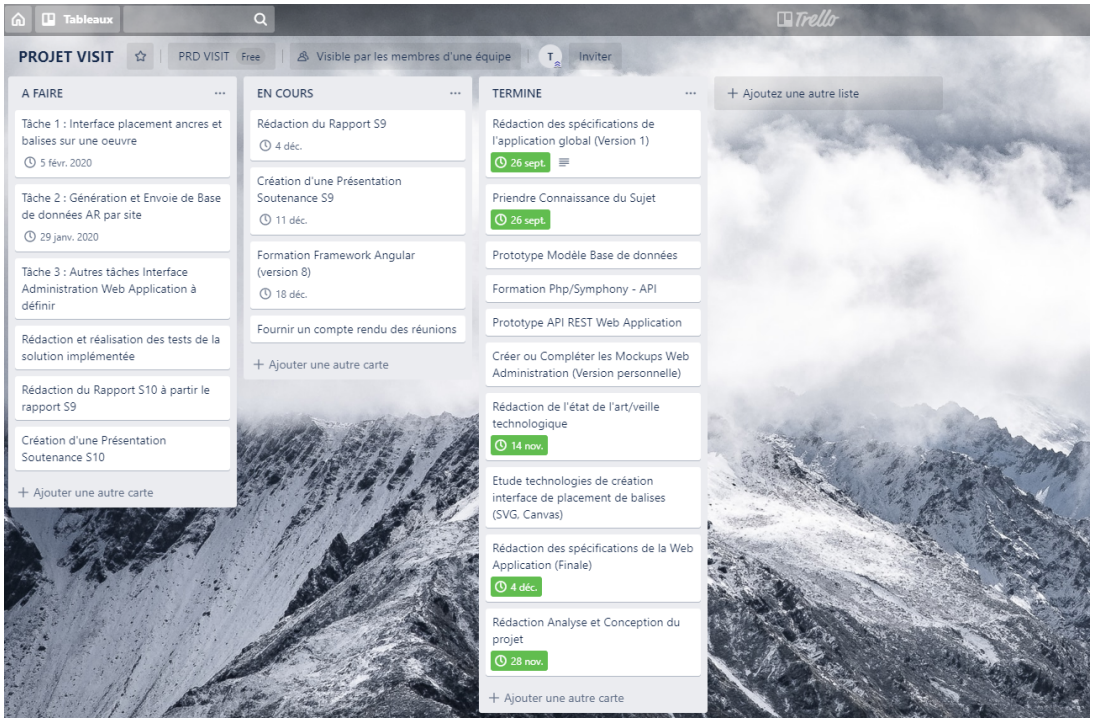


Figure 1 – *Liste des tâches du S9 sur Trello*

La période d'analyse que constitue le S9 a tout d'abord commencé par l'analyse des besoins permettant ainsi d'orienter la recherche de solutions répondants aux problématiques évoquées.

Un processus itératif commençant par une phase de recherche puis de « brainstorming » avec les membres de l'équipe a été réalisé afin de faire surgir de nouvelles idées ou problématiques à résoudre. C'est également durant cette période du semestre qu'a eu lieu la formation aux technologies adoptées par l'équipe pour ce projet et donc la réalisation de prototypes (notamment pour la base de données et l'API).

Des tâches ont pu être affectées à chacun et la réalisation de documents préambules à la phase de développement a pu être réalisée (état de l'art, cahier des spécifications) comme peut le témoigner le « Trello » précédent.

En suivant le schéma précédent on peut dire que les tâches suivantes sont finies :

- Compréhension du sujet ainsi que ces enjeux,
- Analyse des besoins clients, notamment à travers la rédaction d'une première version du cahier des spécifications (concernant l'application dans son ensemble),
- Identifications des problématiques qui découlent du sujet et commencer à réfléchir à des solutions,
- Réalisation des prototypes (Base de données et API) afin de proposer une approche et de se former aux technologies utilisées dans le projet,
- Rédaction du cahier des spécification, état de l'art/veille technologique et plan de tests envisagés du projet,
- Rédaction du rapport et préparation de la soutenance du S9

Le projet n'a pour le moment pas de retard et reste pour le moment en accord avec le diagramme prévisionnel fourni dans la partie planification. Il sera cependant important d'implémenter relativement rapidement la tâche correspondant à la création et l'envoie des bases de données AR à l'application Android étant donné la forte interdépendance des modules applicatifs Web et mobile Android concernant cette partie.

Concernant les tâches restantes, il s'agira de la partie implémentation et mise en place des solutions à savoir :

- Achever rapidement la formation aux technologies utilisées pour ce projet,
- Tâche 1 : Interface de placement d'ancres et de balises sur une œuvre,
- Tâche 2 : Génération et Envoie de Base de données AR par site à l'App Mobile, (Prioritaire)

Chacune de ces tâches se décline en 3 fonctionnalités à implémenter qui sont précisées dans la section spécifications fonctionnelles. Par ailleurs, de nouvelles tâches avec une priorité moindre que les précédentes pourront m'être assignées comme il l'a été notifié dans la section « Planification du projet » présente en annexe.

2 Planning du semestre 10

Une estimation du temps de développement a donc été imaginée afin de mettre en place les différents sprints qui composeront la partie réalisation du projet dans le diagramme prévisionnel. Lesdits sprint sont décrit dans la section planification et prennent en compte le possible ajout de fonctionnalités secondaires à réaliser.

Conformément au plan établi, le prochain lot de sprint consiste à :

- Achever la formation au Framework Angular particulièrement pour la partie Canvas,
- La mise en place des fonctions relatives à la Tâche 2 (en reprenant les notations de la partie précédente) de manière prioritaire,
- La mise en place de celles concernant la Tâche 1,

Les sprints suivants seront réservés à la réalisation des autres tâches attribuées, la rédaction d'un document de test recessant les tests effectués et le rapport final du projet. La description des fonctionnalités peut être consultée en annexe de ce document à la section correspondante.

3 Bilan qualité

La phase de développement une fois terminée, il est maintenant intéressant de tirer un premier bilan qualité de ce qui a été réalisé pour ce projet. Des tests de vérification et validation ont été mis en place à la fin de la phase de développement comme nous l'avions prévu initialement dans le planning prévisionnel (« Sprint 3 »).

Concernant la création des packages de données, les tests effectués ne procèdent qu'à un contrôle de la présence des fichiers souhaités dans une archive type de manière automatique. L'automatisation des tests n'a pas pu aller jusqu'à un contrôle du contenu de chaque fichier compris dans les archives faute de temps. Des tests fonctionnels ont cependant pu être menés aussi bien sur la création de package de données que sur les fonctionnalités disponibles dans l'interface d'administration des sites patrimoniaux. Lesdits tests et contrôles qualité ont fait l'objet d'un cahier de recette et rapport « Tests et Qualité de Codage » disponibles en annexe de ce document.

Ces tests et contrôles effectués permettent de nous assurer du bon fonctionnement de la solution permettant la création de packages de données ARCore pour l'application mobile. De plus, le livrable final correspondant aux solutions implémentées pendant le S10 (gestion packages de données, entités et services relatifs aux annotations) a déjà été intégré à la solution finale du site web d'administration. Le site suite à l'intégration de ces changements reste fonctionnel et les solutions proposées semblent correspondre aux attentes énoncées au S9.

4 Bilan gestion de projet

Revenons maintenant sur le déroulement du projet dans sa globalité, à savoir les périodes du S9 et du S10 confondues. Pour rappel, le S9 s'est déroulé sans encombre, les réunions permettant de faire évoluer une vision commune des solutions à implémenter pour répondre aux besoins du projet. A la suite de cette réflexion, des spécifications avaient été formulées et des tâches, principalement 2 m'avaient été attribuées. Ces dernières constituants le contenu des Sprints 1 et 2 de la partie planification.

Il était alors question dans le planning du S10 de finir de façon prioritaire la tâche 2 consistant pour rappel à la création de packages de données sur le serveur contenant entre autres les bases d'images ARCore. Des risques avaient été identifiées lors de la période d'analyse concernant la phase développement du S10 et plus généralement la gestion du projet dans sa globalité. Lesdits risques était principalement liés à l'apprentissage de nouvelles techniques et nouveaux langages de développement ainsi que les délais de réalisation imposés qui restaient relativement courts. C'est en connaissance de ces risques, que j'avais organisé la planification des tâches que je devais effectuer durant la phase de développement.

Une fois cette phase du S10 terminée je peux affirmer que les risques identifiés durant le S9 sont finalement devenus des obstacles sinon des éléments ayant considérablement ralentis l'implémentation des solutions. De plus d'autres problèmes n'ayant pas auparavant été identifiés comme « risques » sont apparus. Je pense principalement aux problèmes d'intégration de solution suivant l'environnement d'installation ciblé. L'ensemble de ces éléments sont décrits et expliqués dans un tableau disponible en annexe, section «12.7.1 Problèmes rencontrés ».

Néanmoins ces problèmes et retards ont pu être endigués à l'aide de réunion et feedbacks réguliers avec l'équipe de développement et l'encadrant. Ces rencontres ont permis de re prioriser les tâches à effectuer permettant ainsi d'aboutir à la mise en place de solutions finales correspondants aux attentes de départs. Le planning prévisionnel a donc donné suite à des

modifications permettant de réaliser les tâches prévues dans les temps (voir Gantt final, section 12).

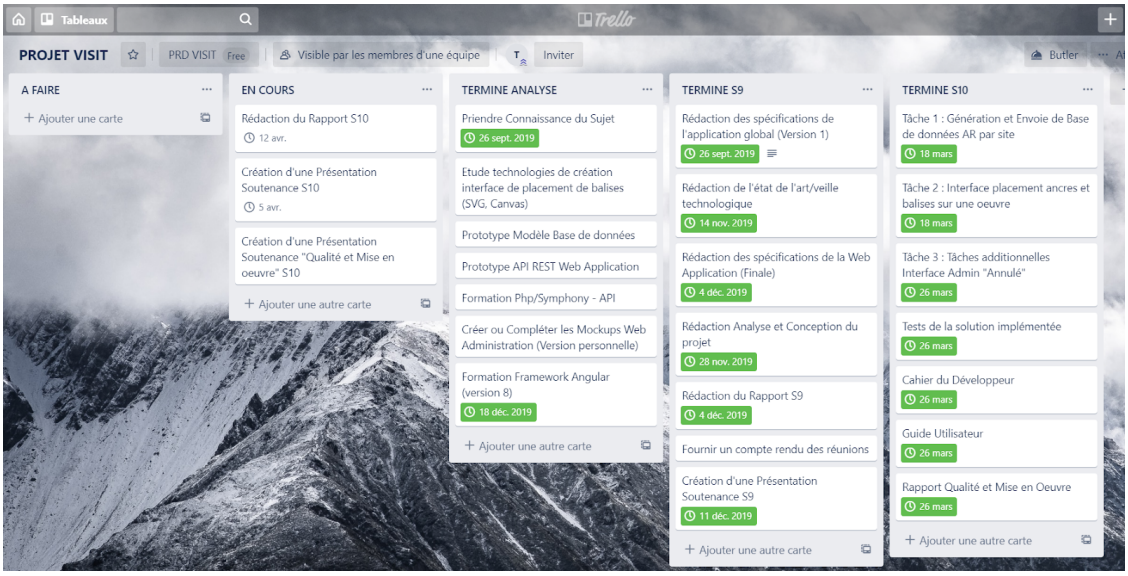


Figure 2 – *Liste des tâches du S9 et S10 sur Trello*

5 Bilan personnel

La réalisation d'une application permettant de mettre en place des visites de sites patrimoniaux à l'aide de la réalité augmentée m'a permis d'explorer dans sa globalité des techniques et outils que je n'avais jamais utilisés auparavant (Framework Php Symphony, Angular ou encore Arcore).

Mon intervention s'est plutôt située sur la partie web application et envoi de données entre la base de données administrée depuis le site web et l'application mobile chargée de retranscrire ces données et de les rendre disponibles aux visiteurs.

Je pense avoir fait preuve d'un bon sens d'adaptation face aux différents problèmes posés par ce projet et solutions qu'il fallait apporter par la suite. J'ai également pu travailler en coopération avec un autre développeur ce qui induit également des adaptations aux niveaux des interactions sur le projet et interdépendances qu'il en découle. Enfin la découverte de nouvelles technologies, procédés et frameworks ne pas empêcher de mettre en place une architecture et des méthodes facilitant la maintenance et reprise du projet.

Néanmoins de potentiels axes d'améliorations pourraient être dégagés de cette expérience. Selon moi, ces derniers sont principalement liés au manque d'expérience dans un projet de cette envergure. L'anticipation des problèmes à venir, l'adoption des bonnes pratiques de codage par exemple ou encore une meilleure gestion du temps fixé sont pour moi des pistes à améliorer dans la façon de mener de futurs projets.

Enfin si je devais résumer mon expérience concernant ce projet, je la qualifierais comme étant très formateur pour la suite de mon parcours.

Annexes

A Description des interfaces externes de l'application

1 Interfaces matériel/logiciel

Le projet global se distingue en deux modules principaux à savoir une web plateforme ainsi qu'une application mobile comme nous avons pu l'évoquer précédemment. Cette partie du projet est consacré à la partie Web API et Web Plateforme, il en découle par conséquent la mise en place d'une procédure de développement et un matériel orienté Web.

Un utilisateur, indépendamment des droits qu'il possède dans l'application devra être muni d'un ordinateur personnel ou d'un autre support lui permettant de se rendre sur l'adresse de la plateforme via un navigateur Web. Pour rappel, l'application Web communiquera avec une API REST, tout comme l'application Android afin d'obtenir les informations stockées dans la base de données.

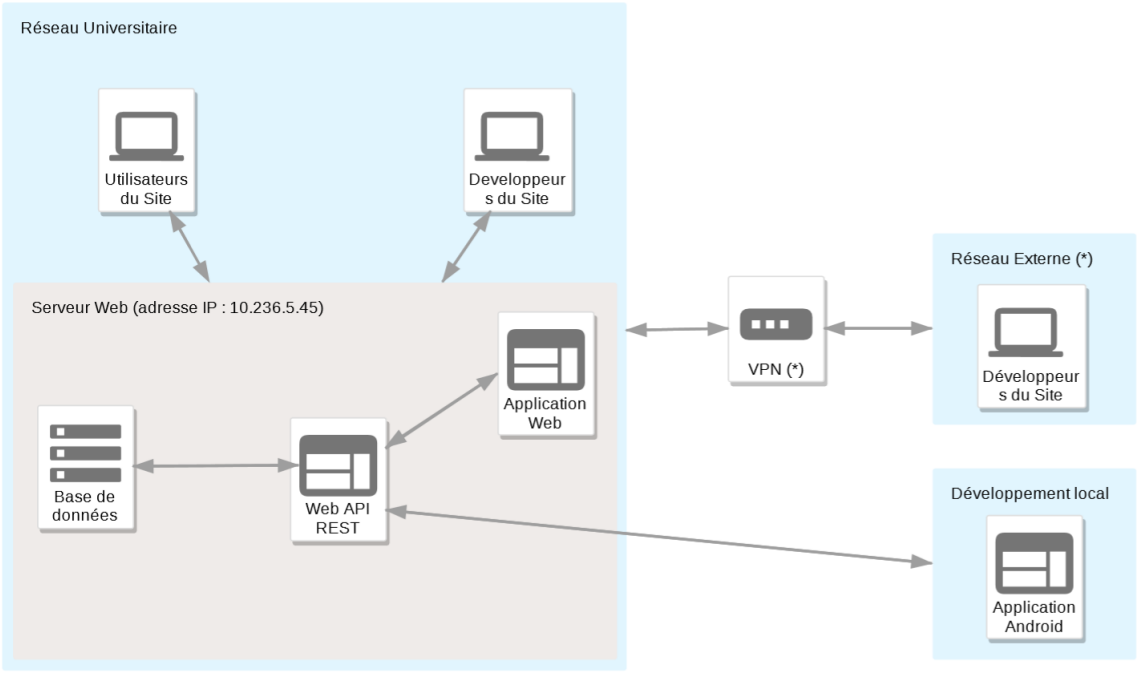


Figure 1 – *Diagramme de Déploiement*

Comme le diagramme peut le suggérer, l'ensemble des composantes de l'application Web seront stockées sur un serveur de l'université (adresse IP 10.236.5.45) pendant son développement. Il serait également possible de bénéficier d'un accès VPN pour permettre aux développeurs de contacter le serveur depuis un réseau externe mais le recours à cette solution n'a pas été confirmé (*). La plateforme Web ainsi que la base de données et la Web API resteront donc uniquement accessibles depuis le réseau universitaire pour le moment.

2 Interfaces homme/machine

L'objectif est de prouver la nécessité d'utiliser cette application Web pour la gestion des visites touristique. Cette dernière doit donc proposer une interface qui soit la plus efficace et satisfaisante possible. On entend par ces deux critères la recherche d'une interface efficiente permettant d'obtenir le résultat voulu tout en étant plaisante (subjectif). L'interface de l'application finale sera réalisée à l'aide du Framework JavaScript Angular permettant de réaliser une implémentation modulaire de la solution adoptée. A noter que la technologie de reconnaissance d'image retenue pour ce projet est « ARCore ». Nous reviendrons dans le rôle de cette technique de modélisation d'image 3D dans la partie consacrée à l'interface « Responsable » à développer.

Pour le moment aucun besoin ne concerne la nécessité de produire une interface « intelligente » pour la partie web plateforme, cet aspect pourra néanmoins faire l'objet d'une amélioration de l'application si la demande est émise dans un futur proche.

Concernant la gestion d'erreurs, l'idée est d'accompagner l'utilisateur dans sa découverte de l'application visant à lui apprendre rapidement la manière la plus efficace de l'utiliser. Dans cette optique, il sera consacré une partie du développement à la réalisation d'aides utilisateur, matérialisées pour la plupart sous forme de Pop-ups.

Comme nous avons pu le voir dans les parties précédentes, cette plateforme est amenée à être parcouru par 3 acteurs ayant accès à des fonctionnalités différentes de l'application selon les droits qui leur sont accordés. L'existence de profil utilisateur différents se matérialise par la création de diverses vues interface. L'ensemble des mockups suivants ont été réalisées à l'aide de l'outil « Balsamiq ». Ils permettent d'entrevoir la disposition des différentes composantes et l'accès aux fonctionnalités constituants l'application Web.

Remarque : Seul certains mockups sont présents dans cette partie, un échantillon plus complet pourra être ajouté en annexe.

2.1 Interface Commune

Dans un premier temps chaque utilisateur de la plateforme souhaitant accéder au contenu patrimonial devra se rendre sur le site de l'application à l'aide d'un navigateur. Il aura alors la possibilité de s'authentifier, ce qui constitut le premier repère permettant à l'application d'adapter ses vues aux fonctionnalités accordées au profil utilisateur connecté.

Les mockups annotés d'un astérisque ont été réalisés par Olivier Roussey.

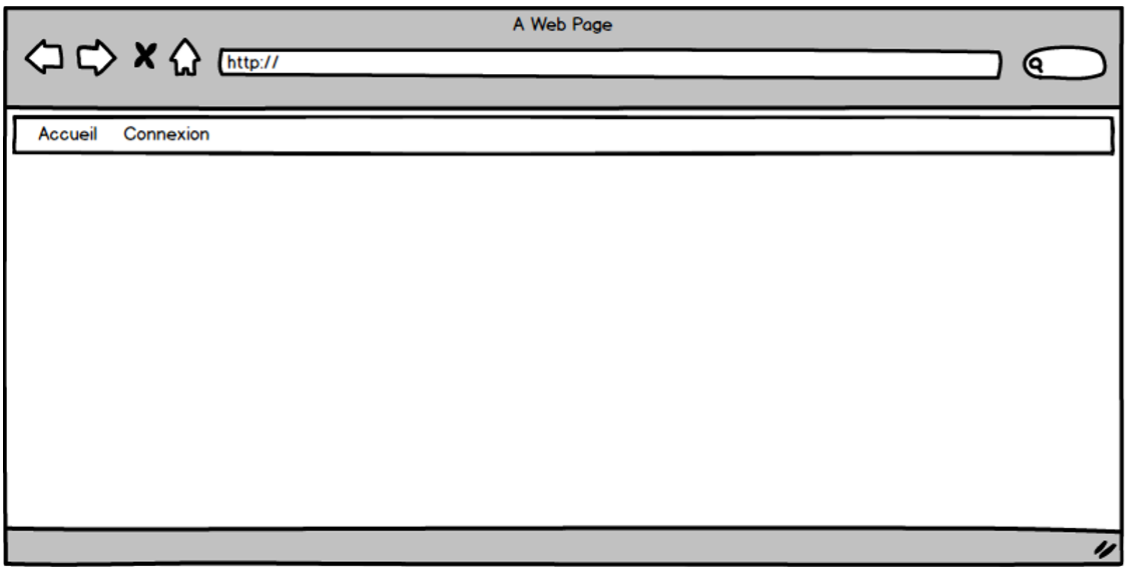


Figure 2 – *Interface de Connexion**

2.2 Interface Responsable

Une fois connecté, un responsable a accès à l'ensemble des informations qui concernent le site dont il est en charge. Il a alors la possibilité d'ajouter, modifier ou supprimer tout ce qui concerne les zones, auteurs ou encore les œuvres appartenant à son site.

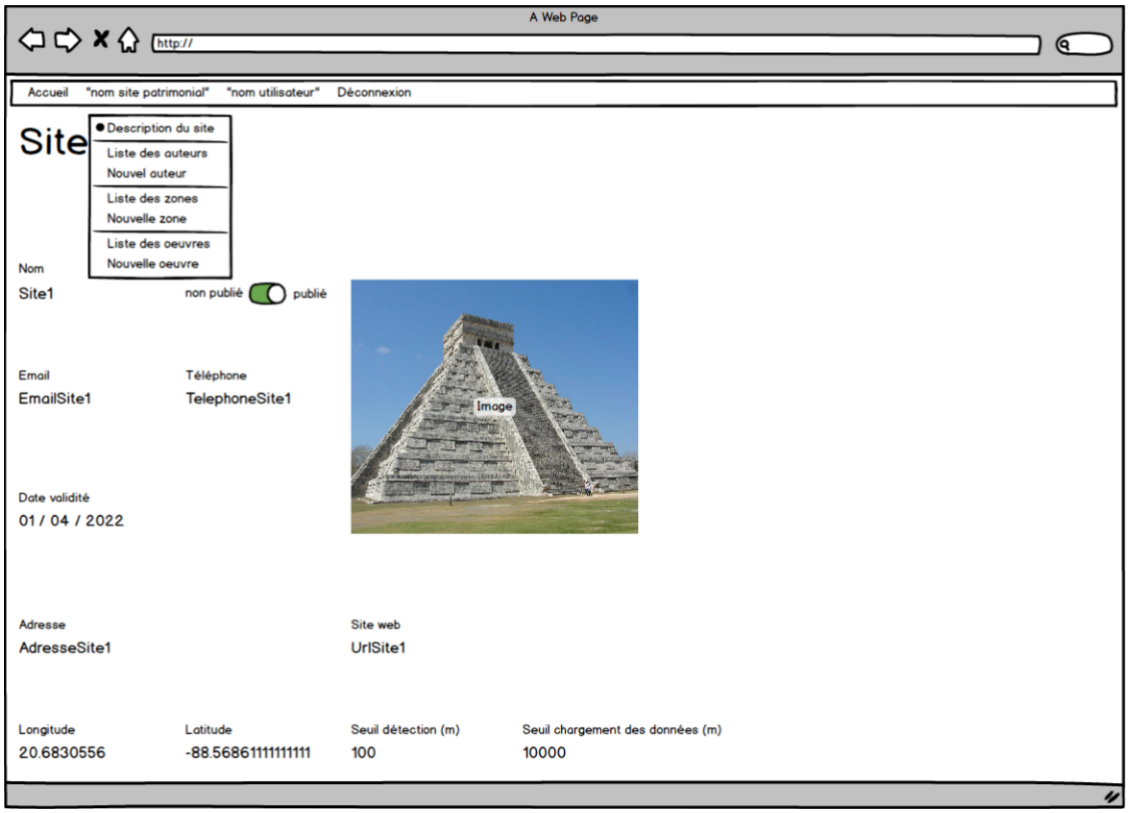


Figure 3 – *Interface de Description du site (Responsable)**

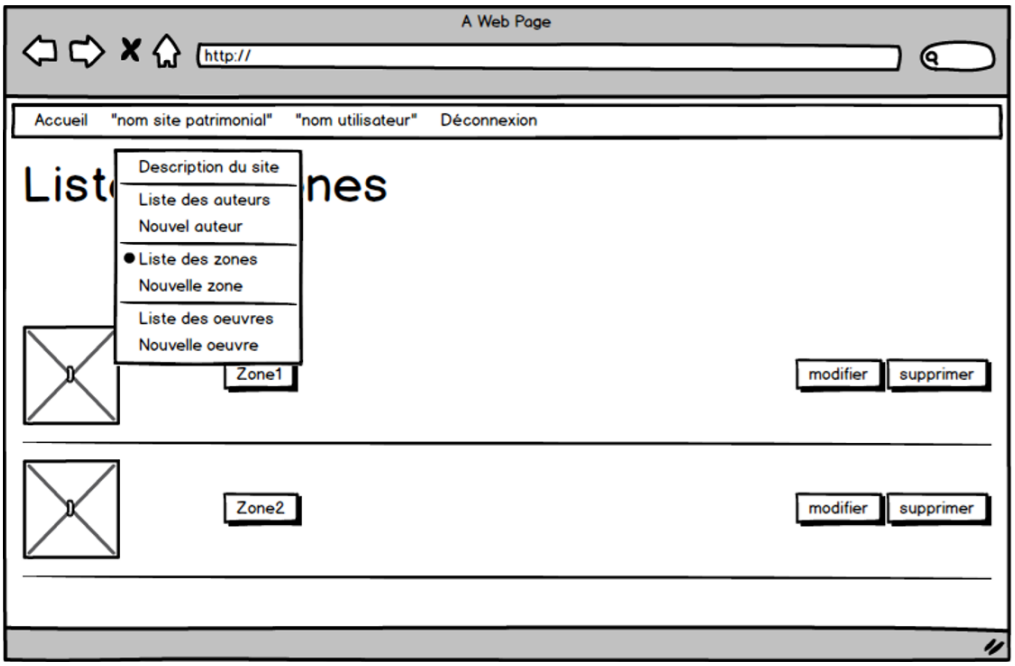


Figure 4 – *Interface de Description de la liste des zones d'un site (Responsable)**

Il a donc été décidé d'utiliser la technologie AR Core pour matérialiser les œuvres en 3D à l'aide de l'appareil photo mobile. Il est donc nécessaire de proposer un système permettant à l'utilisateur d'obtenir des informations sur une œuvre par exemple sous la forme de bulles textuelles activables après activation sur un point défini de l'image. Un aspect intéressant de l'interface responsable concerne donc la mise en place de ces balises et ancres sur une image représentant une œuvre afin de mettre en place les informations nécessaires aux visiteurs.

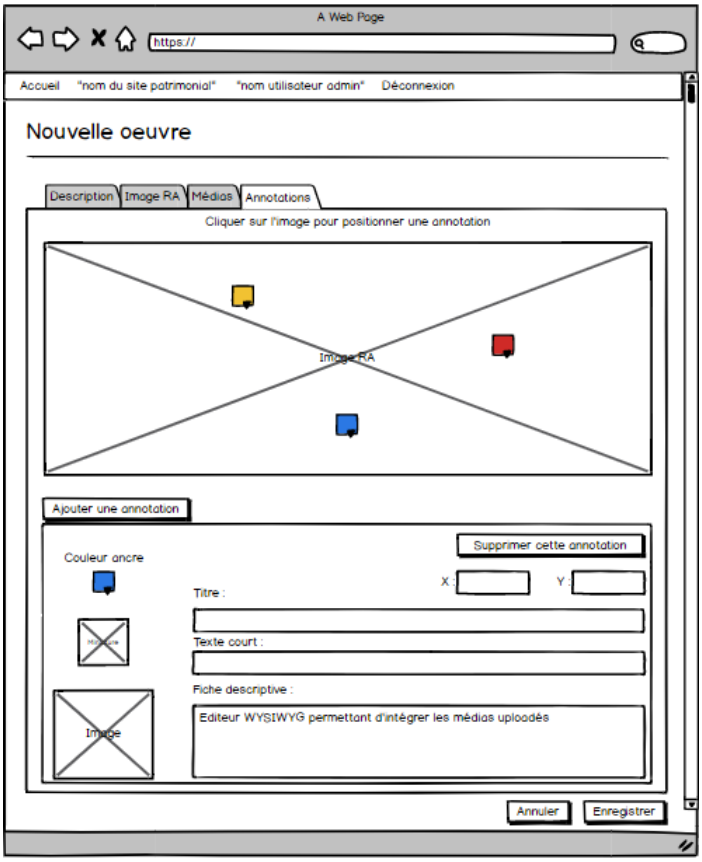


Figure 5 – *Interface de Visualisation d'ajout d'œuvres ancres/balises (Responsable)*

Ledit système de placement pourrait se matérialiser de la manière précédente à l'aide des technologies SVG ou Canevas.

2.3 Interface Administrateur

La partie Interface administrateur lui permet d'exécuter l'ensemble des fonctionnalités accordées aux utilisateurs possédant des droits moindres à savoir les visiteurs et responsables. Il y a toutefois une partie réservée à ce dernier pour le moment à savoir la possibilité d'effectuer (de confirmer/refuser) les inscriptions de nouveaux responsables à la plateforme. Il peut également visualiser l'ensemble des sites recensés sur la plateforme ainsi que ce qu'ils contiennent et a la capacité d'agir sur ce contenu.

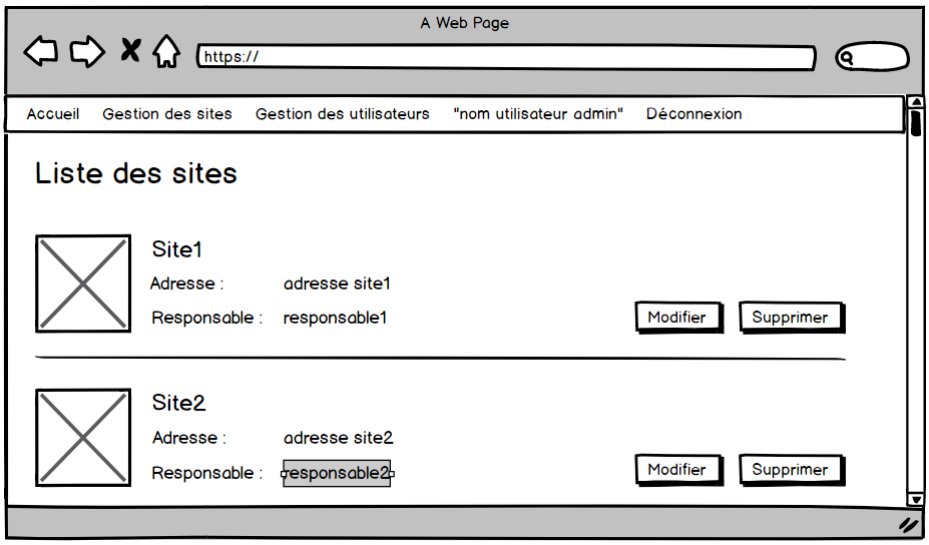


Figure 6 – *Interface de Visualisation des sites (Administrateur)**

3 Interfaces logiciel/logiciel

Pour rappel, l'application Web à savoir la plateforme d'administration du contenu des sites patrimoniaux se situe sur un serveur de Polytech où est également présent la base de données ainsi que la Web API REST. La base de données est donc en lien direct avec l'application Web étant sur le même serveur et nécessite pour le moment des droits « administrateur » pour pouvoir être contactée. La communication entre la base de données et la plateforme s'effectue par l'intermédiaire de la Web API située également sur le serveur. L'application Android sera également amenée à communiquer avec la base de données et aura donc également recours à l'API pour obtenir les données dont elle a besoin.

Pour commencer la mise en place d'une solution et afin de contrôler l'impact des flux sur le fonctionnement de l'application un cas concret « réel » sera matérialisé. Il a été fixé que le système global de l'application aussi bien mobile que web devait être fonctionnel en considérant que le système comporte pour le moment 20 œuvres et 20 sites différents.

B Spécifications Fonctionnelles

Les spécifications fonctionnelles présentes dans ce document sont séparées en 2 catégories :

- Les fonctionnalités qui m'ont été officiellement attribuées.
- Les fonctionnalités globales de l'interface d'administration : Je pourrai en effet être amené à contribuer à la réalisation de ces fonctionnalités dans le futur (ces tâches ne me sont pas attribuées pour le moment).

1 Fonctionnalités attribuées

1.1 Gestion des annotations sur les images AR

Cette partie est consacrée à la tâche de création d'une interface permettant d'ajouter des annotations (ancres, balises, contenu) à une image AR afin de proposer aux responsables d'organiser la présentation des informations visiteurs comme ils le souhaitent.

1. Définition de la Fonction "AJOUTER UNE ANNOTATION"

- Nom de la fonction : ajouterAnnotation()
- Priorité : Primordiale
- Description : Permet d'ajouter une annotation (contenant des informations sur l'œuvre) reliée par l'intermédiaire d'une balise à une ancre qui est placée à un endroit précis sur l'image AR.
- Préconditions :
  - _ Être dans le menu ajouter Œuvre,
  - _ Avoir une image de l'œuvre chargée dans l'interface,
  - _ Droits : Responsable et Administrateur,
- Entrées : Coordonnées (x, y) à saisir, ainsi que les autres informations d'une annotation(titre, description) soit l'objet annotation ;
- Sorties : Annotation ajoutée sur l'image.
- Postconditions : L'annotation a bien été ajoutée sur l'œuvre.

2. Définition de la Fonction “MODIFIER UNE ANNOTATION”

- Nom de la fonction : modifierAnnotation()
- Priorité : Primordiale
- Description : Permet de modifier le contenu, la position, la couleur... ou autres caractéristiques d’une annotation placée sur une image.
- Préconditions :
  - _ Être dans le menu ajouter Œuvre,
  - _ Avoir une image de l’œuvre chargée dans l’interface,
  - _ Avoir sélectionné une annotation sur l’image,
  - _ Droits : Responsable et Administrateur,
- Entrées : Une annotation déjà existante.
- Sorties : Annotation modifiée et intégrée à l’image
- Postconditions : L’annotation a bien été modifiée sur l’œuvre

3. Définition de la Fonction “SUPPRIMER UNE ANNOTATION”

- Nom de la fonction : supprimerAnnotation()
- Priorité : Primordiale
- Description : Permet de supprimer une annotation d’une image AR. La suppression s’effectue sur l’objet annotation au complet c’est-à-dire le contenu et la position.
- Préconditions :
  - _ Être dans le menu ajouter Œuvre,
  - _ Avoir une image de l’œuvre chargée dans l’interface,
  - _ Avoir sélectionné une annotation sur l’image,
  - _ Droits : Responsable et Administrateur,
- Entrées : Une annotation déjà existante.
- Sorties : Annotation supprimée
- Postconditions : L’annotation a bien été supprimée sur l’œuvre

1.2 Génération et envoie de la base de données AR à l’application Android

La génération d’une base de données AR permettant de stocker les images des œuvres afin de les communiquer à l’application Android constitue la seconde tâche qui m’ait été attribuée pour le moment. Une découpe en 3 fonctionnalités de cette tâche a été effectuée permettant de faire ressortir les fonctionnalités à développer suivantes.

1. Définition de la Fonction “EVALUATION QUALITE IMAGE”

- Nom de la fonction : evaluationImage()
- Priorité : Primordiale
- Description : Cette fonction permet de contrôler la qualité d’une image donnée en paramètre afin d’en ressortir son indice qualité. Cette indication permet d’établir si une image donnée pourra être interprétée correctement par la technologie AR Core et être modélisée par la suite.
- Préconditions :
  - _ Une image AR doit être prise en photo,
  - _ Droits : Responsable et Administrateur,
- Entrées : Image AR ;
- Sorties : Score de qualité de l’image (nombre entre 0 et 100)
- Postconditions : Le score qualité a bien été enregistré dans la base de données si on conserve l’image.

2. Définition de la Fonction “CREATION DE LA BASE DE DONNEES AR”

- Nom de la fonction : creerBaseAR()
- Priorité : Primordiale
- Description : Fonction permettant de créer un dossier d’indexation des images AR Core des œuvres par site et par type (« Debug » et « Release »). Un fichier de base AR sera généré par site et par type d’utilisation (« Debug » et Release). Cette fonctionnalité sera relancée à chaque fois qu’une œuvre et ajoutée à un site et un type d’utilisation donné de façon à conserver un dossier d’indexation à jour.
- Préconditions :
  - _ Images ont une évaluation supérieure à 75/100,
  - _ Droits : : Générer automatiquement par le système à la publication d’œuvre par l’Administrateur ou un Responsable,
- Entrées : Chemin du dossier d’images pour un site
- Sorties : Chemin de la base AR à communiquer vers l’application
- Postconditions : La base AR a bien été générée

3. Définition de la Fonction “ENVOI DE LA BASE DE DONNEES AR ”

- Nom de la fonction : envoiBaseAR()
- Priorité : Primordiale
- Description : L’application Android contactera automatiquement l’API afin de bénéficier du fichier de base AR correspondant à un site et un type d’utilisation communiqués via la requête. Cette fonctionnalité vérifiera dans un premier temps si un fichier de base AR existe pour le site et le type d’utilisation demandés puis enverra le fichier correspondant. Si le fichier n’existe pas ou s’il a été supprimé, cette fonctionnalité est également chargée de relancer la création du fichier AR voulu avant de l’envoyer.
- Préconditions :
  - _ Une requête est émise via l’API par un utilisateur de l’application,
  - _ La base AR demandé existe sinon la recréer à l’aide de la fonctionnalité précédente,
  - _ Droits : Générer automatiquement à l’appel du service correspondant à l’envoie de la base par un utilisateur (Pas de droits particuliers),
- Entrées : Un identifiant de la base de données du site et du type (œuvres «Debug ou « Release ») souhaités par l’utilisateur
- Sorties : Base AR envoyée à l’utilisateur
- Postconditions : la base a bien été envoyée

2 Fonctionnalités Interface globale

Compte tenu des délais de réalisation fixes, il a été décidé de m’attribuer les fonctionnalités précédentes pour le moment. De potentiels nouvelles tâches me seront attribuées au cours du développement suivant l’avancement du projet et la nécessité de soutenir un module de développement en particulier. Je pourrai essentiellement être amené à interagir sur les services proposés dans la Web API sinon directement sur la plateforme d’administration en développant certaines fonctionnalités interface. La liste de fonctionnalités suivantes constitue alors un aperçu des tâches que je pourrai être amené à effectuer dans le futur. Dans cette partie le caractère prioritaire des fonctionnalités ci-dessous a été fixé selon la probabilité d’apporter une contribution personnelle à leur développement et non pas en rapport à leur importance dans la mise en place de l’application.

2.1 Gestion authentification

1. Définition de la Fonction “SE CONNECTER”

- Nom de la fonction : login()
- Priorité : Secondaire - Facultative
- Description : Permet à un utilisateur de s’authentifier et de pouvoir accéder aux fonctionnalités propres au droits accordés au profil connecté.
- Préconditions :
  - _ Informations de connexion (login, password) remplies correctement,
  - _ Permet de se connecter à sa session et de pouvoir effectuer les actions liées aux droits attribués au compte,
  - _ Droits : Tous utilisateurs,
- Entrées : login, password
- Sorties : Session
- Postconditions : La session a bien été créée

2. Définition de la Fonction “SE DECONNECTER”

- Nom de la fonction : logout()
- Priorité : Secondaire - Facultative
- Description : Permet à un utilisateur de se déconnecter
- Préconditions : Membre déjà connecté,
- Entrées : Session
- Sorties : Aucune
- Postconditions : Le membre est bien déconnecté de sa session

2.2 Gestion des utilisateurs

1. Définition de la Fonction “AJOUTER UN RESPONSABLE”

- Nom de la fonction : ajouterResponsable()
- Priorité : Secondaire - Facultative
- Description : Un compte peut être créer par l’administrateur uniquement pour un nouveau responsable.
- Préconditions :
  - _ Formulaire rempli correctement (contenant les informations de la personne),
  - _ Administrateur connecté,
  - _ Droits : Administrateur,
- Entrées : Responsable
- Sorties : Responsable a son compte créé avec un mot de passe provisoire
- Postconditions : Le responsable a bien été ajouté à la base de données

2. Définition de la Fonction “MODIFIER SON PROFIL UTILISATEUR”

- Nom de la fonction : modifierProfil()
- Priorité : Secondaire - Facultative
- Description : Permet de modifier les informations du compte utilisateur connecté. Un administrateur a accès à tous les comptes utilisateur du site et peut pour le moment agir dessus librement. Un Responsable n’a accès qu’à son compte et ne peut modifier que son compte personnel.

- Préconditions :
  - _ Responsable existant,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Utilisateur (Responsable ou Admin)
- Sorties : Aucune
- Postconditions : Le responsable a bien été modifié dans la base de données

3. Définition de la Fonction "SUPPRIMER UN RESPONSABLE"

- Nom de la fonction : supprimerResponsable()
- Priorité : Secondaire - Facultative
- Description : Permet de supprimer les informations d'un compte utilisateur. Un administrateur a accès à tous les comptes utilisateur du site et peut pour le moment agir dessus librement. Un Responsable n'a accès qu'à son compte et ne peut supprimer que son compte personnel. Un administrateur ne peut pas supprimer son compte pour le moment.
- Préconditions : Membre déjà connecté,
  - _ Responsable existant,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Administrateur,
- Entrées : Responsable
- Sorties : Pop-up s'ouvrant et demandant la confirmation de la suppression d'un responsable
- Postconditions : Le login et le mot de passe du responsable ont bien été supprimés de la base de données dans le cas où l'on souhaite conserver une trace du responsable ou supprimer définitivement.

2.3 Gestion des œuvres

1. Définition de la Fonction "CONSULTER UNE OEUVRE"

- Nom de la fonction : ConsulterOeuvre()
- Priorité : Secondaire - Facultative
- Description : Permet d'obtenir l'ensemble des informations d'une œuvre. Pour le moment un utilisateur non connecté n'a pas accès à cette fonctionnalité.
- Préconditions :
  - _ Œuvre sélectionnée existe,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur,
- Entrées : Œuvre
- Sorties : Aucune
- Postconditions : Les informations de l'œuvre sont consultables

2. Définition de la Fonction "AJOUTER UNE OEUVRE"

- Nom de la fonction : ajouterOeuvre()
- Priorité : Secondaire - Facultative
- Description : Une œuvre peut être créée, ajoutée à un site et/ou une zone du site par le responsable du site ou l'administrateur.
- Préconditions : Formulaire rempli correctement (contenant les informations de l'œuvre et image AR Core générée),
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,

- Entrées : Œuvre
- Sorties : Aucune
- Postconditions : L'œuvre a bien été ajoutée dans la base de données

3. Définition de la Fonction "MODIFIER UNE OEUVRE"

- Nom de la fonction : modifierOeuvre()
- Priorité : Secondaire - Facultative
- Description : Permet de modifier l'œuvre notamment son appartenance à un site ou une zone et son état de publication (mode « Release » ou « Debug »)
- Préconditions : Membre déjà connecté,
  - _ Œuvre existante et appartient au site du responsable connecté,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Œuvre
- Sorties : Aucune
- Postconditions : L'œuvre a bien été modifiée dans la base de données

4. Définition de la Fonction "SUPPRIMER UNE OEUVRE"

- Nom de la fonction : supprimerOeuvre()
- Priorité : Secondaire - Facultative
- Description : Permet de supprimer une œuvre de la base de données (ou de l'archiver) si l'œuvre appartient au site du responsable connecté ou si c'est l'administrateur qui est connecté.
- Préconditions : Œuvre existante et appartient au site du responsable connecté
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Œuvre
- Sorties : Pop-up s'ouvrant et demandant la confirmation de la suppression d'une œuvre
- Postconditions : L'œuvre a été supprimée de la base de données (ou archivée).

5. Définition de la Fonction "OBTENIR LA LISTE OEUVRES"

- Nom de la fonction : consulterListeOeuvres ()
- Priorité : Secondaire - Facultative
- Description : Permet d'obtenir la liste complète des œuvres stockées dans la base de données sites et zones confondus. Pour le moment un utilisateur non connecté n'a pas le visuel des œuvres observables sur le site.
- Préconditions :
  - _ Œuvres existantes,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Aucune
- Sorties : La liste des œuvres
- Postconditions : La liste des œuvres est consultable.

2.4 Gestion des auteurs

1. Définition de la Fonction “CONSULTER UN AUTEUR”

- Nom de la fonction : ConsulterAuteur()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des informations d’un auteur. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité.
- Préconditions : Utilisateur sélectionnée existe,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur,
- Entrées : Auteur
- Sorties : Aucune
- Postconditions : Les informations de l’auteur sont consultables

2. Définition de la Fonction “AJOUTER UN AUTEUR”

- Nom de la fonction : ajouterAuteur()
- Priorité : Secondaire - Facultative
- Description : Un auteur peut être créé, ajoutée à un site et/ou une zone du site par le responsable du site ou l’administrateur.
- Préconditions : Formulaire rempli correctement (contenant les informations de l’auteur),
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Auteur
- Sorties : Aucune
- Postconditions : L’auteur a bien été ajouté dans la base de données

3. Définition de la Fonction “MODIFIER UN AUTEUR”

- Nom de la fonction : modifierAuteur()
- Priorité : Secondaire - Facultative
- Description : Permet de modifier l’auteur notamment son appartenance à un site ou zone
- Préconditions : Auteur existant et appartient au site du responsable connecté,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Auteur
- Sorties : Aucune
- Postconditions : L’auteur a bien été modifié dans la base de données

4. Définition de la Fonction “SUPPRIMER UN AUTEUR”

- Nom de la fonction : supprimerAuteur()
- Priorité : Secondaire - Facultative
- Description : Permet de supprimer un auteur de la base de données (ou de l’archiver) si l’auteur appartient au site du responsable connecté ou si c’est l’administrateur qui est connecté.
- Préconditions : Auteur existant et appartient au site du responsable connecté,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Auteur
- Sorties : Pop-up s’ouvrant et demandant la confirmation de la suppression d’un auteur
- Postconditions : L’auteur a été supprimé de la base de données (ou archivé).

5. Définition de la Fonction “OBTENIR LA LISTE AUTEURS”

- Nom de la fonction : consulterListeAuteurs()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir la liste complète des auteurs stockés dans la base de données sites et zones confondus. Pour le moment un utilisateur non connecté n’a pas le visuel des auteurs observables sur le site.
- Préconditions : Auteurs existants,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Aucune
- Sorties : La liste des auteurs
- Postconditions : La liste des auteurs est consultable.

2.5 Gestion des sites

1. Définition de la Fonction “OBTENIR LA LISTE DES SITES”

- Nom de la fonction : consulterListeSites ()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir la liste de tous les sites patrimoniaux qu’abrite la plateforme web. Pour le moment uniquement l’administrateur peut accéder à cette fonctionnalité.
- Préconditions : Sites existants,
  - _ Responsable ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur,
- Entrées : Auteur
- Sorties : La liste des site.
- Postconditions : La liste des sites est consultable.

2. Définition de la Fonction “CONSULTER UN SITE”

- Nom de la fonction : consulterSite()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des informations d’un site. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité. Seul le responsable du site en question ou un administrateur a accès à ce contenu.
- Préconditions : Site existant,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Site
- Sorties : Aucune
- Postconditions : Les informations du site sont consultables.

3. Définition de la Fonction “OBTENIR LA LISTE DES OEUVRES D’UN SITE”

- Nom de la fonction : consulterOeuvresSite()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des œuvres situées sur un site. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité. Seul le responsable du site en question ou un administrateur a accès à ce contenu.
- Préconditions : Site existant,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,

- Entrées : Site
- Sorties : La liste des œuvres du site en entrée.
- Postconditions : La liste des œuvres du site est consultable.

4. Définition de la Fonction “OBTENIR LA LISTE DES AUTEURS D’UN SITE”

- Nom de la fonction : consulterAuteursSite()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des auteurs ayant des œuvres appartenant à un site. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité. Seul le responsable du site en question ou un administrateur a accès à ce contenu.
- Préconditions : Site existant,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Site
- Sorties : La liste des auteurs du site en entrée.
- Postconditions : La liste des auteurs du site est consultable.

5. Définition de la Fonction “OBTENIR LA LISTE DES ZONES D’UN SITE”

- Nom de la fonction : consulterZonesSite()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des zones appartenant à un site. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité. Seul le responsable du site en question ou un administrateur a accès à ce contenu.
- Préconditions : Site existant,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Site
- Sorties : La liste des zones du site en entrée
- Postconditions : La liste des zones du site est consultable.

6. Définition de la Fonction “AJOUTER UN SITE”

- Nom de la fonction : ajouterSite()
- Priorité : Secondaire - Facultative
- Description : Un site peut être créé, on peut alors lui ajouter des zones, œuvres et auteurs.
- Préconditions : Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur,
- Entrées : Site
- Sorties : Aucune
- Postconditions : Le site a bien été ajouté dans la base de données

7. Définition de la Fonction “MODIFIER UN SITE”

- Nom de la fonction : modifierSite()
- Priorité : Secondaire - Facultative
- Description : Permet de modifier un site ainsi que ce qu’il contient. Fonctionnalité accessible uniquement pour le responsable du site en question sinon l’administrateur.
- Préconditions : Site Existant,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Site
- Sorties : Aucune
- Postconditions : Le site a bien été modifié dans la base de données.

8. Définition de la Fonction “SUPPRIMER UN SITE”

- Nom de la fonction : supprimerSite()
- Priorité : Secondaire - Facultative
- Description : Permet de supprimer un site de la base de données (ou de l’archiver) si le site est géré par le responsable connecté ou si c’est l’administrateur qui est connecté.
- Préconditions : Site Existant,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Site
- Sorties : Pop-up s’ouvrant et demandant la confirmation de la suppression d’un site.
- Postconditions : Le site a été supprimée de la base de données (ou archivé).

2.6 Gestion des zones

1. Définition de la Fonction “CONSULTER UNE ZONE”

- Nom de la fonction : ConsulterZone ()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des informations d’une zone. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité. Seul le responsable du site comprenant la zone en question ou un administrateur a accès à ce contenu.
- Préconditions : Zone Existante,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur,
- Entrées : Zone
- Sorties : Aucune
- Postconditions : Les informations de la zone sont consultables.

2. Définition de la Fonction “AJOUTER UNE ZONE”

- Nom de la fonction : ajouterZone()
- Priorité : Secondaire - Facultative
- Description : Une zone peut être créée, on peut alors lui ajouter des œuvres et auteurs.
- Préconditions : Formulaire rempli correctement, contenant les informations de la zone,
  - _ Zone Existante,
  - _ Responsable du site ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Zone
- Sorties : Aucune
- Postconditions : La zone a bien été ajoutée dans la base de données.

3. Définition de la Fonction “MODIFIER UNE ZONE”

- Nom de la fonction : modifierZone()
- Priorité : Secondaire - Facultative
- Description : Permet de modifier une zone notamment que ce qu’elle contient. Fonctionnalité accessible uniquement pour le responsable du site en question sinon l’administrateur.
- Préconditions : Zone Existante,
  - _ Responsable du site (contenant la zone) ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Zone
- Sorties : Aucune
- Postconditions : La zone a bien été modifiée dans la base de données.

4. Définition de la Fonction “SUPPRIMER UNE ZONE”

- Nom de la fonction : supprimerZone()
- Priorité : Secondaire - Facultative
- Description : Permet de supprimer une zone de la base de données (ou de l’archiver) si la zone est gérée par le responsable connecté ou si c’est l’administrateur qui est connecté.
- Préconditions :
  - _ Zone Existante,
  - _ Responsable du site (contenant la zone) ou Administrateur connecté,
  - _ Droits : Responsable ou Administrateur,
- Entrées : Zone
- Sorties : Pop-up s’ouvrant et demandant la confirmation de la suppression d’une zone.
- Postconditions : La zone a été supprimée de la base de données (ou archivée).

5. Définition de la Fonction “LISTE DES ŒUVRES D’UNE ZONE”

- Nom de la fonction : consulterOeuvresZone()
- Priorité : Secondaire - Facultative
- Description : Permet d’obtenir l’ensemble des œuvres appartenant à une zone d’un site. Pour le moment un utilisateur non connecté n’a pas accès à cette fonctionnalité. Seul le responsable du site en question ou un administrateur a accès à ce contenu.
- Préconditions :
  - _ Zone Existante,
  - _ Responsable du site (contenant la zone) ou Administrateur connecté,
  - _ Droits : Responsable et Administrateur ,
- Entrées : Zone
- Sorties : La liste des œuvres de la zone en entrée
- Postconditions : La liste des Œuvres de la zone est consultable.

3 Arbre hiérarchique des fonctionnalités

L’arbre hiérarchique décrivant les interactions entre les différentes composantes et fonctionnalités du système est disponible en annexe de ce document.

C Spécifications non Fonct.

1 Contraintes de développement

1.1 Matérielles et environnements nécessaires

En termes de contraintes matérielles aucune restriction n'a été formulée quant au système de développement à utiliser pour la réalisation de ce projet. Concernant l'environnement de développement, il a été fixé par le client que la solution proposée devait être une Web application. Il est donc nécessaire de pouvoir bénéficier d'un serveur Web pour héberger notamment la base de données mais également les autres modules qui composent le projet tel que l'API REST. Pour le moment, le besoin de bénéficier d'un espace de stockage à taille fixe n'a pas été défini. Ce dernier pourra donc être amené à évoluer au cours du projet. Comme évoqué précédemment, un serveur Polytech a déjà été mis à notre disposition. Ledit serveur héberge actuellement une base de données objet et une API REST pour la contacter. L'espace de stockage disponible actuellement sur ce serveur est de 100GB.

1.2 Langage de Programmation

Concernant la reconnaissance d'image et la création d'un modèle 3D des images, la technologie AR Core a été adoptée par l'équipe de conception. Les bibliothèques Android seront alors utilisées pour la création d'un système permettant la modélisation 3D des œuvres ainsi que l'application mobile attendue par le client.

Du côté serveur, l'API permettant l'accès à la base de données objet sera implémentée à l'aide du Framework PHP/Symphony ainsi que les utilitaires présents sur api-platform. Pour finir, la plateforme d'administration des sites patrimoniaux sera réalisée à l'aide du Framework JavaScript Angular (dans sa version 8), permettant l'utilisation d'outils tel que Canvas pour la gestion des ancres et balises sur une image.

1.3 Prototype

Dans un premier temps, des prototypes pourront être proposés afin de déterminer les solutions les plus adaptés en réponse des besoins émis par le client.

Un prototype de base de données et d'API sera dans un premier temps réalisé afin de simuler les demandes et échanges qui auront lieu entre les différentes applications qui compose le système global de l'application.

1.4 Délai de réalisation

Le module Web application Tourisme fait l'objet d'un sujet de projet recherche et développement. Le temps de réflexion et de mise en place de solutions est donc dépendant des dates fixées par Polytech. Cette contrainte implique une réalisation, vérification et une validation des tâches et fonctionnalités dans ce laps de temps en accord avec le client et l'encadrant. L'une des contraintes en lien avec le délai de réalisation serait également le souhait de voir naitre un prototype plus ou moins avancé de l'application et de le présenter au client d'ici la fin du projet.

2 Contraintes de fonctionnement et d'exploitation

Afin de s'assurer le bon fonctionnement du projet, il faudra prendre en compte l'ensemble des éléments et conditions de disfonctionnement évoqués dans les paragraphes suivants.

2.1 Performances

Le client n'a pour le moment pas émis de souhaits concernant les performances de la solution à produire. On pourra néanmoins supposer que le nombre d'utilisateurs sera tout de même relativement important étant donné que cette application recensera un grand nombre de sites patrimoniaux et possiblement beaucoup de visiteurs (utilisateurs de l'application). Afin de produire un prototype nous avons considéré une base de données comportant une vingtaine de sites et d'œuvres par site afin d'effectuer des premiers tests de performance.

Sans chiffre à l'appui, on peut considérer que l'application devra avoir un temps de réponse convenable, la visite effectuée par l'intermédiaire de l'application se réalisant en temps réel. De même, l'application étant à destination d'un grand nombre d'utilisateurs, sa fréquence d'utilisation sera très importante et impose donc un temps d'indisponibilité acceptable.

2.2 Capacités

Comme nous avons pu l'évoquer précédemment la base de données devra gérer un nombre de 20 sites et 20 œuvres par site pour le moment. Bien entendu cette notion est amenée à évoluer vers une capacité de stockage plus importante. C'est néanmoins une base sur laquelle nous avons souhaité tester le fonctionnement de notre application. A titre indicatif, le serveur universitaire sur lequel se situe la base de données a présentement une capacité de stockage de 100 GB. Cette marge plus que convenable concernant le stockage des données nous permettrait de réaliser de plus gros jeux de tests.

L'objectif final sera de faire correspondre un grand nombre de terminaux avec l'application globale afin de pouvoir tester les flux de communications.

2.3 Modes de fonctionnement

Concernant la plateforme web, le site sera en fonctionnement continu tant que le serveur est actif. Il pourra néanmoins être mis sous tension lors de maintenance ou l'apport d'évolutions aux composantes situées sur le serveur.

Le contact entre la base de données et l'application Android se fera essentiellement au lancement de l'application par l'utilisateur sinon au cours de l'utilisation de cette dernière. Le lancement de l'application Android et les fonctionnalités disponibles seront dépendants du mode sélectionné à savoir « Debug » (uniquement accessible par les responsables et l'administrateur) ou « Release ».

2.4 Contrôlabilité

Il est prévu d'intégrer dans un second temps des fichiers de logs permettant de suivre notamment le contrôle des transferts de fichiers entre le serveur et l'application Android concernant les images AR correspondant aux œuvres d'un site patrimonial. A cet effet, une table « Log » a été imaginée et sera adjointe au prototype de base de données afin de matérialiser cet aspect. Pour rappel, les images accessibles par l'intermédiaire de l'application mobile pour un site données dépendront du mode de lancement de l'application à savoir « Debug » ou « Release ». La version « Release » ne faisant apparaitre que les œuvre avec le statut publié.

2.5 Sécurité

Concernant l'accès à l'application et à toutes ses composantes, le serveur mis à notre disposition n'est pour le moment accessible que depuis le réseau universitaire. De ce fait, il ne peut y avoir de failles de sécurité liées à l'accessibilité du projet. De plus chaque utilisateur voulant avoir accès à la plateforme web devra se connecter à l'aide d'un identifiant et d'un mot de passe. Le processus de création de compte ne s'effectue que par l'intermédiaire de l'administrateur système qui est le seul à pouvoir ajouter un responsable de site.

(*) Les conditions de sécurité sont donc dépendantes du type d'utilisateur connecté (pour davantage d'informations sur les types d'utilisateur veuillez-vous rendre à la section 3.2.).

2.6 Intégrité

Dans le cas du chargement d'un fichier de base AR (contenant les images d'un site Debug ou Release) et de l'absence du dit fichier, l'application sera chargée de recréer le fichier en local puis d'effectuer l'envoi de ce dernier à l'utilisateur concerné conformément sa demande.

3 Maintenance et évolution du système

L'objectif à long terme de ce projet serait de pouvoir à terme proposer des évolutions du système tout en continuant à le maintenir. Pour rejoindre cet objectif à long terme fixé au début du projet, il faudra proposer une implémentation des fonctionnalités modulaire. Cette méthodologie permet de procéder aisément à l'ajout de nouveaux modules à la chaine de fonctionnement du système et contribue à faciliter l'évolution de l'application.

Devant faire face à des contraintes impactant le délai de réalisation comme évoqué précédemment, il a été décidé de fournir une documentation technique et utilisateur sur l'existant à la

fin du développement de l'application. Ce document ayant vocation à faciliter la reprise et la maintenance du projet.

Ce document aura pour premier objectif de faciliter la mise en place de l'environnement de développement ayant servi jusqu'alors. Il exposera donc les différentes procédures d'installations aussi bien sur les postes personnels que sur le serveur d'hébergement, cet aspect permettant la migration inter-serveurs. Une migration de serveur pouvant avoir lieu dans l'optique d'une recherche d'espace de stockage plus important par exemple. Pour rappel, la taille de l'espace de stockage n'a pas été fixé.

Dans le cadre de la mise en place d'une interface permettant la mise en place des balises et ancres sur une image, il est recherché d'avoir la possibilité de recourir à une interface 3D, amélioration envisageable pour une évolution de l'application. Cette évolution pourra faire l'objet d'un projet ayant pour but d'améliorer l'application principale si le délai de réalisation imposé pour ce projet ne le permet pas.

D Cahier du développeur

L'objectif est de décrire l'ensemble des étapes nécessaires à la mise en place de l'environnement de développement ayant servi durant tout le long de ce projet. L'outil Git a été utilisé pour le versionning de ce projet. Le dépôt contient donc déjà les informations permettant la mise en place de l'environnement de développement sous la forme d'un fichier « readme ». La suite de cette partie constitue donc une aide supplémentaire à la maintenance et la reprise du projet.

1 Installation de la partie back-end

Prérequis :

- PHP 7.4.2
- Composer, dans mon cas précis la version «version 1.10.1 2020-03-13 20 :34 :27 » a suffit pour faire fonctionner le projet.

Concernant la partie back-end, l'installation commence par copier le dépôt correspondant à la partie back de l'application à savoir « web-backend-core ». Le moment est venu d'avoir recours à notre utilitaire d'installation « composer ». Rendez-vous dans le dossier correspondant au dépôt une fois téléchargé et démarrez une Console de commande.

- Exécutez la commande « composer install » pour télécharger toutes les dépendances du projet.

Dans le cas ou votre base de données est stockée localement sur un serveur de type XAMPP ou WAMP il faudra modifier l'adresse du serveur base de données dans le fichier « .env » à la racine du serveur de l'API. Il faudra ensuite créer la base de données sur ce serveur à l'aide du schéma pré-enregistré dans le code serveur de l'utilitaire « doctrine ». Exécutez alors les commandes suivantes après modification du « .env » et démarrage de votre serveur sur lequel sera stockée la base de données de l'application.

- « php bin/console doctrine :database :create », permet la création de la base de données à partir du schéma doctrine.
- « php bin/console doctrine :schema :update –force », permet de forcer la mise à jour de la base de données et ainsi ne pas passer par des interactions utilisateur avant d'autoriser la mise à jour.

Pour finir vous prouvez démarrer le serveur de l'API REST :

- « php -S 127.0.0.1 :8000 -t public » (en adaptant l'adresse et le port souhaité sur lesquels déployer l'API REST si c'est nécessaire).

Remarque : Si vous souhaitez quitter le serveur, un simple Ctrl+C suffit dans l'invite de commande où avait été précédemment démarrée l'API.

2 Installation de la partie front-end

Pour la partie Front, il faudra commencer par s'assurer que vous êtes en capacité d'effectuer les commandes de type « npm ». Cette condition remplie, vous pouvez télécharger les fichiers correspondants au dépôt front à savoir « visit-backend-ui ».

De même que pour la partie Back, rendez-vous dans le dossier correspondant au dépôt Git téléchargé et démarrez une invite de Commande.

- Téléchargez les dépendances de votre projet à l'aide de la commande « npm install » (d'où l'aspect critique de la première étape car sans cette condition, impossible de poursuivre l'installation).
- Pour être certain d'avoir télécharger l'ensemble des dépendances, exécutez la commande « npm install -g @angular/cli ». Dans le cas où cette dépendance n'avait pas été correctement installée via la commande précédente, elle le sera maintenant.

Enfin vous pouvez démarrer le serveur de l'interface Web à l'aide de la commande suivante : « ng serve »

Afin de faciliter la reprise ou maintenance du projet, un guide décrivant l'architecture des parties Back-end et Front-end qui constituent l'application VISIT a également été joint à ce document.

3 Architecture Back-end

Pour rappel l'API REST constituant la partie Back-end de l'application a été réalisée à l'aide des technologies PHP Symfony et API Platform. Cette partie du projet VISIT est architecturée de la manière suivante.

Les dossiers de configuration du projet propres aux Frameworks, utilitaires et langages ayant servi pour créer l'API se situent dans les dossiers « /bin », « /config », « /vendor » et « /public/bundle ». Le dossier « /var » comporte les informations propres à l'utilisateur tel que les logs serveur. Le dossier « /arcoreimg » contient des utilitaires permettant de gérer les images AR que nous manipulons dans ce projet selon l'OS utilisé. Les sources de l'ensemble du projet sont présentes dans le dossier « /src ». De même pour les tests qui sont dans le dossier associé « /tests ».

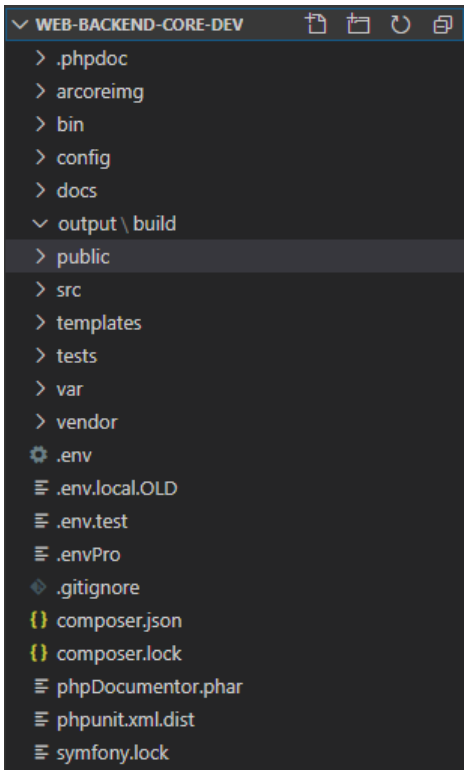


Figure 1 – *Architecture Back-end*

Concernant la fonctionnalité de création de package de données à destination de l'application mobile comme nous avons pu l'évoquer précédemment. Cette commande va chercher les médias situés sur le serveur dans le dossier « /public/media », les JsonStates correspondants présents dans le dossier « /public/state » et les bases de données AR créées à partir des fichiers précédents, stockées dans « /public/arcore» pour créer une archive zip « debug » et « release » de chaque site dans le dossier « /public/package ».

4 Architecture Front-end

Pour l'architecture utilisée pour l'implémentation de la partie front-end, nous avons suivi celle proposée par le Framework angular, cette dernière se présentant de la manière suivante.

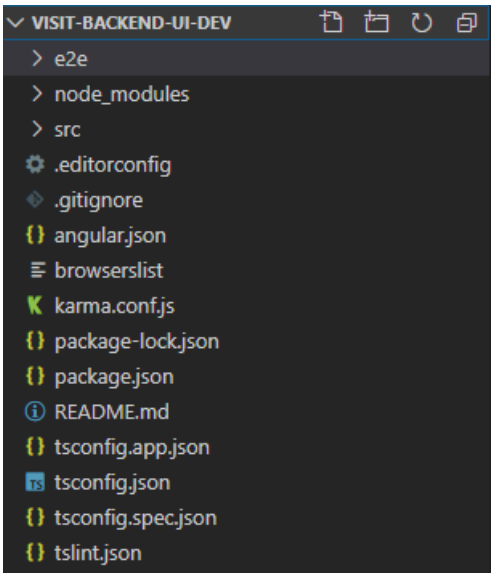


Figure 2 – *Architecture Front-end*

Les dossiers de configuration du projet propres aux Frameworks, utilitaires et langages ayant servi pour créer l'interface d'administration des sites patrimoniaux se situent dans les dossiers « /node_modules », « /e2e ». Les sources de l'ensemble du projet sont présentes dans le dossier « /src ».

5 Architecture du code de création de la commande de package

Cette sous partie vise à faciliter la navigation du développeur dans les différents fichiers s'il souhaite trouver le code constituant le cœur de la création de package de données à destination de l'application mobile. Ledit code est présent dans la partie back-end du serveur, dans le dossier « /src » conformément aux normes choisies.

Il se décompose en deux sous fichiers, lesquelles étant « /src/Command/CreateBundleCommand.php » et « /src/Service/PackageService.php ». Le premier étant la déclaration de la commande php suivant le Framework PHP Symfony et le second constituant le cœur de la création des différents packages.

On se sert également des fichiers se rapportant aux différents « repository » des tables de la bases de données pour créer nos packages ainsi que le fichier « /src/Service/SiteStateService.php » qui établit le lien entre les JsonStates présents sur le serveur dans le dossier « /public/state » et les siteStates enregistrés dans la base.

6 Lancement de la commande de création de package

Pour lancer la commande de création d'un package correspondant à un site donné, il suffit de se rendre dans une invite de commande à la racine de la partie back-end du serveur et d'effectuer la commande : « php bin/console app :create-package idSite»

"idSite" est un argument de la commande de création de package suivant sa valeur la commande ne produira pas le même résultat.

- « all » : la saisie de « all » comme argument de la commande entrainera la génération des packages release et debug pour l'ensemble des sites de la base de données possédant un jsonState (c'est à dire tous les sites patrimoniaux dans la base sauf erreur d'intégrité de la base de données).
- « 5 » : A valeur d'exemple la saisie de l'idSite « 5 » va générer les deux packages release et debug pour le site id 5.

E Plan de développement

La méthode de développement choisie et notifiée au début de ce document est la méthode agile ou « pseudo-agile », le retour client s'effectuant auprès de l'encadrant du projet. Cette méthode implique de mettre en place des retours clients réguliers afin de lui communiquer les avancées du projet.

Ce retour prendra la forme de livrables, qui seront déposés à l'emplacement voulu sur la zone de dépôt prévue à cet effet (Dépôt Git). Chaque livrable comprendra l'ensemble des fonctionnalités développées. Ces dernières seront intégrées à la plateforme après validation.

Des documents présentant chaque phase de réalisation seront joints à ce livrable si des précisions sur les démarches effectuées s'avéraient nécessaire. De plus, il sera fourni dans la mesure du possible, une documentation technique et utilisateur à chacune de ces phases (sinon assurément à la fin du projet).

Le choix de la méthode de gestion de projet adopté s'étant porté sur un Scrum « flexible », la suite du document décrit les tâches à effectuer en amont de la phase de développement et revient plus précisément sur les potentiels Sprints à effectuer lors du projet.

La suite du document constitue donc une planification prévisionnelle du projet qui sera sujet à modification au cours de la phase de développement de la solution en accord avec le client. (Encadrant)

1 Spécifications - Recherche de solutions

La base de données et l'API REST utilisant api-platform ainsi que la configuration serveur pour l'hébergement de l'application seront mises en place lors de la phase de recherche de solutions permettant ainsi de démarrer directement la phase de développement par l'implémentation des fonctionnalités qui m'ont été attribuées. Cette phase pré-développement sera également l'occasion d'explorer diverses technologies permettant de répondre aux besoins clients. Des recherches autour des technologies AR Core et techniques SVG / Canvas seront réalisées durant ce laps de temps.

2 Sprint 1

Ce premier Sprint vise à réaliser les fonctionnalités étant identifiées comme ayant la plus grande priorité à savoir celles qui m'ont été officiellement attribuées. Pour rappel, ces deux tâches sont :

- La génération et l'envoi de la base de données AR à l'application Android,
- La gestion des annotations sur les images AR,

Etant donné une interdépendance forte entre le développement du module de l'application Android et la génération - envoi de la base de données AR, il a été décidé d'accorder une priorité particulière à cette tâche. Les fonctionnalités concernant cette partie de la phase de développement sont donc à réaliser en premier. Des documents contenant des informations sur la mise en place des solutions correspondant à ces fonctionnalités seront fournis en même temps que le livrable correspondant à cette partie. Une validation sera donc attendu à la fin de ce Sprint pour convenir de la suite du processus de développement.

3 Sprint 2

La seconde phase consistera à mettre en place des processus de tests afin de respecter le plan d'assurance qualité joint à ce document. Cette phase consistera à la mise en place de correctif sinon à l'intégration de la solution à l'application globale après validation du client. C'est également dans cette phase que seront possiblement attribués de nouvelles tâches parmi celles évoquées comme secondaires ou facultatives dans la section spécifications. La suite du développement dépendra donc des nouvelles fonctionnalités attribuées. La proposition d'un premier livrable à l'issu de ce sprint consistera pour le développeur un contrôle lui permettant de planifier les Sprints suivants et pour le client, un aperçu des solutions proposées sur les fonctionnalités additionnelles.

4 Sprint 3

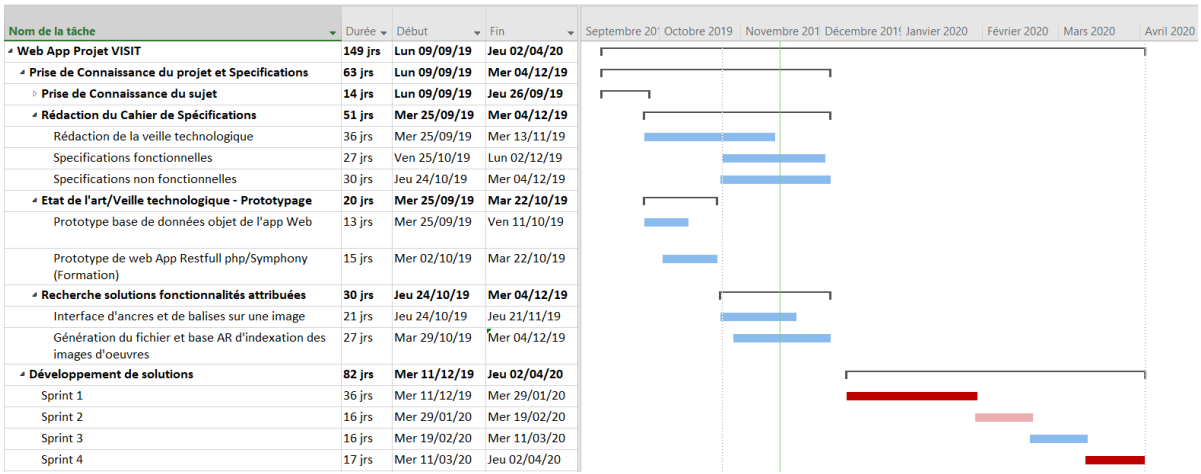
L'objectif de cette partie sera de continuer l'implémentation de solutions proposées précédemment pour aboutir à la mise en place de l'ensemble des tâches attribuées à la fin de cette période. Le livrable sera accompagné d'une documentation expliquant la mise en place des solutions proposées dans la phase précédente. Dans le cas où l'ensemble des tâches attribuées ne peut être réalisé compte tenu des contraintes de délai de réalisation, il sera demandé au client de donner une nouvelle priorité aux tâches restantes afin de satisfaire à moindre mal ses besoins.

5 Sprint 4

Ce sprint sera l'occasion de réaliser des tests (pour la plupart fonctionnelles) de l'application afin de respecter le plan de test mis en place initialement et décrit dans le plan d'assurance qualité présent en annexe. Une revue de code sera également effectuée permettant également de revoir la documentation du code. On réservera également ce temps post-développement à la création sinon modification de l'ensemble des documents pouvant contribuer la maintenabilité du projet et à l'apport d'évolution.

6 Diagramme de Gantt prévisionnel

L'anticipation du processus de développement a permis de décrire les étapes menant à la résolution des problématiques et besoins exprimés par le client.



Nom de la tâche	Durée	Début	Fin
Web App Projet VISIT	**149 jrs**	**Lun 09/09/19**	**Jeu 02/04/20**
Prise de Connaissance du projet et Specifications	**63 jrs**	**Lun 09/09/19**	**Mer 04/12/19**
Prise de Connaissance du sujet	**14 jrs**	**Lun 09/09/19**	**Jeu 26/09/19**
Rédaction du Cahier de Spécifications	**51 jrs**	**Mer 25/09/19**	**Mer 04/12/19**
Rédaction de la veille technologique	36 jrs	Mer 25/09/19	Mer 13/11/19
Specifications fonctionnelles	27 jrs	Ven 25/10/19	Lun 02/12/19
Specifications non fonctionnelles	30 jrs	Jeu 24/10/19	Mer 04/12/19
Etat de l'art/Veille technologique - Prototypage	**20 jrs**	**Mer 25/09/19**	**Mar 22/10/19**
Prototype base de données objet de l'app Web	13 jrs	Mer 25/09/19	Ven 11/10/19
Prototype de web App Restfull php/Symphony (Formation)	15 jrs	Mer 02/10/19	Mar 22/10/19
Recherche solutions fonctionnalités attribuées	**30 jrs**	**Jeu 24/10/19**	**Mer 04/12/19**
Interface d'ancres et de balises sur une image	21 jrs	Jeu 24/10/19	Jeu 21/11/19
Génération du fichier et base AR d'indexation des images d'oeuvres	27 jrs	Mar 29/10/19	Mer 04/12/19
Développement de solutions	**82 jrs**	**Mer 11/12/19**	**Jeu 02/04/20**
Sprint 1	36 jrs	Mer 11/12/19	Mer 29/01/20
Sprint 2	16 jrs	Mer 29/01/20	Mer 19/02/20
Sprint 3	16 jrs	Mer 19/02/20	Mer 11/03/20
Sprint 4	17 jrs	Mer 11/03/20	Jeu 02/04/20

Figure 1 – *Diagramme de Gantt prévisionnel*

7 Diagramme de Gantt final

Une fois la phase de développement passée, nous pouvons revenir sur le planning prévisionnel établit pendant la période d'analyse du projet afin d'exposer les modifications qui ont eu lieu.

7.1 Problèmes rencontrés

Ce premier point de cette partie expose les problèmes rencontrés sinon éléments ayant ralenti où bloquer le développement des solutions du projet. Ces derniers ont été décrits dans un tableau en fin de cette sous partie

Le premier et troisième point bloquants exposés dans ce tableau sont liés de manière globale à l'apprentissage de nouveaux langages et techniques utilisés. Ce qui peut en réalité être un problème commun avec un grand nombre de projets. En effet, ces derniers mettent en jeu des problématiques variés et l'utilisation d'un langage et/ou technique particulière peut s'avérer être une bonne solution de départ. Ce risque avait donc été identifié pendant la phase d'analyse du projet mais il s'est vu être difficilement mesurable en termes de temps à accorder à la phase de compréhension et formation (PHP Symfony / Angular).

Le projet m'amenant à travailler en collaboration avec d'autres membres de l'équipe, il a été difficile d'évoluer en parallèle sans avoir le même temps à accorder au développement de l'application. Cette interdépendance entre les différents blocs à développer ou à comprendre/reprendre pour implémenter ses propres solutions a en effet compliqué l'avancée du projet dans les temps.

Problème rencontré	Risque(s) identifié(s)	Alerte(s) levée(s)	Action(s) menée(s)
Non connaissance des langages et techniques utilisés	*Risque identifié : Apprentissage*	Echange avec membre de l'équipe connaissant les technologies utilisées et recherches personnelles.	Etude des technologies qui a nécessité un certain temps.
Evolution de projet en parallèle, interdépendances	*Risque identifié : Délai de réalisation*	Mise à niveau avec ce même membre de l'équipe pour tenter de rattraper le retard accumulé.	1 personne avançant tous les jours sur le projet et moi devant rattraper le retard lié principalement au fait que nous ne sommes que 2 jours par semaine sur le projet (heures prévues).
Architecture projet	*Risque identifié : Apprentissage*	Mise à niveau avec ce même membre de l'équipe qui a facilité ma compréhension.	Non connaissance des technologies et frameworks utilisés ce qui m'a ralenti dans la navigation de notre architecture de code.
Changement d'environnement de développement	*Risque identifié : Apprentissage, intégration, délai de réalisation*	Echange avec membre de l'équipe connaissant les technologies utilisées et recherches personnelles qui n'a pas abouti par la suite. , m'obligeant par la suite à revenir vers l'environnement de départ.	Passage d'un environnement de développement classique a une image docker (seulement quelques notions dans la conteneurisation avant le début du projet) pour finalement revenir vers l'environnement de base pour ma part.
Contraintes temporelles	*Risque identifié : Délai de réalisation*	Un feedback régulier ayant eu lieu sur l'état d'avancement du projet, une solution a pu être adopté en accord avec l'encadrant. Finir la tâche prioritaire	Le temps de développement fixé par l'école n'a permis de réaliser que la tâche 1 : Création et génération de package. Tâche prioritaire en accord avec l'encadrant.

Figure 2 – *Problèmes rencontrés*

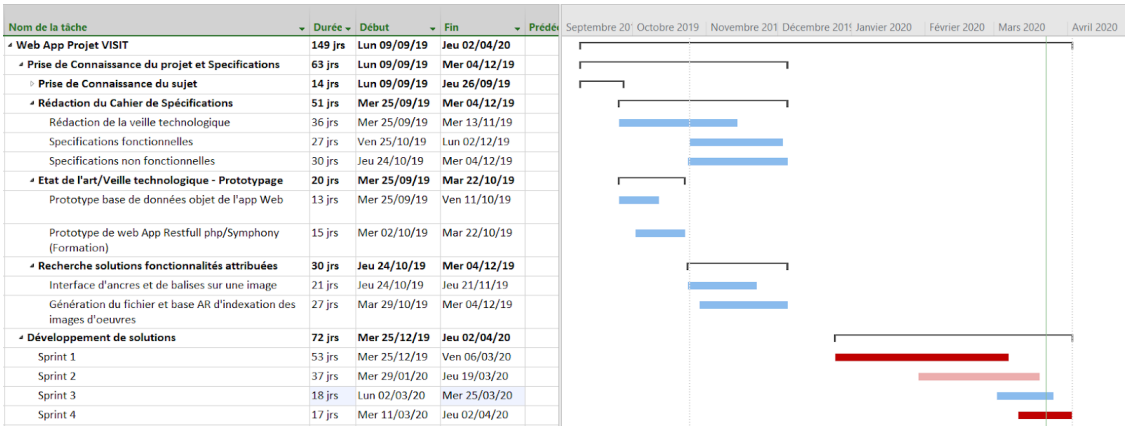
7.2 Conséquences sur le développement

Devant faire face à des contraintes temporelles imposées par la durée de l'année comme il l'a été évoqué dans la partie précédente, il était nécessaire de faire un point régulier avec l'encadrant et les membres de l'équipe afin d'être en mesure d'avancer sur le développement. Ces feedbacks réguliers ont permis de recentrer les priorités de développement aboutissant à la décision de se concentrer davantage sur la tâche constituant la création des packages de données pour l'application mobile.

Ce choix s'est donc fait au détriment de la tâche de création d'une interface de gestion d'annotations sur une œuvre. L'entité « annotation d'œuvre » a tout de même été ajoutée à la base de données et des services permettant de la manipuler via l'api ont été implémentés. Nous n'avons cependant pas pu aller jusqu'à la phase de création d'une interface de gestion des balises/annotations sur une œuvre.

Les tâches constituants les sprints 3 et 4 ont néanmoins pu être menées à bien à savoir la création de documentions et rapport utilisateurs et développeurs permettant de faciliter la prise en main de l'application et d'assurer le suivi et la maintenance du projet. Des tests ont également pu être menés permettant de valider les solutions mises en place. Pour plus de détails sur ces différents éléments veuillez-vous référer aux parties « guide utilisateur », cahier de développeur » et « rapport de tests ».

L'ensemble des éléments repris dans le tableau précédent a eu néanmoins pour conséquence de modifier le diagramme de Gantt de ce projet l'amenant à la version finale suivante :



Nom de la tâche	Durée	Début	Fin	Prédé
Web App Projet VISIT	**149 jrs**	**Lun 09/09/19**	**Jeu 02/04/20**	
Prise de Connaissance du projet et Specifications	**63 jrs**	**Lun 09/09/19**	**Mer 04/12/19**	
Prise de Connaissance du sujet	**14 jrs**	**Lun 09/09/19**	**Jeu 26/09/19**	
Rédaction du Cahier de Spécifications	**51 jrs**	**Mer 25/09/19**	**Mer 04/12/19**	
Rédaction de la veille technologique	36 jrs	Mer 25/09/19	Mer 13/11/19	
Specifications fonctionnelles	27 jrs	Ven 25/10/19	Lun 02/12/19	
Specifications non fonctionnelles	30 jrs	Jeu 24/10/19	Mer 04/12/19	
Etat de l'art/Veille technologique - Prototypage	**20 jrs**	**Mer 25/09/19**	**Mar 22/10/19**	
Prototype base de données objet de l'app Web	13 jrs	Mer 25/09/19	Ven 11/10/19	
Prototype de web App Restfull php/Symphony (Formation)	15 jrs	Mer 02/10/19	Mar 22/10/19	
Recherche solutions fonctionnalités attribuées	**30 jrs**	**Jeu 24/10/19**	**Mer 04/12/19**	
Interface d'ancres et de balises sur une image	21 jrs	Jeu 24/10/19	Jeu 21/11/19	
Génération du fichier et base AR d'indexation des images d'oeuvres	27 jrs	Mar 29/10/19	Mer 04/12/19	
Développement de solutions	**72 jrs**	**Mer 25/12/19**	**Jeu 02/04/20**	
Sprint 1	53 jrs	Mer 25/12/19	Ven 06/03/20	
Sprint 2	37 jrs	Mer 29/01/20	Jeu 19/03/20	
Sprint 3	18 jrs	Lun 02/03/20	Mer 25/03/20	
Sprint 4	17 jrs	Mer 11/03/20	Jeu 02/04/20	

Figure 3 – *Diagramme de Gantt final*

F Test / Assurance qualité

Les méthodes de test et le plan d'assurance qualité sont décrits dans cette partie qui complète le cahier des spécifications.

1 Méthodologie

Chaque personne en charge d'une tâche est également en charge de rédiger les vérifications qui y correspondent. Le chef de projet effectuera également une seconde vérification lors la mise en ligne du travail et effectue les retours nécessaires au(x) responsable(s) de cette fonctionnalité. Toute personne rencontrant un document ne suivant pas les recommandations du PAQ (Plan d'Assurance Qualité) est priée d'en faire part au responsable de la tâche et/ou au chef de projet. Tout ce qui sera produit durant le projet sera écrit en français.

2 Vérifications – Interface graphique

Chaque interface doit être commentée et décrire précisément à quelles fonctionnalités elle est liée, via quelle entrée (bouton, champs) et quelle sortie (messages, boite de dialogue). Son fonctionnement général doit être explicite et intuitif.

3 Vérifications – Fonctionnalités

Chaque fonctionnalité doit présenter les 4 points suivants :

- - But/Objectif
- - Entrées
- - Principe/Fonctionnement
- - Sorties

4 Vérifications – Code

- Général :
  - L'initialisation des variables se fait au début de la fonction et est suivie par un saut de ligne dans un soucis de clarté
  - Chaque bloc d'opérations liées entre-elles est séparer par un saut de ligne. Exemple : Une ouverture de fichier, une écriture et fermeture sont un même bloc ; un boucle de traitement est un même bloc.
  - Toute classe, fonction, méthode doit être testée ET passer ces tests sans quoi elle ne sera pas validée.
- Variables :
  - Les noms de variables sont explicites, succincts et commence par une minuscule.
  - Les variables ont un seul usage qui est défini par leur nom.
  - Toute propriété ou variable ayant un nombre limité et/ou précis de valeurs doit passer par un enum (soit dans le fichier d'enum soit en static dans la classe le concernant).
- Classes :
  - Les classes doivent commencer par une majuscule et respecter la convention de nommage fixée par l'équipe de projet.
  - Toute classe doit être documentée et commentée :
    - _ Auteur (responsable)
    - _ Description/Objectif
    - _ Propriétés
    - _ Méthodes (sauf constructeur)
  - Les fonctions qui affectent des données doivent vérifier la modification et retourner un booléen, ainsi qu'un message visuel clair en cas de problème.
- Fonctions :
  - Les fonctions doivent être nommées avec des verbes à l'infinitif, commencer par une minuscule et chaque nouveau mot doit commencer par une majuscule (exemple : ajouteOeuvre).
  - Toute fonction doit être documentée et commentée à l'exception des getters/setters ainsi que les fonctions auto-générées(sauf si elles ont été modifiées)
    - _ Auteur (responsable)
    - _ Description/Objectif
    - _ Entrées
    - _ Fonctionnement si complexe
    - _ Sortie
  - Les fonctions qui affectent des données doivent vérifier la modification et retourner un booléen, ainsi qu'un message visuel clair en cas de problème.

5 Documentation

Une documentation Doxygène a été générée concernant le code du serveur back-end (API, commande de génération des packages et autres utilitaires permettant de gérées les images RA). Cette documentation sera jointe à l'archive de rendu

5.1 documentation d'installation et de déploiement

Une documentation d'installation ainsi qu'une aide développeur pour la reprise du projet a été mise en place dans le cahier du développeur correspondant à la partie du même nom dans ce rapport (description choix techniques, installation et architecture de codage).

5.2 Documentation d'utilisation

Une guide utilisateur a été rédigé afin d'aider l'utilisateur à appréhender l'interface d'administration des sites patrimoniaux. Ce document d'utilisation est présent en annexe du présent rapport. (« Guide Utilisateur »)

6 Rapport de tests

Le rapport des tests effectués et des mesures mis en place afin de garantir la qualité du code produit pour procéder au contrôle de Vérification et Validation des solutions implémentées est présent dans le rapport du même nom sinon en annexe de ce présent rapport. (« Tests et Qualité de codage »)

G Guide Utilisateur

Ce guide explique le fonctionnement de l'application web dans sa globalité pour tous les types d'utilisateur souhaitant y accéder.

1 Authentification

En lançant l'interface web de l'application VISIT vous allez arriver sur la page d'accueil du site. Il faudra ensuite vous authentifier pour accéder au contenu correspondant à vos droits utilisateurs définis par le super administrateur du site

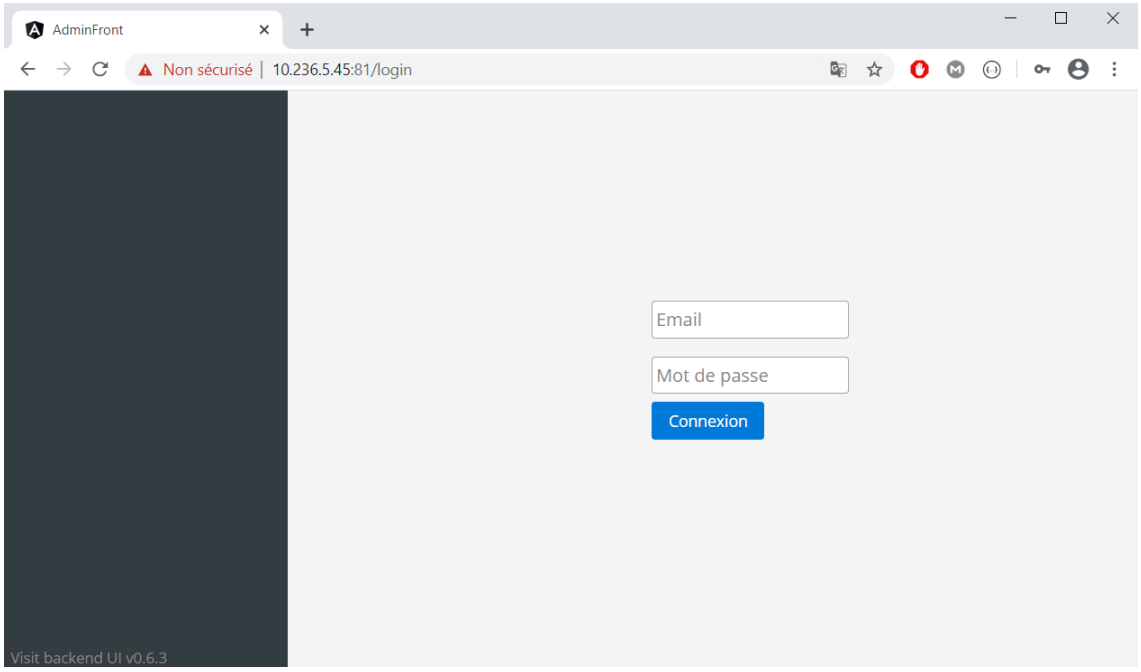


Figure 1 – *Page d'accueil du site*

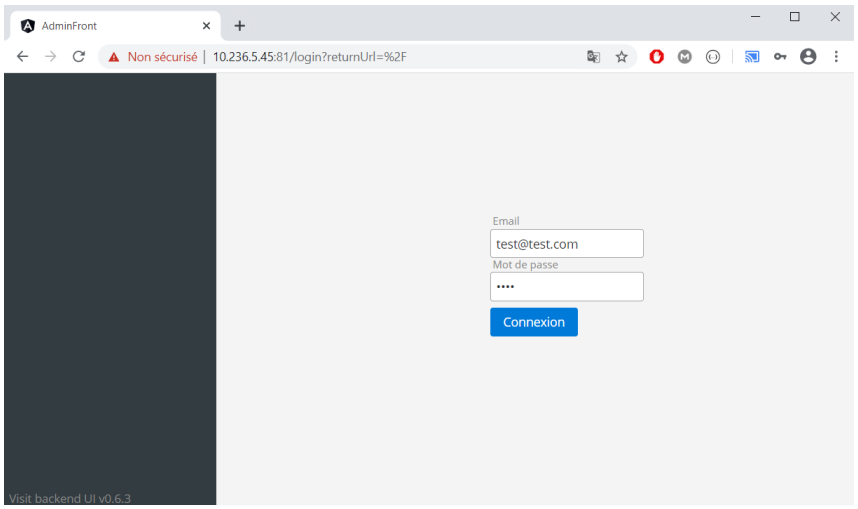


Figure 2 – *Authentification*

2 Responsable de site

2.1 Page d'accueil Responsable de site

Un responsable de site une fois authentifié sera automatiquement redirigé vers sa page d'accueil personnalisée. Il a alors accès à diverses options disponibles pour la plupart sous forme de menus déroulants.

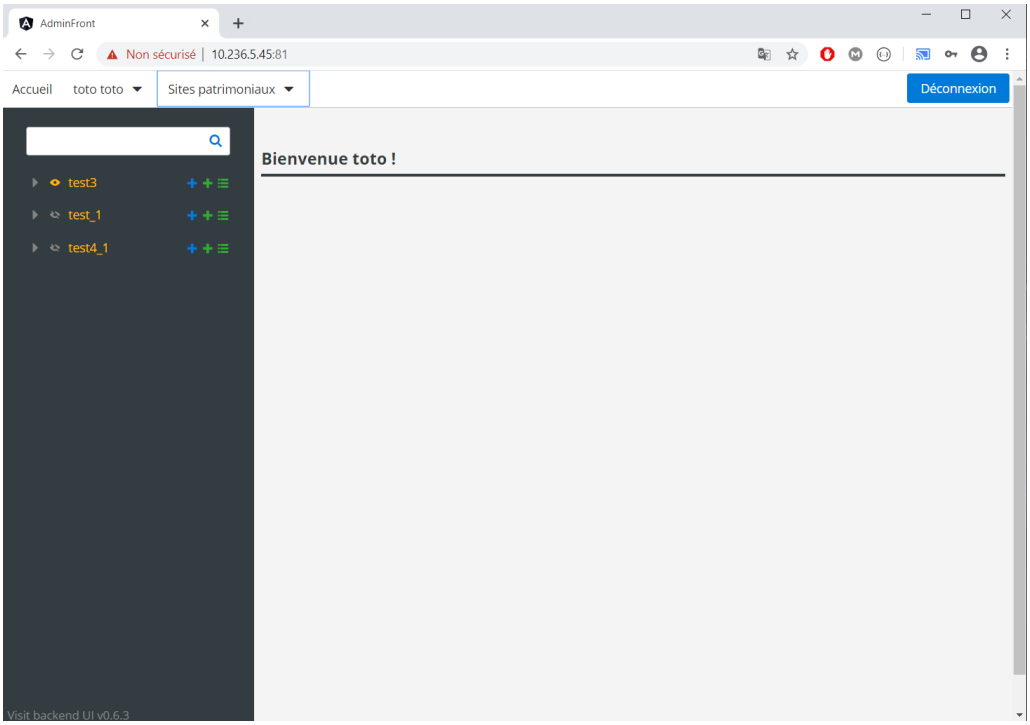


Figure 3 – *Page d'accueil une fois Authentifiée (Responsable de site)*

_ Le bouton « Accueil » permet de revenir à la page d'accueil de l'utilisateur connecté, c'est-à-dire la page décrite dans l'image précédente.
_ Le menu déroulant portant l'intitulé correspondant au nom et prénom de l'utilisateur connecté permet d'avoir accès à deux fonctionnalités primaires du site à savoir la consultation du profil utilisateur connecté et la demande de modification de mot de passe.

_ Le dernier menu de cette page « Sites patrimoniaux » permet au responsable connecté de s'informer sur les sites patrimoniaux recensés dans la base de données. Il ne pourra avoir accès qu'à des actions limitées tel que la consultation dans le cas où le site en cours de consultation n'est pas le sien.

2.2 Page de gestion de son profil

A la sélection de l'option « Profil » située dans le menu déroulant correspondant au nom et prénom de l'utilisateur connecté, il est possible d'accéder à ses informations personnelles tel que le nom, prénom, email et numéro de téléphone et de pouvoir les modifier. Le statut utilisateur lui ne peut être modifié que par le super administrateur du site. (voir la section gestion des utilisateurs)

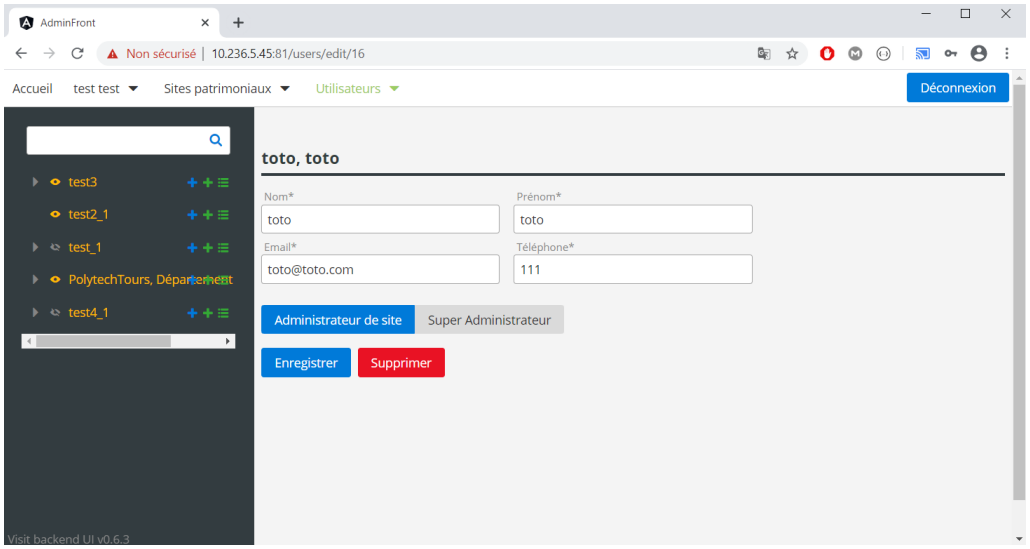


Figure 4 – *Page d'accès au profil utilisateur(Responsable de site)*

2.3 Page d'ajout, modification de son site patrimonial

L'utilisateur a également la possibilité de consulter le site patrimonial dont il est le responsable par l'intermédiaire du menu déroulant « site patrimoniaux » en sélectionnant la ligne correspondant à son site. Il a alors la possibilité de voir l'ensemble des informations caractérisant le site dont il a la charge. Il peut également interagir avec ses dernières s'il souhaite en modifier leur contenue. Il est chargé de déterminer la localisation de son site patrimonial. Il a à sa disposition pour cela une carte lui permettant de situer de manière graphique la localité dans laquelle se trouve son site (les données latitude et longitude se mettant à jour automatiquement suivant la saisie). Il doit également s'assurer d'entrer un seuil de détection sur site (permettant le chargement des données pour les utilisateurs souhaitant effectuer une visite) si le site patrimonial en question n'est pas suffisamment couvert du point de vu connexion à internet. Remarque : Le mauvais paramétrage de cette zone peut s'avérer problématique pour la mise en place de visites étant donné la nécessité de télécharger les données via un réseau pour permettre aux visiteurs d'effectuer leurs visites sans perte de données.

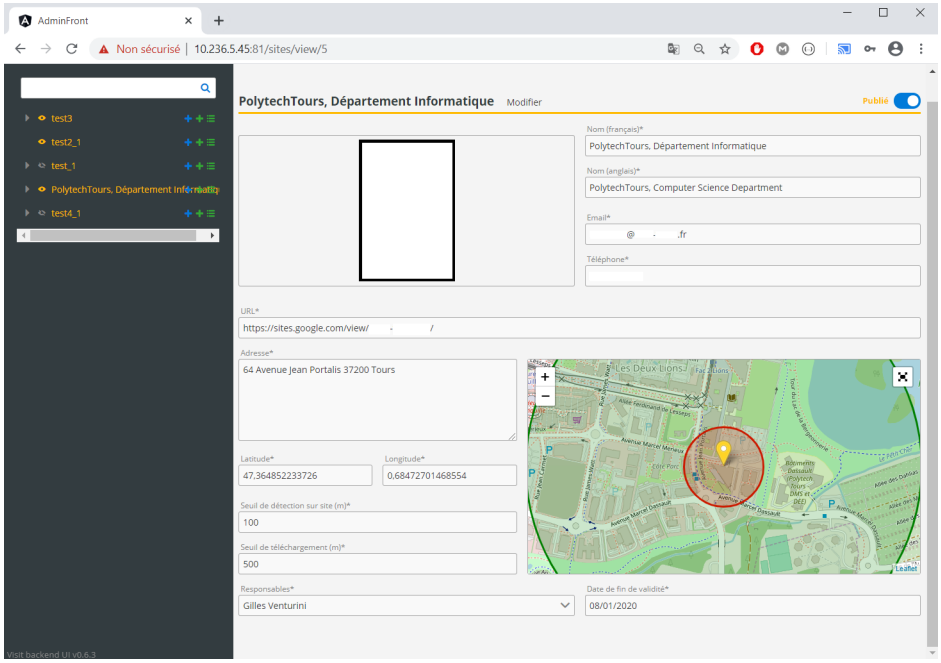


Figure 5 – *Page d'ajout, modifications de son site patrimonial (Responsable de site)*

2.4 Page de consultation des zones de son site patrimonial

L'utilisateur peut consulter les différentes zones que son site patrimonial abrite. Ces dernières sont accessibles par le menu « Sites patrimoniaux » et sous menu liste des zones en sélectionnant le site géré par l'utilisateur connecté. Il est possible d'entrer dans le menu de modification ou de suppression d'une zone en appuyant sur les boutons créés à cet effet tel qu'il est possible de le voir dans l'image ci-dessous. Remarque : L'accès à cette liste est également possible depuis le menu de consultation du site patrimonial en question en suivant la même procédure à ceci près qu'il faut sélectionner « liste des zones » lorsque l'on est dans le menu de consultation de son site.

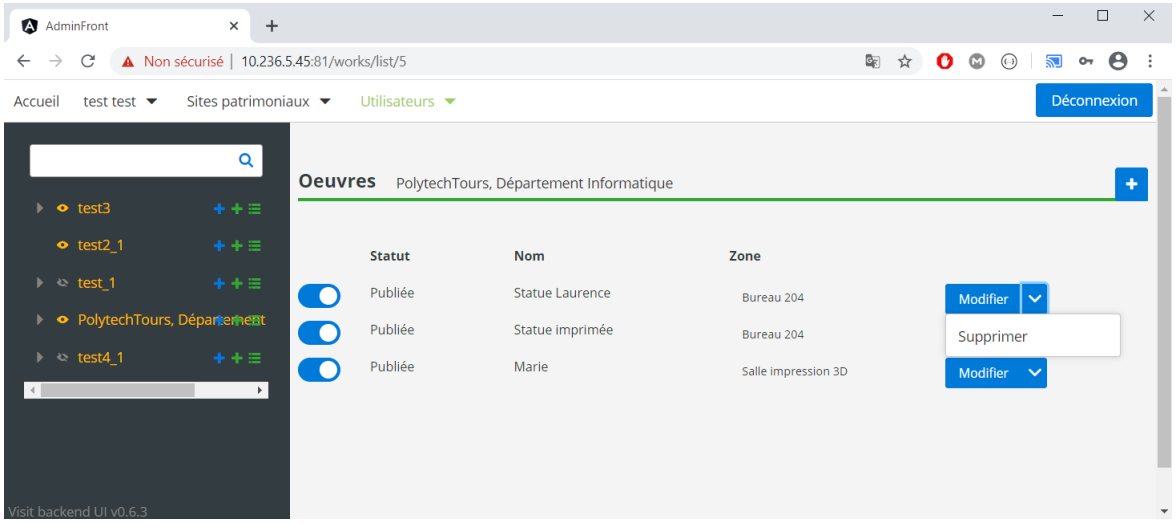


Figure 6 – *Page de consultation des zones d'un site (Responsable de site)*

2.5 Page de création d'une nouvelle zone de son site patrimonial

Si l'on sélectionne « Nouvelle zone » à la place de liste des zones dans le menu déroulant «Sites patrimoniaux » et la sous-section correspondant à son site, l'application nous propose d'ajouter une nouvelle zone à notre site patrimonial. L'interface se présente alors de la manière suivante et permet l'ajout de cette nouvelle zone après validation de l'utilisateur à l'aide du bouton « Enregistrer ».

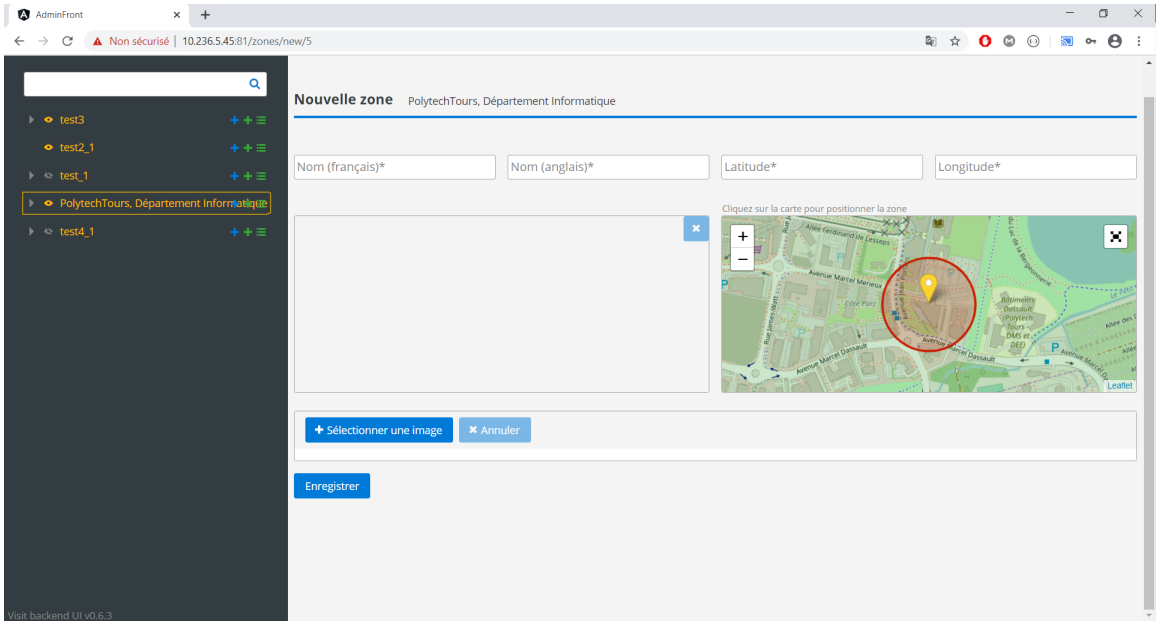


Figure 7 – *Page d'ajout d'une nouvelle zone (Responsable de site)*

2.6 Page des œuvres de son site patrimonial

L'utilisateur a la possibilité de consulter les différentes œuvres que son site patrimonial abrite. Ces dernières sont accessibles par le menu « Sites patrimoniaux » et sous menu liste de œuvres en sélectionnant le site géré par l'utilisateur connecté.

Une fois la liste affichée, l'utilisateur souhaitant modifier le statut de l'œuvre de « publié » à « non publié » ou inversement n'aura qu'à activer le bouton correspondant. Il est également possible d'entrer dans le menu de modification ou de suppression d'une œuvre en appuyant sur les boutons créés à cet effet tel qu'il est possible de le voir dans l'image ci-dessous.

Remarque : L'accès à cette liste est également possible depuis le menu de consultation des zones du site patrimonial en question en suivant la même procédure à ceci près qu'il faut sélectionner « liste des zones » puis « liste des œuvres » de cette zone.

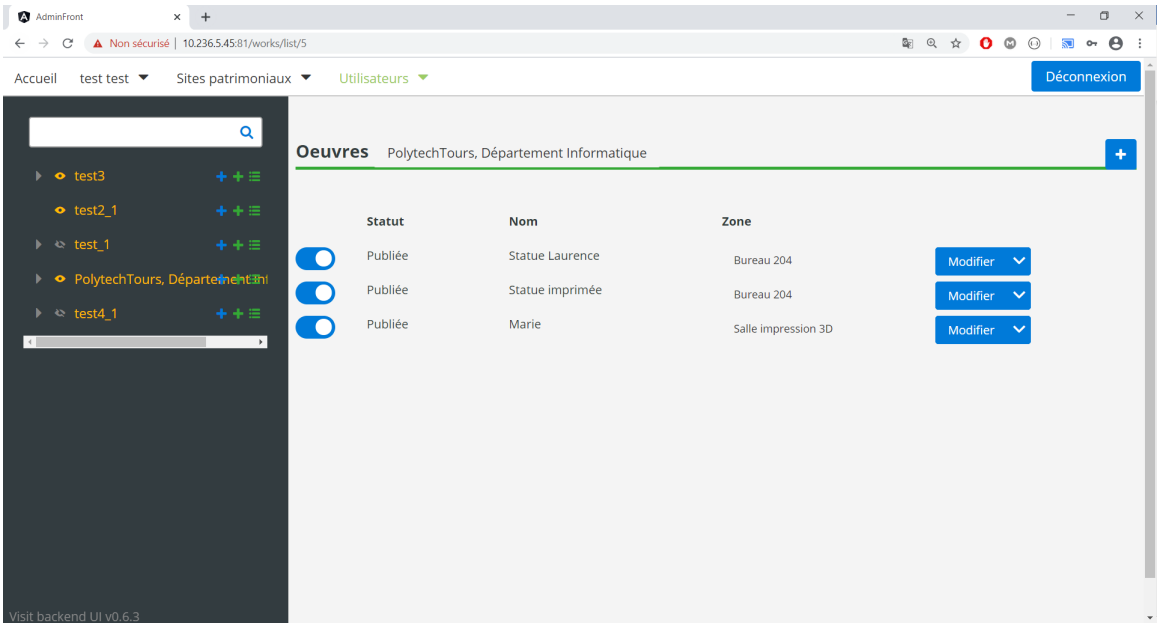


Figure 8 – *Page de consultation des œuvres d'un site (Responsable de site)*

2.7 Page de création d'une nouvelle œuvre de son site patrimonial

De même que pour l'ajout d'une Nouvelle zone comme nous avons pu l'évoquer précédemment il est possible d'ajouter une nouvelle œuvre en sélectionnant l'option « Nouvelle œuvre » dans le menu « Sites patrimoniaux » et le sous menu correspondant à votre site. Que vous décidiez de modifier une œuvre via le bouton « Modifier » dans la page de consultation des œuvres ou que vous souhaitiez ajouter une nouvelle œuvre vous serez redirigés vers une page tel que l'interface exposée dans l'image suivante.

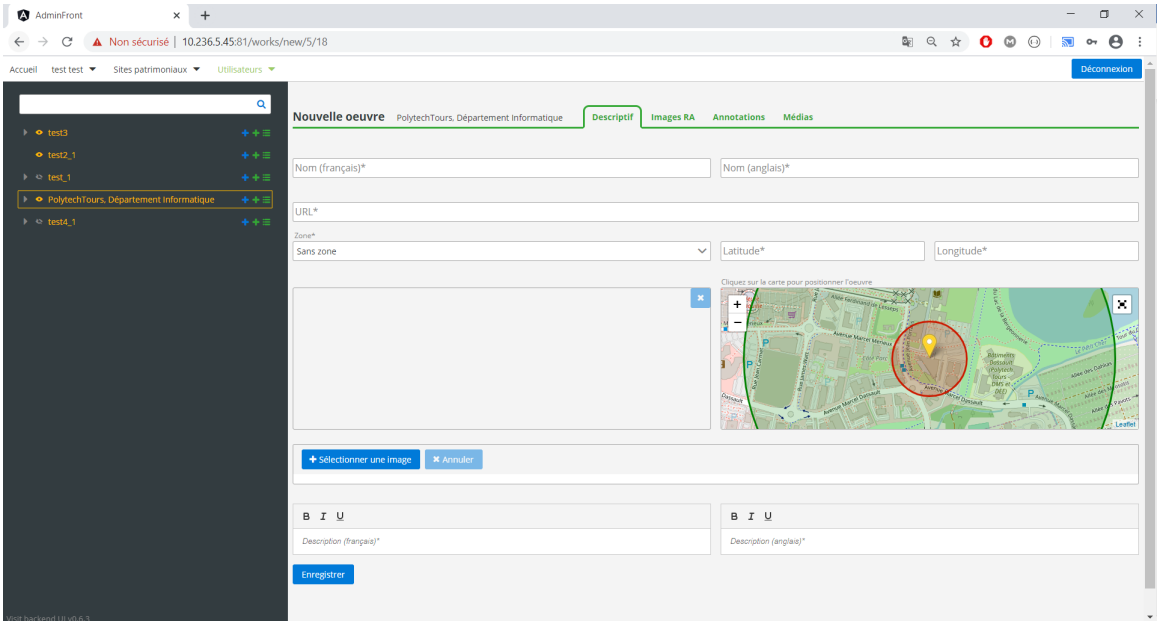


Figure 9 – *Page d'ajout d'une œuvre d'un site (Responsable de site)*

2.8 Page d'ajout image RA à une œuvre de son site patrimonial

Dans la même page de création ou de modification d'une œuvre il est possible d'ajouter du contenu tel qu'une image RA en accédant à l'onglet images RA de la page précédemment décrite. A la sélection d'une image un contrôle de qualité de cette dernière sera effectué pour être sûr que l'image ajoutée soit valide (assez bien définie) pour que l'on puisse l'utiliser avec de la réalité augmentée. (qualité supérieure ou égale à 75/100)

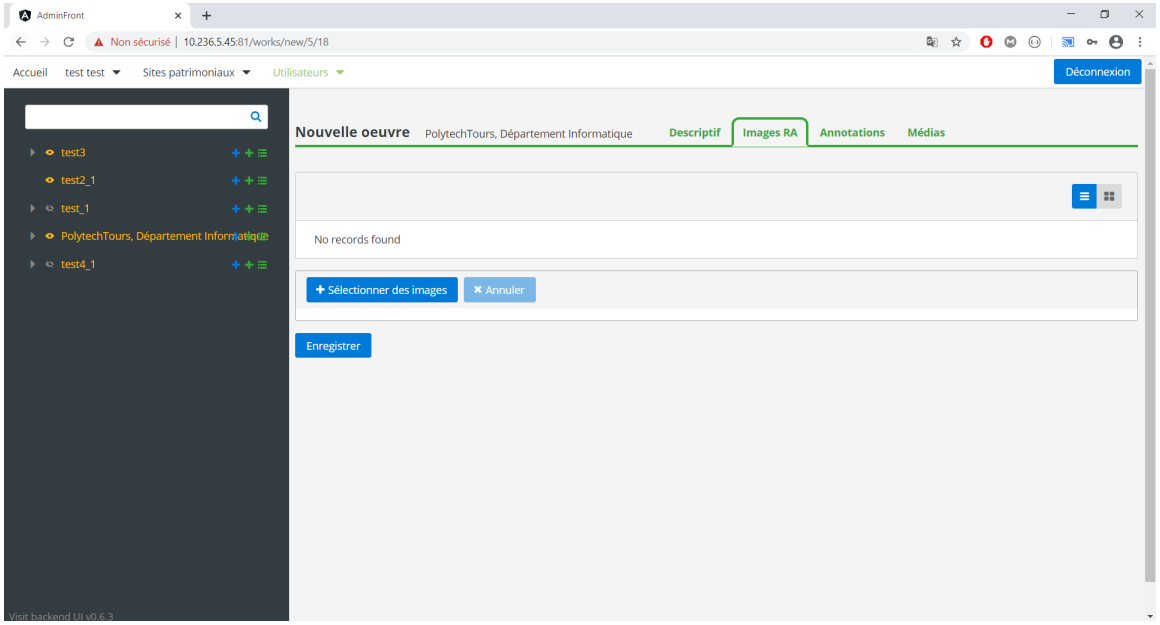


Figure 10 – *Page d'ajout d'une image RA lors de l'ajout ou modification d'une œuvre d'un site (Responsable de site)*

2.9 Page d'ajout de médias à une œuvre de son site patrimonial

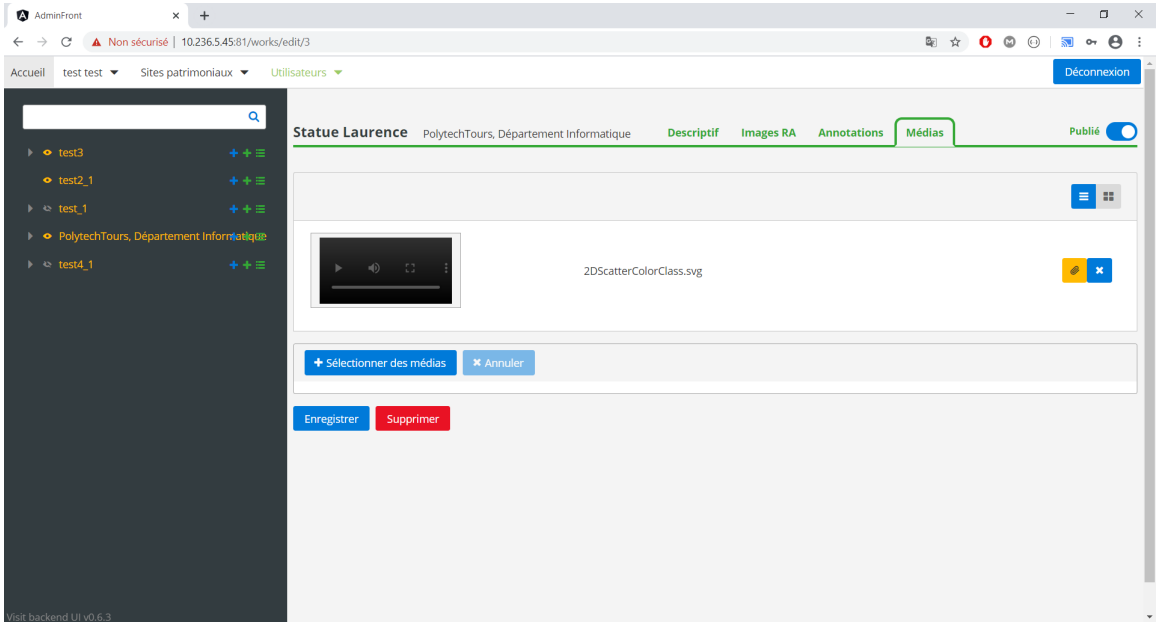


Figure 11 – *Page d'ajout d'un media lors de l'ajout ou modification d'une œuvre d'un site (Responsable de site)*

Toujours dans la même page de création ou de modification d'une œuvre, l'utilisateur peut ajouter des médias à une œuvre tel qu'un enregistrement audio par exemple. Il suffit pour cela de se rendre dans l'onglet « Médias » de la page de création ou de modification d'une œuvre. Sélectionnez un nouveau média en appuyant sur « Sélectionner des médias » et valider son ajout à l'aide du bouton « Enregistrer ».

2.10 Page d'ajout d'annotations à une œuvre de son site patrimonial

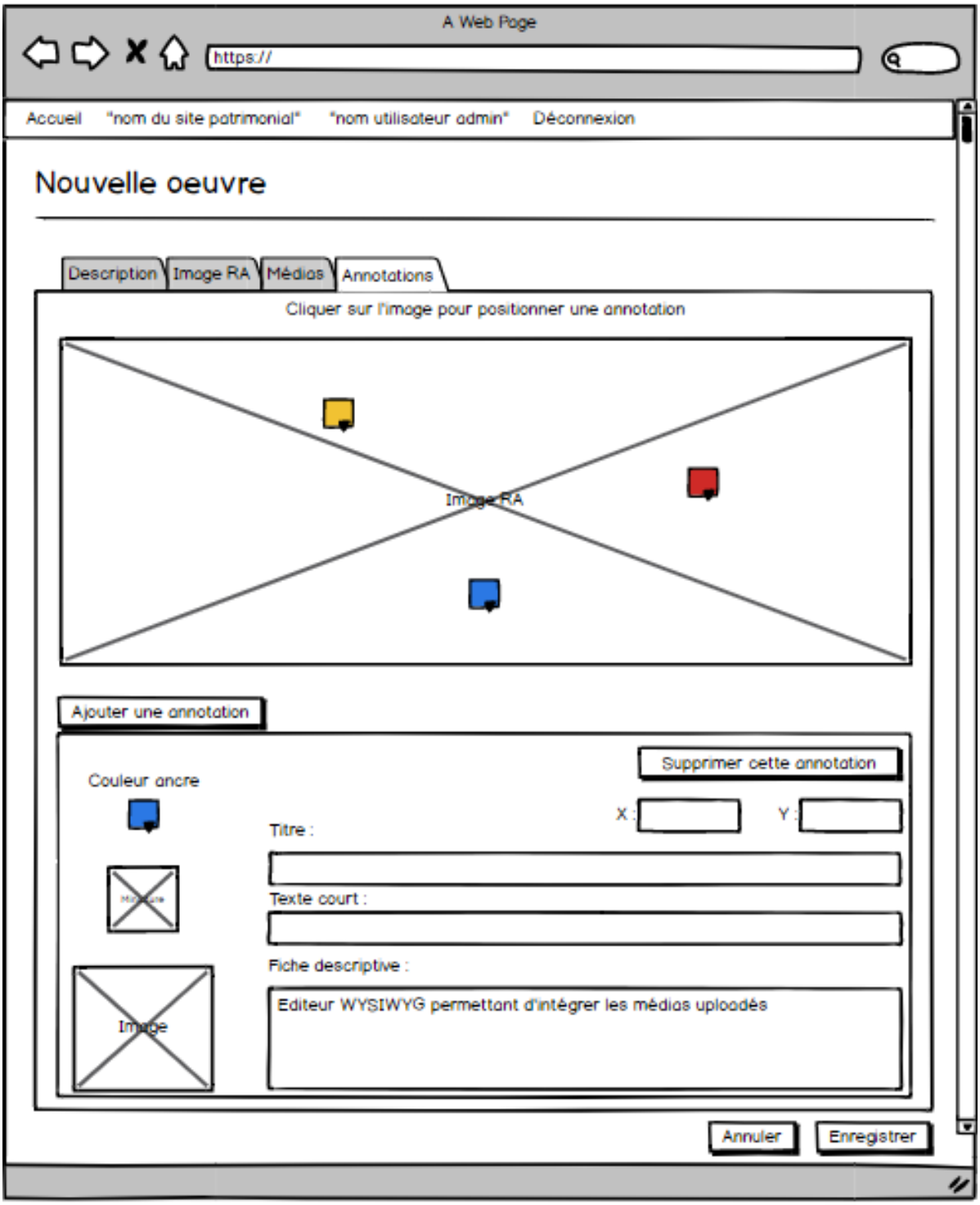


Figure 12 – *Page de l'interface de Visualisation d'ajout œuvres ancres/balises (Responsable de site)*

Dans le même esprit d'ajout de contenu à une œuvre il est également possible toujours depuis cette page de création/modification d'une œuvre, d'ajouter des annotations à une œuvre (balises contenants textes, audio ou autres pour agrémenter la visite d'un touriste et lui décrire plus précisément l'œuvre).

L'interface correspondant à cette partie n'est pas encore terminée. Nous utiliserons pour la démonstration des fonctionnalités associées à cet onglet un mockup (schéma) proche de l'implémentation finale de cette partie pour étayer notre explication.

Le schéma en question est visible ci-dessus et présente une interface proposant la personnalisation des balises déposées sur une œuvre. Par « déposé » on entend le fait qu'il sera possible pour un utilisateur de faire du « glissé – déposé » avec une balise et ainsi faciliter la manipulation de ces éléments. Chaque balise sera personnalisable à l'aide de diverses options comme un titre, un texte court, une description ou encore une couleur permettant de la différencier des autres.

3 Administrateur

Bien entendu l'administrateur du site web possède des droits lui permettant de réaliser toutes les actions précédemment énoncées pour un responsable de site. Ce dernier possède néanmoins la possibilité d'effectuer d'autres actions.

3.1 Page d'accueil Administrateur

Pour commencer la page d'accueil de l'administrateur du site web est légèrement différente de celle d'une responsable de site patrimonial connecté. Il a accès aux onglets disponibles pour un responsable de site ainsi qu'un onglet propre à son statut, l'onglet « Utilisateurs ». Ce dernier permet à l'administrateur de visualiser la liste des utilisateurs du site sinon d'en créer un nouveau.

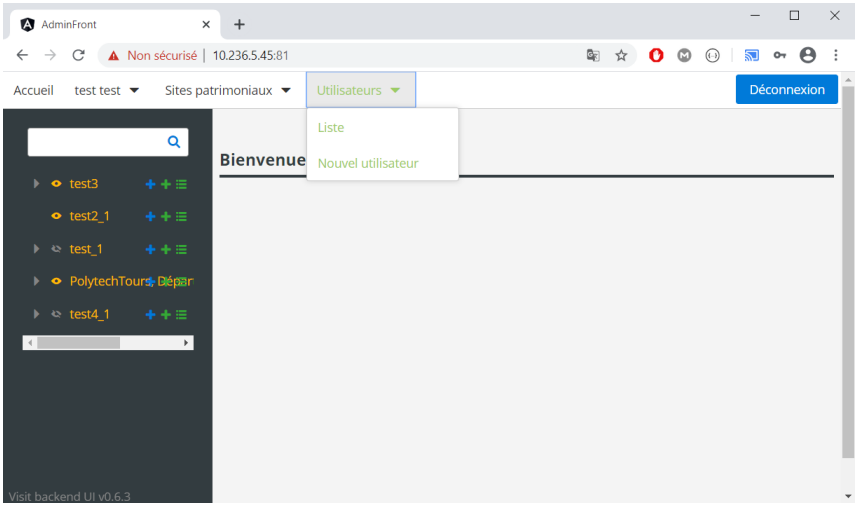


Figure 13 – *Page d'accueil Authentifié (Administrateur)*

3.2 Page de consultations de utilisateurs du site web

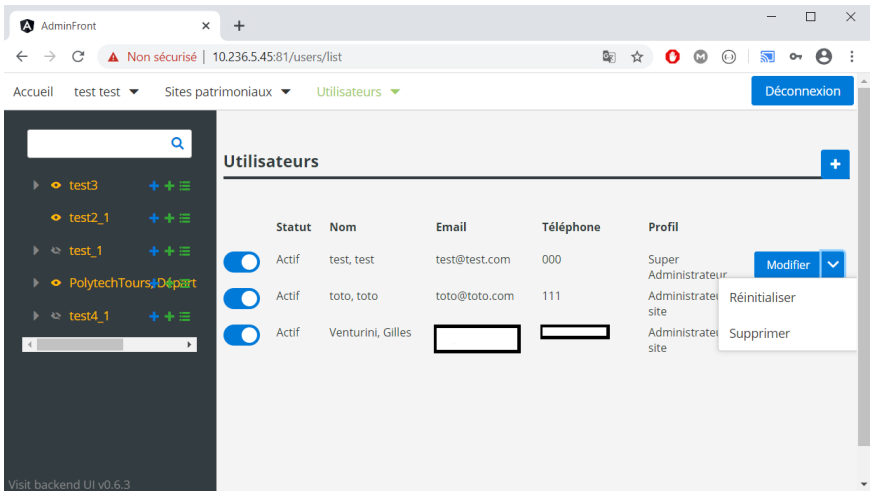


Figure 14 – *Page de consultation des utilisateurs du site web(Administrateur)*

L'image ci-dessus présente l'interface accessible en se rendant sur la liste des utilisateurs dans l'onglet « Utilisateurs ». Il permet de définir entre autres si un utilisateur est actif ou non. Cette distinction permet d'autoriser la connexion d'un utilisateur si son compte est à l'état actif. L'accès à la page de consultation, de suppression ou modification d'un utilisateur se fait par l'intermédiaire du menu déroulant de l'élément de la liste choisi.

3.3 Page de création d'un nouvel utilisateur

L'autre option exclusivement disponible pour l'administrateur du site est la possibilité d'ajouter des nouveaux utilisateurs en sélectionnant l'option « Nouvel Utilisateur » du menu déroulant « Utilisateurs ». Il est alors proposé de saisir les informations personnelles d'un nouvel utilisateur tel que le prénom, nom, email et téléphone. C'est également à la création de cet utilisateur que l'administrateur défini les droits d'accès du nouvel arrivant (Administrateur de Site = Responsable de Site ou Super Administrateur)

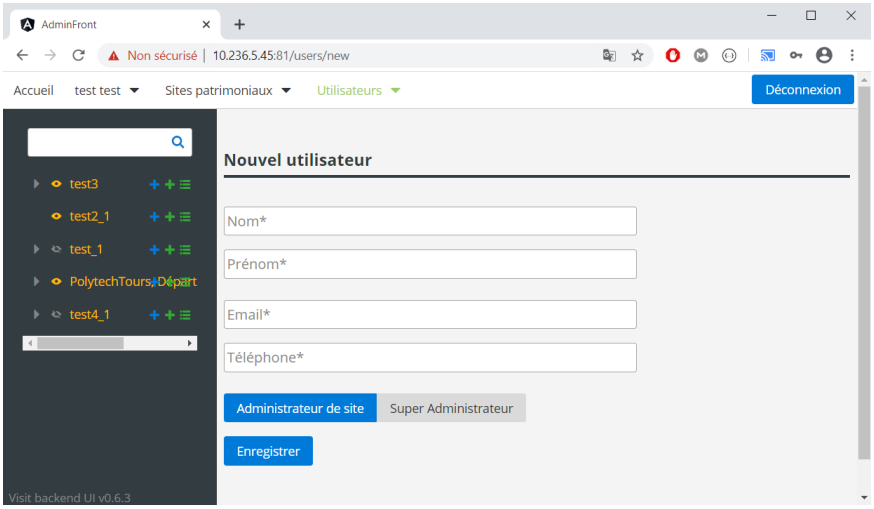


Figure 15 – *Page d'ajout d'un nouvel utilisateur du site web(Administrateur)*

3.4 Page de consultations de tous les sites patrimoniaux

Tout comme le responsable d'un site patrimonial le super administrateur peut visualiser l'ensemble des sites recensés dans la base de données à ceci près que ses droits ne s'arrêtent pas à la consultation et ce même s'il n'est pas défini comme gérant d'un site patrimonial.

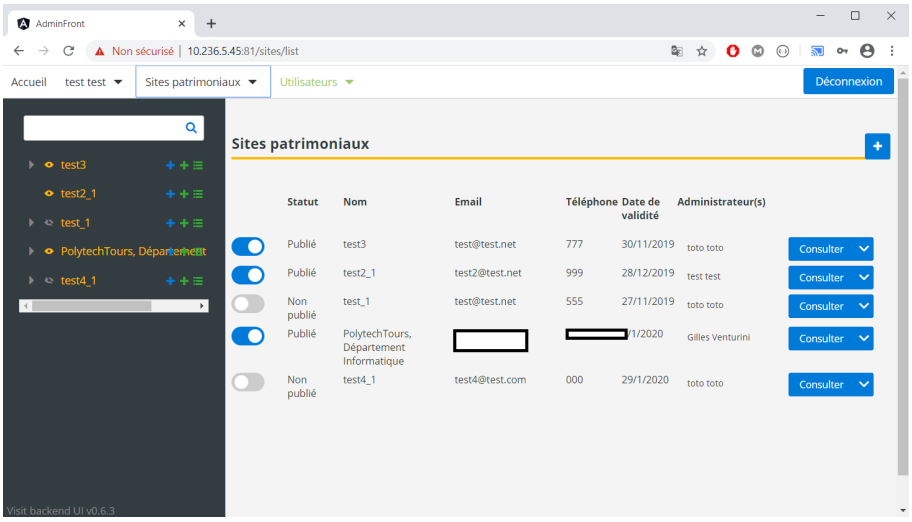


Figure 16 – *Page de consultation des sites patrimoniaux recensés sur le site web(Administrateur)*

3.5 Page de création d'un nouveau site patrimonial

Comme nous avons pu l'évoquer précédemment le super admin est donc en capacité de créer un nouveau site patrimonial ou encore modifier le contenu d'un site déjà existant. Il a en quelque sorte un contrôle total sur les fonctionnalités qu'il est possible de réaliser sur l'application web d'administration VISIT.

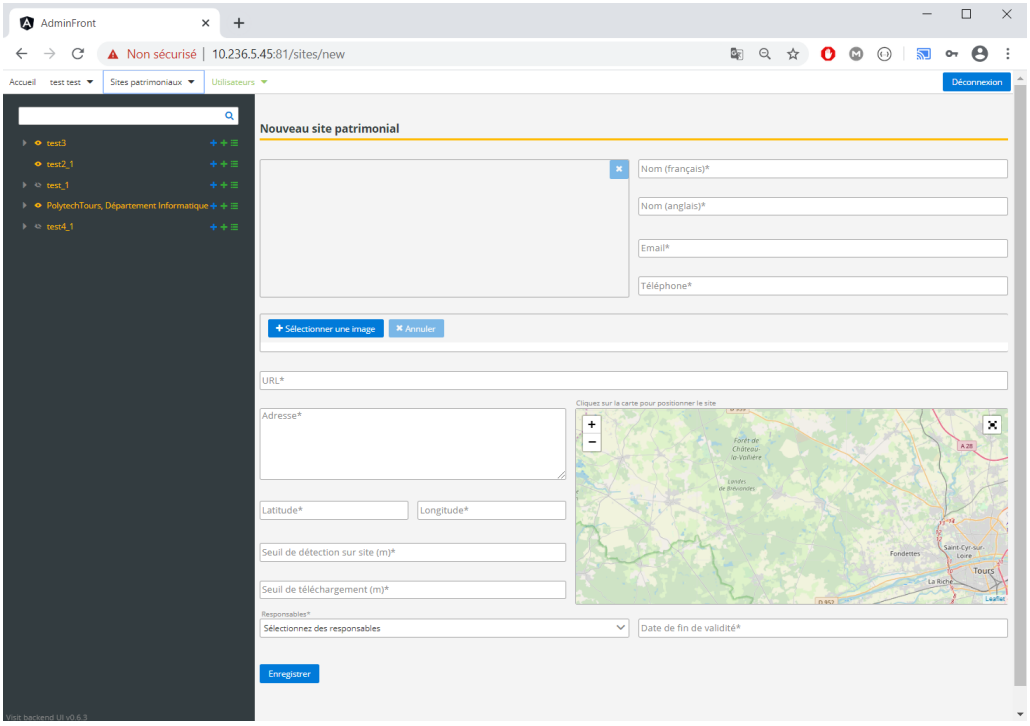


Figure 17 – *Page d'ajout d'un nouveau site patrimonial(Administrateur)*

H Tests et Qualité de codage

Cette partie recense toutes les méthodes et pratiques ayant été adoptées pendant le développement de ce projet afin de garantir la qualité et la pertinence des solutions choisies.

1 Design Pattern

Nous avons utilisé pour la partie back-end le design pattern Symfony PHP ainsi que le Framework de création d'api PHP « api-platform ». Pour la partie front-end, nous nous sommes servis du Framework Angular basé sur du Javascript. L'utilisation de ces technologies a permis de mettre en place des architectures de codage particulières et normées permettant ainsi de contribuer à la mise en place d'un code de qualité pouvant être facilement repris en connaissance des frameworks et patterns précédemment cités.

2 Structure du code

Concernant la structure du code, nous avons choisi de respecter les conventions imposées sinon conseillées par les frameworks et technologies utilisées. Les 3 sous parties suivantes correspondent aux respectivement aux parties « Architecture Back-end », « Architecture Front-end » et « Architecture du code de création de la commande de package» du cahier du développeur. Il est néanmoins utile de rappeler les caractéristiques architecturales mises en place pour ce projet, ces dernières ayant contribuer à garantir un code de qualité.

2.1 Architecture - librairies Back-end

Pour rappel l'API REST constituant la partie Back-end de l'application a été réalisée à l'aide des technologies PHP Symfony et API Platform. Cette partie du projet VISIT est architecturée de la manière suivante.

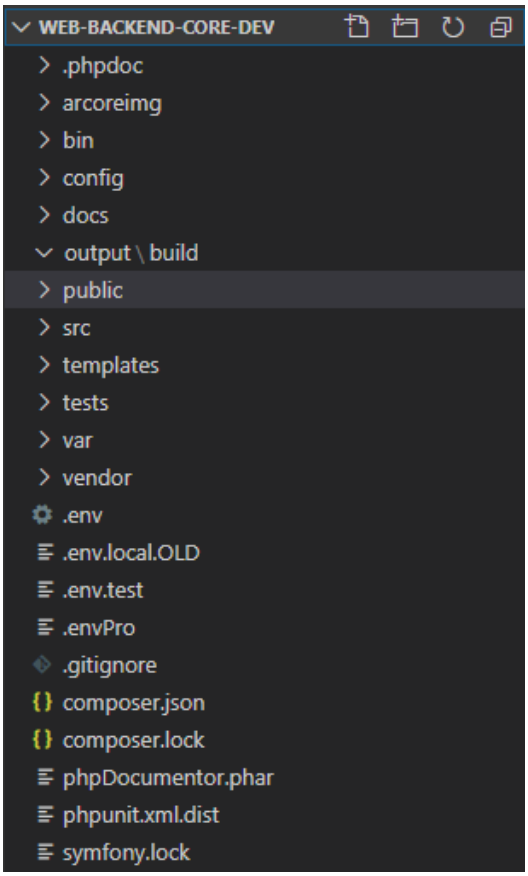


Figure 1 – *Architecture Back-end*

Les dossiers de configuration du projet propres aux Frameworks, utilitaires et langages ayant servi pour créer l'API se situent dans les dossiers « /bin », « /config », « /vendor » et « /public/-bundle ». Le dossier « /var » comporte les informations propres à l'utilisateur tel que les logs serveur. Le dossier « /arcoreimg » contient des utilitaires et librairies permettant de gérer les images AR que nous manipulons dans ce projet selon l'OS utilisé. Les sources de l'ensemble du projet sont présentes dans le dossier « /src ». De même pour les tests qui sont dans le dossier associé « /tests ».

Concernant la fonctionnalité de création de package de données à destination de l'application mobile comme nous avons pu l'évoquer précédemment. Cette commande va chercher les médias situés sur le serveur dans le dossier « /public/media », les JsonStates correspondants, présents dans le dossier « /public/state » et les bases de données AR créés à partir des fichiers précédents et stockés dans « /public/arcore» pour créer une archive zip « debug » et « release » de chaque site dans le dossier « /public/package ».

2.2 Architecture - librairies Front-end

Pour l'architecture utilisée pour l'implémentation de la partie front-end, nous avons suivi celle proposée par le Framework angular, cette dernière se présentant de la manière suivante.

Les dossiers de configuration du projet propres aux Frameworks, utilitaires et langages ayant servi pour créer l'interface utilisateur se situent dans les dossiers « /node_modules », « /e2e ». Les sources de l'ensemble du projet sont présentes dans le dossier « /src ».

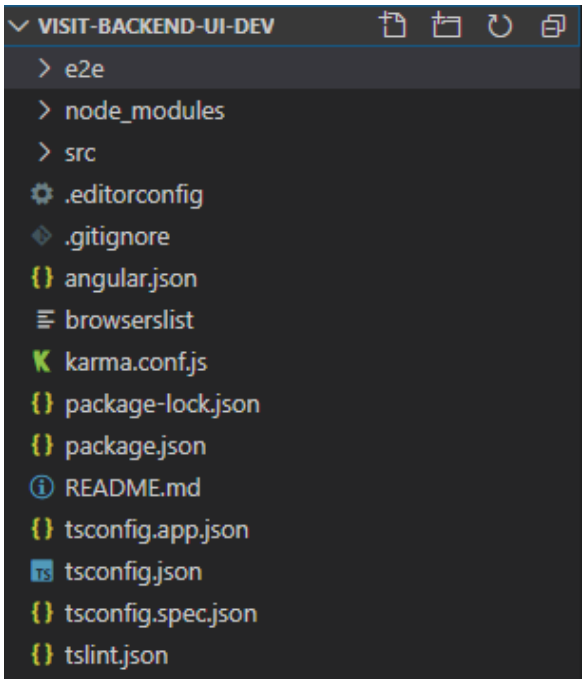


Figure 2 – *Architecture Front-end*

2.3 Architecture du code de création de la commande de package

Cette sous partie vise à faciliter la navigation du développeur dans les différents fichiers s'il souhaite trouver le code constituant le cœur de la création de package de données à destination de l'application mobile. Ledit code est présent dans la partie back-end du serveur, dans le dossier « /src » conformément aux normes choisies.

Il se décompose en deux sous fichiers, lesquels étant « /src/Command/CreateBundleCommand.php » et « /src/Service/PackageService.php ». Le premier étant la déclaration de la commande php suivant le Framework PHP Symfony et le second constituant le cœur de la création des différents packages.

On se sert également des fichiers se rapportant aux différents « repository » des tables de la base de données pour créer nos packages ainsi que le fichier « /src/Service/SiteStateService.php » qui établit le lien entre les JsonStates présents sur le serveur dans le dossier « /public/state » et les siteStates enregistrés dans la base.

3 Lisibilité des productions

Concernant la structuration du code, veuillez-vous référer aux sous parties précédentes « Architecture et librairies » expliquant en détails l'architecture de chacune des parties (back et front) de l'application.

3.1 Conditions de nommage

Les conditions de nommage adoptées ont été fixées dans la phase pré-développement afin d'implémenté les solutions proposées sous une vision commune compréhensible pour tous. Les règles de nommage étant les suivantes :

- Les noms des méthodes et fonctions présentes dans le code de l'application web dans sa totalité sont en anglais, commence par une minuscule et chaque nouveau mot par la suite commence par une majuscule suivant le schéma suivant : « getName() »,
- Les noms des variables utilisés dans notre application suivent la même logique de nommage que les fonctions et méthodes : noms en anglais commençant par une minuscule et suivi par des majuscules à chaque nouveau mot,
- Les commentaires des fonctions générées automatiquement par les fonctions doctrine de php par exemple, sont en anglais. Pour les fonctions ajoutées par les développeurs à la main, les commentaires sont en français.

3.2 Documentations

Des documents à l'intention des utilisateurs de l'application web et des développeurs chargés de la maintenance et/ou la reprise du code ont été rédigés. Ces deux documents sont présents respectivement sous les noms de « Guide utilisateur » et « Cahier du développeur » au sein de ce même rapport. Une documentation Doxygene a également été générée concernant le code du serveur back-end (API, commande de génération des packages et autres utilitaires permettant de gérées les images RA).

4 Déploiement et maintenance des productions

Comme nous avons pu l'évoquer dans la partie bases méthodologiques, nous avons choisi d'utiliser un dépôt Git pour versionner notre code. Par l'intermédiaire de l'interface de Gestion GitLab il est ainsi possible d'assurer une maintenance efficace de l'application et un suivi de son développement. GitLab propose également des outils permettant de mettre en place une intégration continue. Ayant dû faire face aux contraintes temporelles nous n'avons pas pu aller jusqu'à cet aspect de contrôle de fonctionnement de cette application.

Le déploiement se fait actuellement sur un serveur de l'université par l'intermédiaire d'une image docker qui facilite ainsi l'intégration des différents composants que comptent notre application.

Un guide d'installation est présent sur le dépôt et/ou fait également partie du cahier de développeur fourni pour faciliter la reprise du projet. Le guide d'installation fait part de la manière de charger les dépendances en cas de déploiement à savoir la saisie de commande « composer install » (API) et « npm install » (Interface Web) suivies des commandes de démarrage serveur. Les seules saisies de ses commandes doivent suffirent à lancer le projet ce qui facilite le déploiement de l'application.

5 Validation des productions

Concernant les tests et autres dispositifs misent en place pour assurer la validation des solutions implémentées, nous n'avons pu mettre en place qu'un contrôle à minima fonctionnel devant faire face aux contraintes temporelles de développement.

Concernant la tâche de création des packages ARCore à destination de l'application mobile, tâche décrite dans le cahier de spécification de ce projet, l'automatisation des tests unitaires et fonctionnelles n'a pu aboutir qu'à un contrôle du contenu d'un package du point de vu « fichier » et non « data ». Le lancement des tests automatisés se fait à la racine du serveur de l'API REST en lançant la commande « php bin/phpunit » en sachant qu'il est nécessaire que le serveur de la base de données soit à minima lancé pour que les tests fonctionnent.

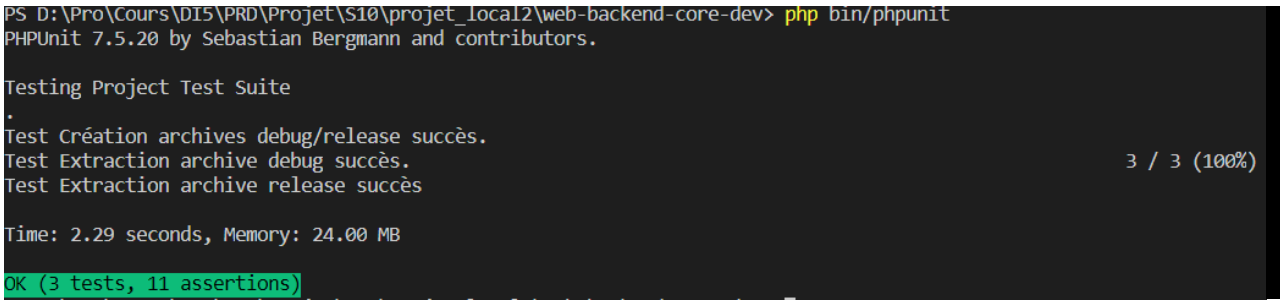


Figure 3 – *Lancement de commande de tests création package de données*

Un cahier de recette a cependant été rédigé en réalisant des tests purement fonctionnels concernant la création de ces package de données. Lesquels se basant sur des fichiers de données complets et complexes pour couvrir le plus de cas particuliers possible et vérifier l'ensemble des contraintes mises en place dans l'algorithme de création des packages.

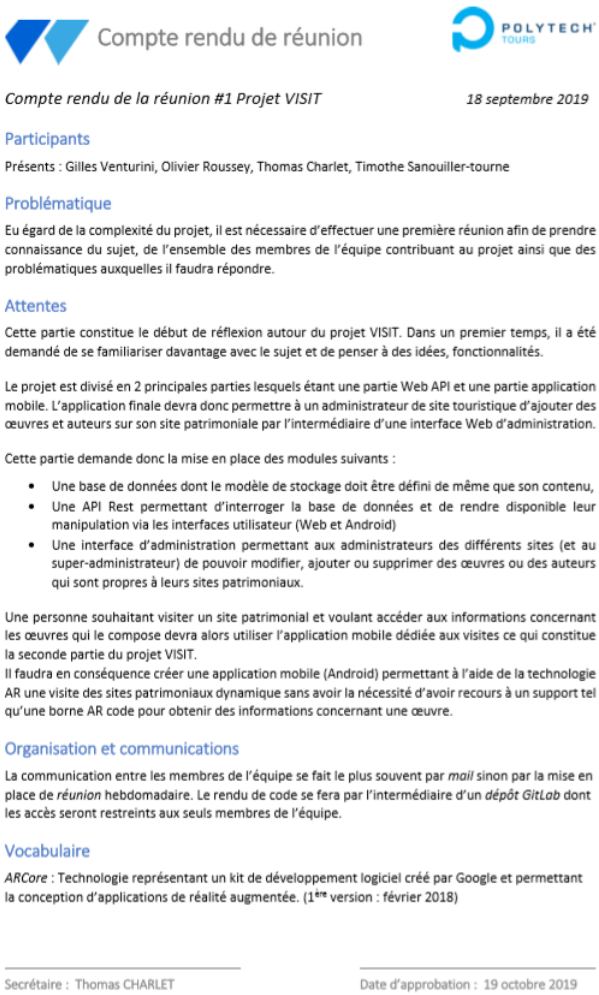
Ledit cahier de recette contrôle également les actions pouvant être effectuées depuis l'interface d'administration des sites patrimoniaux. Son contenu est présent en annexe à la suite de cette partie sinon dans le document correspondant au rapport de tests.

N°	Module	Type Test	Prérequis	Action	Attendu	Résultat
1	BACK	FONCTIONNEL		Test Creation "User" valide [POST]	Nouvel "User" créé puis récupéré (intégrité données vérifiée), réponse 201 attendue. idUser généré 17	OK
2	BACK	FONCTIONNEL		Test Get "User" [GET]	"User" 17 récupéré (intégrité données vérifiée), réponse 200 attendue	OK
3	BACK	FONCTIONNEL		Test Creation "Site" valide [POST]	Nouveau "Site" créé puis récupéré (intégrité données vérifiée), réponse 201 attendue. idSite généré 5	OK
4	BACK	FONCTIONNEL		Test Get "Site" [GET]	"Site" 5 récupéré (intégrité données vérifiée), réponse 200 attendue	OK
5	BACK	FONCTIONNEL		Test Creation "Zone" [POST]	Nouvelle "Zone" créée puis récupérée (intégrité données vérifiée), réponse 201 attendue	OK
6	BACK	FONCTIONNEL		Test Get "Zone" [GET]	"Zone" récupérée (intégrité données vérifiée), réponse 200 attendue	OK
7	BACK	FONCTIONNEL		Test Create "MediaObject" [POST]	Nouveaux "MediaObject" créés puis récupérées (intégrité données vérifiée), réponse 201 attendue.	OK
8	BACK	FONCTIONNEL		Test Fonctionnalité évaluation de la qualité d'une image	Indication de la qualité du media en paramètre	OK
9	BACK	FONCTIONNEL		Test Get "MediaObject" [GET]	"MediaObject" précédemments créés sont récupérées (intégrité données vérifiée), réponse 200 attendue	OK
10	BACK	FONCTIONNEL		Test Create "CreativeWork" [POST]	Nouveau "CreativeWork" créé puis récupéré (intégrité données vérifiée), réponse 201 attendue.	OK
11	BACK	FONCTIONNEL		Test Get "CreativeWork" [GET]	"CreativeWork" récupéré (intégrité données vérifiée), réponse 200 attendue	OK
12	BACK	FONCTIONNEL		Test Update état published d'un "CreativeWork" [PUT]	"CreativeWork" modifié avec changement d'état de publication de l'oeuvre, réponse 200 attendue	OK
13	BACK	FONCTIONNEL		Test Create Add ImageAR à "CreativeWork" [PUT]	"CreativeWork" modifié avec ajout imageRA en paramètre (intégrité données vérifiée), réponse 200 attendue	OK
14	BACK	FONCTIONNEL		Test Create Add Annotation à une "AnnotatedImage" d'un "CreativeWork" [PUT]	"CreativeWork" modifié avec ajout annotation en paramètre (intégrité données vérifiée), réponse 200 attendue	OK
VALIDATION	BACK	FONCTIONNEL		Tests Fonctionnels - Création d'un site "complexe" et génération/modification du Json StateSite validé	Tests Fonctionnels validés	14/14 OK
15	BACK	FONCTIONNEL		Test Creation de package pour tous les sites de la base "creat-package all"	Packages générés contenenant bases AR, medias et json state de version debug et release de tous les sites	OK
16	BACK	FONCTIONNEL		Test Creation de package pour tous les sites de la base "creat-package idSite" (idSite = 5 dans test)	Package généré contenenant bases AR, medias et json state de version debug et release du site en paramètre	OK
VALIDATION	BACK	FONCTIONNEL		Tests Fonctionnels - Création des archives selon les contraintes du problème	Tests Fonctionnels validés	2/2 OK
17	BACK	FONCTIONNEL		Test Ajout de la contrainte de la qualité supérieure ou égale 75 pour génération archive	Les bases AR ne contiennent que des images de qualité supérieure ou égale à 75	OK
18	BACK	FONCTIONNEL		Test Contenu de la base AR pour archive Debug d'un site	La base AR contient l'ensemble des images RA que l'oeuvre soit à l'état published true ou false	OK
19	BACK	FONCTIONNEL		Test Contenu de la base AR pour archive Release d'un site	La base AR ne contient que les images RA pour une oeuvre à l'état published true	OK
20	BACK	FONCTIONNEL		Test Contenu dossier media pour archive Debug d'un site	Le dossier media contient du site, des zones si site publié, des annotations des creativeWorks publiés (+ simplification)	OK
21	BACK	FONCTIONNEL		Test Contenu dossier media pour archive Release d'un site	Le dossier media contient du site, des zones si site publié, des annotations des creativeWorks publiés (+ simplification)	OK
22	BACK	FONCTIONNEL		Test Contenu dossier state pour archive Debug d'un site	Liste Le dossier contient le jsonState simplifié administrators site supprimée, conservation de l'attribut validity date stringimage descriptif du site ->enlever name, isImage et quality, image descriptif d'une zone -> enlever name, isImage et quality, image descriptif d'une oeuvre -> enlever name, isImage et quality, zones qui comportent pas d'anotations à enlever, oeuvres pas d'annotations à enlever également, dans les oeuvres -> retirer le champ media, pas besoin d'image annotée par default (isDefault=true)enlever champ is Default pour images annotées)	OK
23	BACK	FONCTIONNEL		Test Contenu dossier state pour archive Release d'un site	Liste Le dossier contient le jsonState simplifié administrators site supprimée, conservation de l'attribut validity date stringimage descriptif du site ->enlever name, isImage et quality, image descriptif d'une zone -> enlever name, isImage et quality, image descriptif d'une oeuvre -> enlever name, isImage et quality, zones qui comportent pas d'anotations à enlever, oeuvres pas d'annotations à enlever également, dans les oeuvres -> retirer le champ media, pas besoin d'image annotée par default (isDefault=true)enlever champ is Default pour images annotées) + spécificités de version release	OK
VALIDATION	BACK	FONCTIONNEL		Tests Fonctionnels - Contraintes devant être respectées par les archives générées	Tests Fonctionnels validés	7/7 OK

Figure 4 – *Cahier de recette des tests effectués*

I Compte rendu de réunion

1 Compte rendu de réunion 1



Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #1 Projet VISIT *18 septembre 2019*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Eu égard de la complexité du projet, il est nécessaire d'effectuer une première réunion afin de prendre connaissance du sujet, de l'ensemble des membres de l'équipe contribuant au projet ainsi que des problématiques auxquelles il faudra répondre.

Attentes

Cette partie constitue le début de réflexion autour du projet VISIT. Dans un premier temps, il a été demandé de se familiariser davantage avec le sujet et de penser à des idées, fonctionnalités.

Le projet est divisé en 2 principales parties lesquels étant une partie Web API et une partie application mobile. L'application finale devra donc permettre à un administrateur de site touristique d'ajouter des œuvres et auteurs sur son site patrimoniale par l'intermédiaire d'une interface Web d'administration.

Cette partie demande donc la mise en place des modules suivants :

- Une base de données dont le modèle de stockage doit être défini de même que son contenu,
- Une API Rest permettant d'interroger la base de données et de rendre disponible leur manipulation via les interfaces utilisateur (Web et Android)
- Une interface d'administration permettant aux administrateurs des différents sites (et au super-administrateur) de pouvoir modifier, ajouter ou supprimer des œuvres ou des auteurs qui sont propres à leurs sites patrimoniaux.

Une personne souhaitant visiter un site patrimonial et voulant accéder aux informations concernant les œuvres qui le compose devra alors utiliser l'application mobile dédiée aux visites ce qui constitue la seconde partie du projet VISIT.
Il faudra en conséquence créer une application mobile (Android) permettant à l'aide de la technologie AR une visite des sites patrimoniaux dynamique sans avoir la nécessité d'avoir recours à un support tel qu'une borne AR code pour obtenir des informations concernant une œuvre.

Organisation et communications

La communication entre les membres de l'équipe se fait le plus souvent par *mail* sinon par la mise en place de *réunion* hebdomadaire. Le rendu de code se fera par l'intermédiaire d'un *dépôt GitLab* dont les accès seront restreints aux seuls membres de l'équipe.

Vocabulaire

ARCore : Technologie représentant un kit de développement logiciel créé par Google et permettant la conception d'applications de réalité augmentée. (1ère version : février 2018)

Secrétaire : Thomas CHARLET

Date d'approbation : 19 octobre 2019

Figure 1 – *Compte rendu de réunion 1*

2 Compte rendu de réunion 2

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #2 Projet VISIT *2 octobre 2019*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Après une première phase de recherche autour des fonctionnalités à développer pour répondre aux besoins évoqués lors de la précédente réunion, une première version d'un modèle de données à adopter est proposée et doit faire l'objet d'une discussion.

Attentes

En réponse à la problématique ci-dessus le modèle de donnée a été revu. Le choix d'un système de stockage par la mise en place d'un modèle objet a été validé par l'équipe. L'évocation de nouvelles problématiques de conception nous a poussé à envisager l'ajout de caractéristiques aux différentes entités qui composent la base.

Les fonctionnalités n'ayant pas été attribuées exclusivement à une personne, une nouvelle phase de réflexion et de tâches à effectuer a été réalisée suivant les points suivants :

- Compléter le modèle de base de données pour l'intégration des nouvelles caractéristiques
- Effectuer des recherches concernant l'architecture *Symfony* et son fonctionnement dans le cadre de la création d'une API Rest exploitant la base de données. L'objectif étant de débuter la création d'un prototype d'API exploitant (***Nouveau)***
- Etudier la question du contrôle qualité des photos prise par un responsable dans le but de créer une image AR. Cette information permettrait de calibrer l'application mobile pour aider l'utilisateur dans son comportement à adopter pour pouvoir voir l'œuvre en VR de manière correcte.
- Analyser des scripts AR Core de manière à comprendre comment cette technologie analyse une image.
- Réfléchir à un système permettant de différencier les images/œuvres disponibles en mode « release » ou « debug » ainsi qu'aux problèmes liés à cette distinction.
- Penser à une manière de généraliser les données en transaction entre les différents modules.

Vocabulaire

Symfony : Ensemble de composants et Framework MVC libre PHP permettant de faciliter et d'accélérer le développement web à l'aide de fonctionnalités modulables et adaptables mis à disposition des développeurs.

API REST : L'appellation REST (REpresentational State Transfer) ou RESTFULL est un ensemble de contraintes utilisées pour créer des services web sous la forme d'un style d'architecture logicielle propre. Elle permet entre autres d'établir une interopérabilité entre les ordinateurs sur Internet.

Api-platform : Est un ensemble d'outils et de fonctionnalités permettant de construire et d'exploiter les web APIs.

Secrétaire : Thomas CHARLET

Date d'approbation : 3 octobre 2019

Figure 2 – *Compte rendu de réunion 2*

3 Compte rendu de réunion 3

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #3 Projet VISIT *16 octobre 2019*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Un prototype de base de données et un début d'API REST utilisant Symfony/API platform ont été réalisés. Une liste de fonctionnalités concernant la partie web API a été fixé, il reste néanmoins à établir les fonctionnalités que je serai chargé de mettre en place dans le cadre de l'instrumentation de ce module.

Existant

Un modèle de base de données a été défini. L'élaboration de l'API a commencé et des mockups concernant les interfaces d'administration et de l'application mobile ont été effectués. La poursuite de recherches en lien avec les besoins de l'application ont fait surgir de nouvelles problématiques.

Attentes

A la suite de cette réunion, certaines fonctionnalités ont été évoquées comme pouvant faire l'objet de tâches que je serai amenée à réaliser au cours du projet, lesquelles sont :

- Création d'une interface de manipulation d'ancres et de balises sur des images importées par l'administrateur d'un site patrimonial.
    - Dans ce sens, il est suggéré de s'orienter vers des technologies telles que SVG et Canvas pour la manipulation de balises et d'encres sur une images 2D.
- Un exécutable qui pour chaque site associe des œuvres présentes dans une base de données AR Core.
- Effectuer des recherches sur Angular qui est le Framework sur lequel l'équipe va s'appuyer pour créer l'interface d'administration des sites patrimoniaux.

Le modèle de données a également été revu suivant les nouvelles idées et sera encore amené à évoluer tout au long du projet. De nouvelles pistes de recherche doivent également être explorer notamment concernant les points suivants :

- L'authentification et reconnaissance de l'identité utilisateur via l'application,
- La sécurisation des transactions et connections avec l'application,
- La distinction des modes « Release/Debug » de l'App et ce que cela implique pour l'utilisateur,

Vocabulaire

SVG : Format de données ASCII conçu pour décrire des graphiques vectoriels et basé sur XML.

Canvas : Cet élément est une composante HTML , spécification depuis HTML5 permettant d'effectuer des rendus dynamiques d'images bitmap via des scripts.

Angular : Angular est une réécriture complète de Angular JS. C'est donc un Framework JavaScript libre et open source développée par Google permettant de développer des pages Web.

Secrétaire : Thomas CHARLET

Date d'approbation : 17 octobre 2019

Figure 3 – *Compte rendu de réunion 3*

4 Compte rendu de réunion 4

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #4 Projet VISIT *24 octobre 2019*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Eu égard du nombre et de la complexité des fonctionnalités à réaliser dans le cadre de ce projet, il est nécessaire de déterminer les fonctionnalités qui vont m'être attribué parmi celles pointés du doigt lors des réunions précédentes.

Existant

Un modèle de base de données a été défini et une API Rest a été mis en place pour contacter la base. L'étude des besoins continue a permis de faire surgir de nouvelles problématiques et ainsi accroître le nombre de fonctionnalités devant être réalisées.

Attentes

Deux fonctionnalités à réaliser ont été déterminées et m'ont été attribuées officiellement , lesquelles sont les suivantes :

- Une Interface de planification des ancres/balises pour placement de texte et vidéo

sur les ancres, une image pour la représenter et un système pour symboliser un mini-texte vidéo téléphone ou vidéo en réalité augmenté. Il est possible d'envisager l'existence d'ancres sans balises.

Le but est de créer un système pour l'administration des ajouts d'ancres et de balises à une image.

Difficultés identifiées : _ Ancres/balises dans le cas de représentation d'objets 3D
_ Position d'une balise par rapport à une ancre, (idéal : balise devant l'ancre)

Technologies : Utilisation de *svg* pour vectorisation d'image, 2D ou Canvas

- Un exécutable qui pour chaque site associe des œuvres présentes dans une base de données AR Core

Mise en place exécutable qui pour un site prend les œuvres en mode debug et celles qui ne le sont pas pour envoyer toutes ces images AR core associé dans un fichier de destination précis.

→ Extraction du dossier image de la base de données
→ Utiliser « AR core img » pour la génération d'un fichier AR en base de données
→ Envoyer ce fichier à application, ce dernier permettant de retrouver l'appartenance d'une image à une œuvre sur l'application mobile.

Pour la distinction des œuvres « Debug » et « Release », On a l'obligation de générer le fichier pour toutes les images puis d'adopter la méthode suivante. Soit la mise en place des images qui nous intéresse dans un dossier temporaire avant de lancer la commande de génération du fichier AR. Il y aura donc un fichier « Debug » et un fichier « Release » par site.

Secrétaire : Thomas CHARLET

Date d'approbation : 25 octobre 2019

Figure 4 – *Compte rendu de réunion 4*

5 Compte rendu de réunion 5

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #5 Projet VISIT *06 novembre 2019*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Eu égard des dates de rendu des parties spécifications et veille technologique qui approche ? il est nécessaire de faire une réunion pour faire le point sur l'avancé des tâches en cours. Compte tenu de la date d'attribution des fonctionnalités qui m'ont été attribuées (24 octobre 2019), l'objectif serait de prioriser les aspects à explorer d'ici à la date de rendu du cahier des spécifications.

Existant

Un modèle de base de données a été défini et une API Rest a été mis en place pour contacter la base. Des recherches ont été effectuées concernant les fonctionnalités m'ayant été attribuées à savoir :

- Pour le système d'administration des ajouts d'ancres et de balises à une image, des recherches sur les technologies SVG sont en cours. Des difficultés ont été identifiées pour les objets 3D.
- La création d'un exécutable pour l'envoie d'un fichier à l'application mobile contenant les images AR pour chaque site.

Attentes

- Poursuivre les recherches concernant les fonctionnalités précédemment évoquées et analyser les nouvelles problématiques que celles-ci peuvent engendrer.
    - Priorisation du modèle 2D avec possibilité de mise à l'échelle demandé à la personne ayant pris la photo (norme à définir par cet utilisateur). Explorer la piste d'une échelle de valeur permettant de définir la distance entre la balise à l'image pour répondre à la problématique des objets 3D.
- Effectuer des recherches concernant la fonctionnalité se rapportant à la détection de la qualité d'une image AR Core. L'existence d'un exécutable permettant de répondre à ce besoin a été évoqué. ***(Nouveau)***
- Communiquer une version de l'interface d'administration de l'application Web, sous forme de mockups ainsi qu'un rapport des problématiques soulevées au membre de l'équipe dans l'objectif de faire surgir de nouvelles idées pour la conception de l'interface.
- Avancer sur les spécifications et établir une nouvelle version du cahier de spécifications plus adapté aux fonctionnalités confiées.

Dates communiquées

- *14 novembre 2019* : Rendez-vous retour avancé de Projet N.Ragot,
- 27 novembre 2019 : Rendez-vous retour avancé de Projet Sopra Steria,
- *11 et 12 décembre 2019* : Soutenance de Projet sur les parties CAS et veille technologique.

Secrétaire : Thomas CHARLET

Date d'approbation : 7 novembre 2019

Figure 5 – *Compte rendu de réunion 5*

6 Compte rendu de réunion 6

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #6 Projet VISIT *13 novembre 2019*

Participants

Présents : Gilles Venturini, Barthelemy Serres, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Eu égard des dates de rendu des parties spécifications et veille technologique qui approche ? il est nécessaire de faire une réunion pour faire le point sur l'avancé des tâches en cours. Compte tenu de la date d'attribution des fonctionnalités qui m'ont été attribuées (24 octobre 2019), l'objectif serait de prioriser les aspects à explorer d'ici à la date de rendu du cahier des spécifications.

Existant

Un modèle de base de données a été défini et une API Rest a été mis en place pour contacter la base. Des recherches ont été effectuées concernant les fonctionnalités m'ayant été attribuées à savoir :

- Pour le système d'administration des ajouts d'ancres et de balises à une image, des recherches sur les technologies SVG sont en cours. Des difficultés ont été identifiées pour les objets 3D.
- La création d'un exécutable pour l'envoie d'un fichier à l'application mobile contenant les images AR pour chaque site.

Attentes

- Poursuivre les recherches concernant les fonctionnalités précédemment évoquées et analyser les nouvelles problématiques que celles-ci peuvent engendrer.
  - Priorisation du modèle 2D avec possibilité de mise à l'échelle demandé à la personne ayant pris la photo (norme à définir par cet utilisateur). Explorer la piste d'une échelle de valeur permettant de définir la distance entre la balise à l'image pour répondre à la problématique des objets 3D.
- Effectuer des recherches concernant la fonctionnalité se rapportant à la détection de la qualité d'une image AR Core. L'existence d'un exécutable permettant de répondre à ce besoin a été évoqué. ***(Nouveau)***
- Communiquer une version de l'interface d'administration de l'application Web, sous forme de mockups ainsi qu'un rapport des problématiques soulevées au membre de l'équipe dans l'objectif de faire surgir de nouvelles idées pour la conception de l'interface.
- Avancer sur les spécifications et établir une nouvelle version du cahier de spécifications plus adapté aux fonctionnalités confiées.

Dates communiquées

- *Mai 2019* : Application Opérationnel

Secrétaire : Thomas CHARLET

Date d'approbation: 14 novembre2019

Figure 6 – *Compte rendu de réunion 6*

7 Compte rendu de réunion 7

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #7 Projet VISIT *16 janvier 2020*

Participants

Présents : Gilles Venturini, Barthelemy Serres, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Eu égard des difficultés d'accès au code présent sur le serveur universitaire donc uniquement accessible depuis l'école, un accès VPN pourrait permettre d'accélérer l'implémentation de solutions.

Existant

Un modèle de base de données a été défini et l'API Rest a été mis en place dans une première version sur une machine virtuelle de tests. Des recherches ont été effectuées concernant les fonctionnalités m'ayant été attribuées à savoir :

- Pour le système d'administration des ajouts d'ancres et de balises à une image, la technologie SVG a été retenue. Aucune solution n'a pour le moment été trouvée pour le placement d'annotations sur les objets 3D.
- Des recherches pour aboutir à la création d'une commande symfony, exécutable pour l'envoie d'un fichier à l'application mobile contenant les images AR pour chaque site ont été réalisées.

Attentes

- Poursuivre les recherches et l'implémentation de solutions de gestion des annotations du point de vu interface.
  - Pour le moment priorisation du modèle 2D avec possibilité de mise à l'échelle demandé à la personne ayant pris la photo (norme à définir par cet utilisateur).
- Se renseigner auprès du service informatique pour l'accès au VPN, difficile de mettre en place sur une machine personnelle. ***(Nouveau)***
- Produire des premières commandes simple symfony afin d'avancer rapidement sur la tâche de création de package à destination de l'application mobile.

Secrétaire : Thomas CHARLET

Date d'approbation: 16 janvier 2020

Figure 7 – *Compte rendu de réunion 7*

8 Compte rendu de réunion 8

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #8 Projet VISIT *29 janvier 2020*

Participants

Présents : Olivier Roussey, Thomas Charlet

Problématique

Devant anticiper les problèmes à venir pour l'intégration de la solution finale à proposer, une réflexion s'articulant autour des problématiques d'intégration a été menée.

Existant

Un modèle évolué de la base et de l'API ont été mis en place. Ces dernières comportant notamment des précisions concernant certains attributs des entités qui composent la base de données. Une machine virtuelle de test comportant le serveur de base de données, l'API Rest ainsi qu'une première version du site web d'administration a été mis en place. Une première version de l'entité annotation dans la base de données a été réalisée.

Attentes

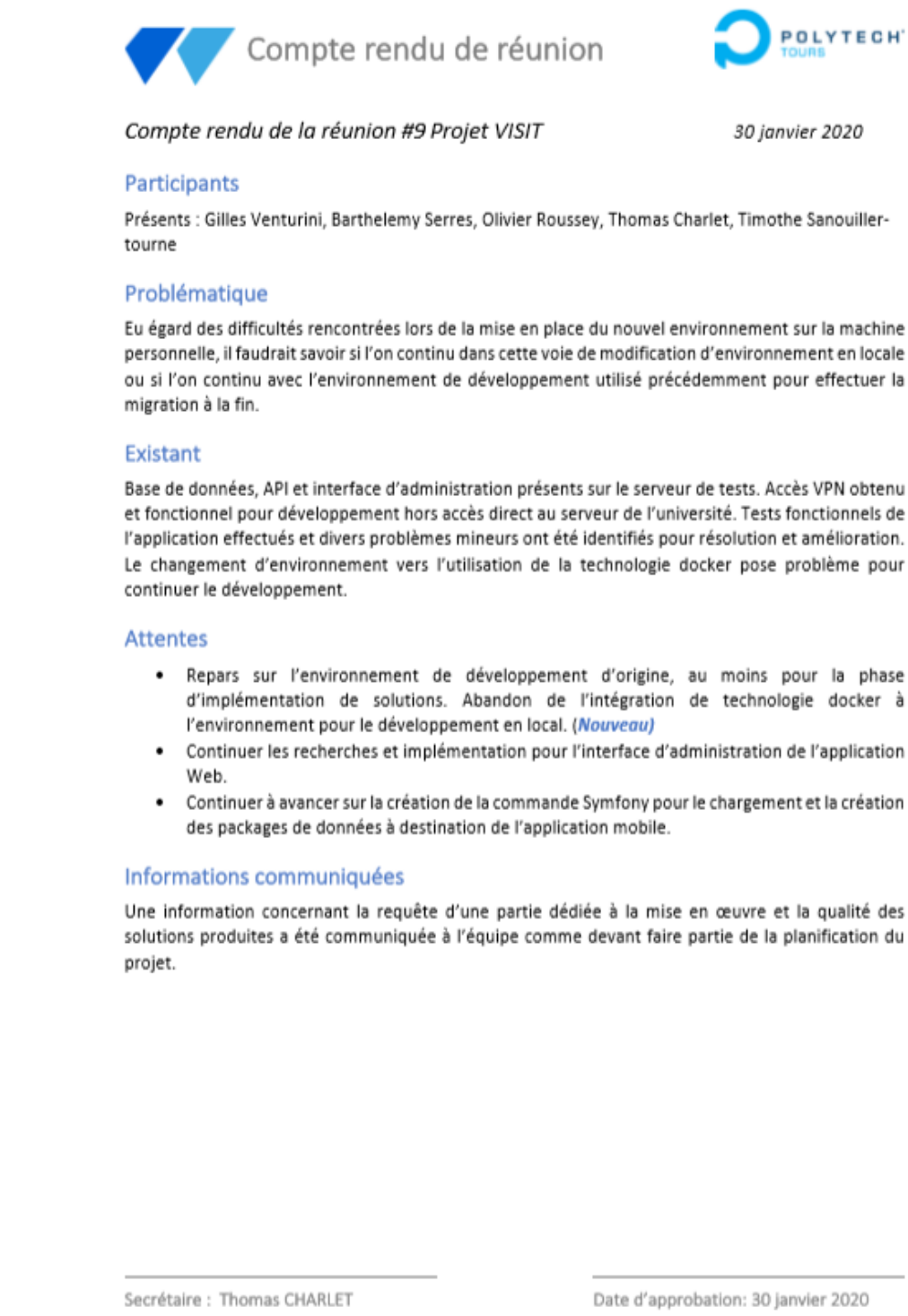
- Concernant l'interface de placement d'annotation, l'utilisation du plugin « leaflet 3if» a été mentionné ainsi que la correspondance de la déclaration de l'entité annotation avec la norme « w3c ». ***(Nouveau)***
- Mise en place d'une réflexion concernant la suppression d'une image RA dans la base de données. Cette suppression occasionnera le rattachement des annotations anciennement reliées à l'image supprimée vers une image par défaut. C'est sur cette dernière image que seront rattachées l'ensemble des annotations qui ne correspondent à aucune image RA. ***(Nouveau)***
- Tentative de changement de l'environnement de développement local vers une version conteneurisée « docker ». ***(Nouveau)***
- Appréhender la technologie de conteneurisation afin de pouvoir s'en servir pour le projet. Quelques commandes ont été communiquées : « docker compose ps », « docker compose stop », « docker compose down », « docker compose up -d. ***(Nouveau)***
- Avancer sur les tâches de création de package de données et de création d'interface une fois la mise en place du nouvel environnement terminée.

Secrétaire : Thomas CHARLET

Date d'approbation: 30 janvier 2020

Figure 8 – *Compte rendu de réunion 8*

9 Compte rendu de réunion 9



Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #9 Projet VISIT *30 janvier 2020*

Participants

Présents : Gilles Venturini, Barthelemy Serres, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Eu égard des difficultés rencontrées lors de la mise en place du nouvel environnement sur la machine personnelle, il faudrait savoir si l'on continu dans cette voie de modification d'environnement en locale ou si l'on continu avec l'environnement de développement utilisé précédemment pour effectuer la migration à la fin.

Existant

Base de données, API et interface d'administration présents sur le serveur de tests. Accès VPN obtenu et fonctionnel pour développement hors accès direct au serveur de l'université. Tests fonctionnels de l'application effectués et divers problèmes mineurs ont été identifiés pour résolution et amélioration. Le changement d'environnement vers l'utilisation de la technologie docker pose problème pour continuer le développement.

Attentes

- Repars sur l'environnement de développement d'origine, au moins pour la phase d'implémentation de solutions. Abandon de l'intégration de technologie docker à l'environnement pour le développement en local. (*Nouveau)*
- Continuer les recherches et implémentation pour l'interface d'administration de l'application Web.
- Continuer à avancer sur la création de la commande Symfony pour le chargement et la création des packages de données à destination de l'application mobile.

Informations communiquées

Une information concernant la requête d'une partie dédiée à la mise en œuvre et la qualité des solutions produites a été communiquée à l'équipe comme devant faire partie de la planification du projet.

Secrétaire : Thomas CHARLET

Date d'approbation: 30 janvier 2020

Figure 9 – *Compte rendu de réunion 9*

10 Compte rendu de réunion 10

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #10 Projet VISIT *5 février 2020*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

La création de l'entité dédiée à enregistrer les informations d'une annotation d'une image dans la base de données ayant été effectuée, et des recherches ayant été effectué pour créer la commande Symfony pour la création des packages de données, des précisions sont nécessaires pour avancer.

Existant

La base de données, l'api et l'interface utilisateur sont à jour sur la machine de tests. Une première version de l'entité annotation a été créée dans la base de données et je suis maintenant en capacité de créer la commande Symfony pour la création du package de données contenant les images RA de chaque site.

Attentes

- Pour l'entité « Annotation » de la base de données, des modifications sont à effectuer :
  - Ajout de coordonnées x et y permettant de localiser l'annotation sur l'image, (*Nouveau)*
  - Modification de la relation entre Annotation et « Annotated_image », inverser le lien existant entre ces deux entités, (*Nouveau)*
  - Ajout de commentaires aux différentes méthodes de classes pour la compréhension et si besoin reprise de code (conformément aux mesures de qualité mis en place), (*Nouveau)*
- Concernant la connexion à une version local de l'application web, régler le problème d'authentification en lien avec le cache Symfony. (*Nouveau)*
- Création de la commande de package basée sur l'algorithme précédemment pensé et réfléchi dans l'une des réunions précédentes.
- Intégrer l'entité « Annotation » ainsi que les services liés à la VM de tests ce qui constitue un premier jalon pour le projet. (*Nouveau)*

Secrétaire : Thomas CHARLET

Date d'approbation: 6 février 2020

Figure 10 – *Compte rendu de réunion 10*

11 Compte rendu de réunion 11

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #11 Projet VISIT *12 février 2020*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet, Timothe Sanouiller-tourne

Problématique

Compte tenu des difficultés rencontrées jusqu'à présent et du délai imposé pour la partie réalisation du projet, il serait primordial de faire une réunion afin de repérorer les tâches à effectuer.

Existant

La base de données est à jour ainsi que l'api et l'interface web utilisateur pour l'administration de sites patrimoniaux.

- L'entité annotation est présente dans la base et l'api permet de les gérer (ajout, modification, suppression), il reste à intégrer ces modifications à l'interface utilisateur,
- L'algorithme de création de la base de données correspond à l'idée fixé par l'équipe de développement quant aux tâches que doit remplir cette commande serveur.

Attentes

Pour donner suite à une re priorisation des objectifs, il a été décidé de laisser la partie intégration interface de gestion des annotations de côté au profit de la tâche de création du package de données à destination de l'application mobile.

- Création de la commande Symfony de packaging des données des images RA :
  - Format de l'archive sera « Tar.gz », (***Nouveau)***
  - Méthode permettant l'exploitation de l'utilitaire « *arcorimg* » pour créer des bases de données d'indexation à partir d'un fichier en entrée créée, (***Nouveau)***
  - Une première version des informations base AR « debug » a été réalisée,
  - Des difficultés sont pour le moment rencontrées pour le contenu de la base AR release ;

Dates communiquées

- *26 mars 2020* : Réunion Management de projet et accompagnement Sopra
- *23 mars 2020* : Soutenance Mise en Œuvre et Qualité
- *1 et 2 avril 2020* : Soutenance final Projet de Recherche

Secrétaire : Thomas CHARLET

Date d'approbation: 13 février 2020

Figure 11 – *Compte rendu de réunion 11*

12 Compte rendu de réunion 12

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #12 Projet VISIT *4 mars 2020*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet

Problématique

Des précisions sont nécessaires pour avancer plus rapidement sur le développement du package de données par site contenant entre autres la base de données des images RA du site correspondant.

Existant

La base de données, l'api Rest et l'interface d'administration des sites patrimoniaux. L'entité annotation a été intégré à la version de tests de l'application web à défaut de pouvoir avancer sur l'interface de gestion de ces annotations pour le moment.

Attentes

- Création de la commande pour créer le package de données par site et selon le mode de lancement de l'application mobile « debug »(état published des œuvres à true) ou « release »(état published des œuvres à false). Ajout de contraintes sur le contenu de l'archive.
  - Qualité image doit être de 75% minimum pour être pris en compte,
  - Différences existantes entre les fichiers « media » en debug et en release pour les Json d'état correspondants.
    - retirer les œuvres non publiées,
    - retirer de la liste des images annotées, celles dont l'image RA a un score insuffisant,
    - retirer la liste des médias de l'onglet Médias car ceux qui sont utilisés sont référencés dans les annotations,
  - Construction des nouveau Json d'état du site patrimonial :
    - Pas besoin de la liste « administrators » d'un site patrimonial,
    - Conservation du champs « validity date » de type string et suppression de « validity date »
    - Enlever les champs « name », « isImage » et « quality » de l'image descriptif d'un site,
    - Enlever les champs « name », « isImage » et « quality » de l'image descriptif d'une zone,
    - Enlever les champs « name », « isImage » et « quality » de l'image descriptif d'une œuvre,
    - Enlever zone ne comportant pas d'annotations,
    - Enlever oeuvre ne comportant pas d'annotations,
    - Dans Œuvre :
      - Retirer champ « media »
      - Enlever Image annotée par default (isDefault = true),
      - Elever champ isDefault pour autres images annotées,
  - Changement de format de l'archive de tar.gz vers zip

Secrétaire : Thomas CHARLET

Date d'approbation: 5 mars 2020

Figure 12 – *Compte rendu de réunion 12*

13 Compte rendu de réunion 13

Compte rendu de réunion

POLYTECH TOURS

Compte rendu de la réunion #13 Projet VISIT *11 mars 2020*

Participants

Présents : Gilles Venturini, Olivier Roussey, Thomas Charlet

Problématique

Un contrôle des tâches effectuées ainsi que de nouvelles précisions sont nécessaires pour valider les solutions produites et avancer plus rapidement sur le développement des méthodes de création des packages de données.

Existant

La base de données, l'api Rest et l'interface d'administration des sites patrimoniaux. Une première version de l'algorithme permettant la création des packages de données a été produite.

Attentes

- Amélioration de la commande pour créer le package de données par site et selon le mode de lancement de l'application mobile « debug « ou « release ».
  - Modification des chemins absolus des différents dossiers contenus dans les archives construites à partir de la commande Symfony personnalisée.
  - Changement du nom des images indexées dans les bases de données AR générées.
  - Ajout de point de validation de la construction des packages à l'aide du logger serveur permettant de mieux diagnostiquer de futurs problèmes et, pour le moment débugger le code plus facilement,
- Commencer la réflexion et la mise ne place d'un plan de test essentiellement sur la partie réalisée par moi-même mais pourrait amener à tester de manière plus globale l'application,
  - Création d'un site complexe et complet (tester le plus de contraintes),
  - Vérification de génération du Json d'état du nouveau site créé,
  - Test de la commande « create-package » de ce site,
  - Vérification des contraintes obtenues sur le package
    - Base AR avec image de qualité >=75
      - Pour la version « Debug » de l'archive,
        - Doit contenir l'ensemble des images RA (dans annotations)
      - Pour la version « Release » de l'archive,
        - Ne contient que les images dans annotations où « creativeWork » a « published » à true,
    - Pour le dossier media,
      - Contient l'image du site,
      - Contient l'image zone si le site est published,
      - Contient l'image creativeWork si un creativeWork est published,
      - Contient les images dans body des annotations si le type est « video » ou « image »,
    - Pour le Json State d'un site, le mettre à jour tel que les Json d'état avait été décrits lors de la précédente réunion.

Secrétaire : Thomas CHARLET

Date d'approbation: 12 mars 2020

Figure 13 – *Compte rendu de réunion 13*

J Glossaire

Angular : Angular est une réécriture complète de Angular JS. C'est donc un Framework JavaScript libre et open source développée par Google permettant de développer des pages Web.

Api-platform : Est un ensemble d'outils et de fonctionnalités permettant de construire et d'exploiter les web APIs.

API REST : L'appellation REST (REpresentational State Transfer) ou RESTFULL est un ensemble de contraintes utilisées pour créer des services web sous la forme d'un style d'architecture logicielle propre. Elle permet entre autres d'établir une interopérabilité entre les ordinateurs sur Internet.

ARCore : Technologie représentant un kit de développement logiciel créé par Google et permettant la conception d'applications de réalité augmentée. (1ère version : février 2018)

Canvas : Cet élément est une composante HTML , spécification depuis HTML5 permettant d'effectuer des rendus dynamiques d'images bitmap via des scripts.

SVG : Format de données ASCII conçu pour décrire des graphiques vectoriels et basé sur XML.

Symfony : Ensemble de composants et Framework MVC libre PHP permettant de faciliter et d'accélérer le développement web à l'aide de fonctionnalités modulables et adaptables mis à disposition des développeurs.

K Webographie

Base de données :

https://www.objectdb.com/

https://www.mysql.com/fr/

API

https://www.lafermeduweb.net/tag/api

https://blog.octo.com/strategie-d-architecture-api/#why

https://blog.octo.com/wp-content/uploads/2017/01/octo-refcard_api_archi_en_v1-0.pdf

Open API api-platform

https://en.wikipedia.org/wiki/OpenAPI_Specification

https://swagger.io/docs/specification/about/

https://api-platform.com/

https://symfony.com/

Interface Web

https://www.usabilis.com/definition-utilisabilite-usabilite/

https://angularjs.org/

https://angular.io/

https://developers.google.com/ar

https://fr.wikipedia.org/wiki/Scalable_Vector_Graphics

https://blog.angularindepth.com/how-to-get-started-with-canvas-animations-in-angular-2f797257e5b4

Comparaison technologies graphiques

https://www.youtube.com/watch?v=30li6w62eCo

Canvas 3D

https://www.alsacreations.com/tuto/lire/1572-webgl-3d-three-canvas-threejs.html

https://threejs.org/

Projet Recherche & Développement

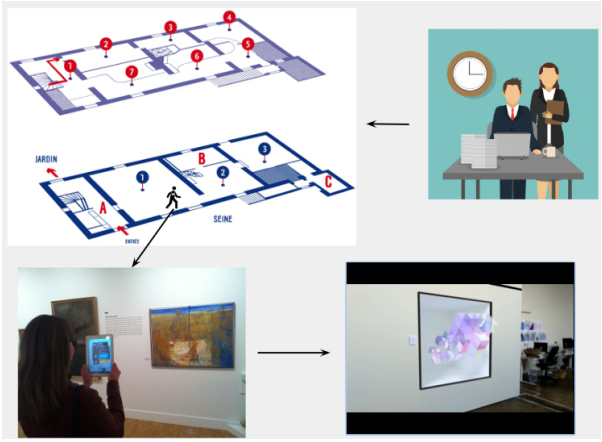
Projet VISIT : Web Application Administration des sites patrimoniaux de la région Centre

Thomas Charlet

Encadrement : Gilles Venturini

Objetcifs

Création d'une application pour le tourisme de la région Centre permettant de réaliser des visites en réalité augmentée. Projet composé d'une application mobile qui reçoit les informations permettant d'effectuer des visites depuis un serveur dont le contenu est administrer par un site web qui fait l'objet du projet de recherche.

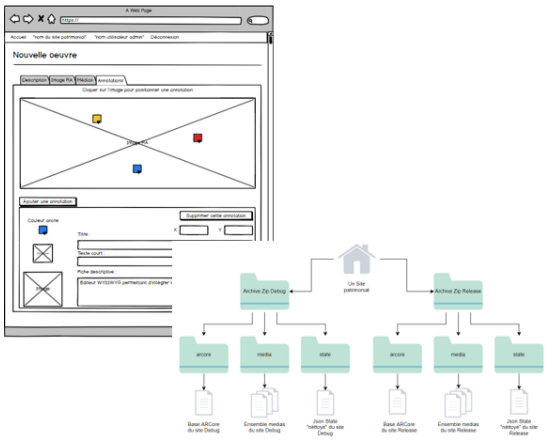


Création de visites touristiques "2.0"

Mise en oeuvre

Principalement 2 tâches :

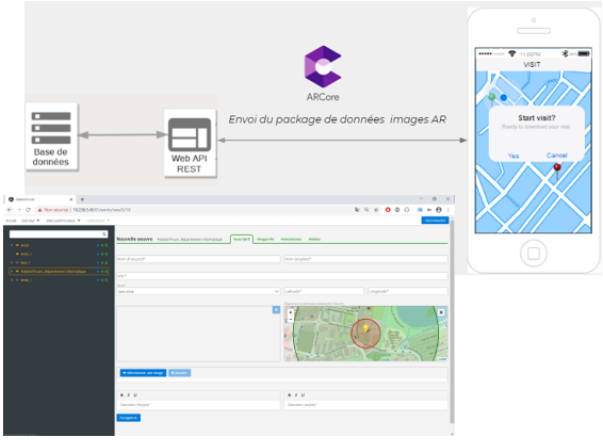
- Création et mise à disposition de packages de données contenant entre autres les bases ARCore pour l'application mobile.
- Création d'entités et services web permettants la gestion d'annotations côté serveur Back-end.



Création des packages de données et Interface de gestion d'annotations

Résultats attendus

- Le serveur est capable d'envoyer à l'app mobile les données pour de réaliser une visite touristique d'un site patrimonial voulu à l'aide d'images RA,
- Un administrateur de site patrimonial peut ajouter des annotations sur les oeuvres d'arts qu'il possède.



Transfert de données de visites entre le serveur et l'app mobile

Ecole Polytechnique de l'Université François Rabelais de Tours
Département Informatique
64 avenue Jean Portalis, 37200 Tours, France
polytech.univ-tours.fr

POLYTECH TOURS
Informatique

Projet Recherche & Développement

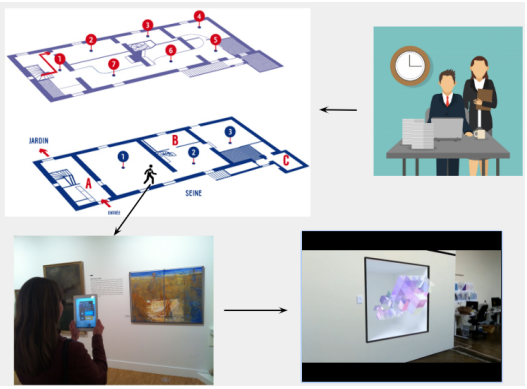
Projet VISIT : Web Application Administration des sites patrimoniaux de la région Centre

Thomas Charlet

Encadrement : Gilles Venturini

Objetcifs

Création d'une application pour le tourisme de la région Centre permettant de réaliser des visites en réalité augmentée. Projet composé d'une application mobile qui reçoit les informations permettant d'effectuer des visites depuis un serveur dont le contenu est administrer par un site web qui fait l'objet du projet de recherche.

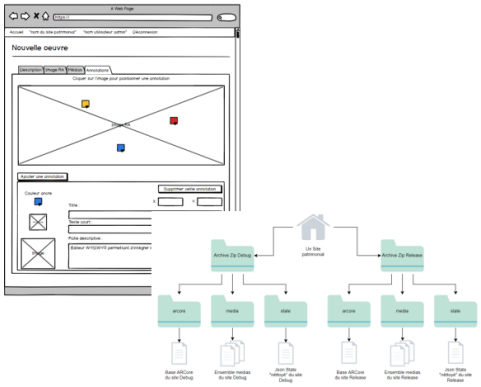


Création de visites touristiques "2.0"

Mise en oeuvre

Principalement 2 tâches :

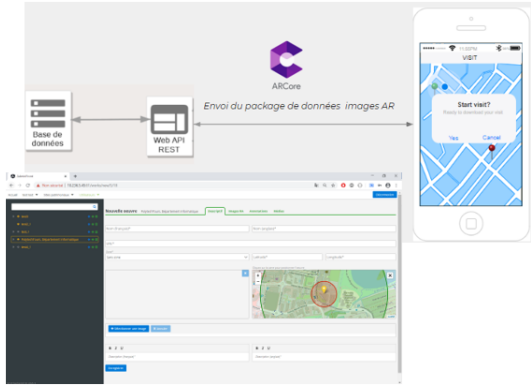
- Création et mise à disposition de packages de données contenant entre autres les bases ARCore pour l'application mobile.
- Création d'entités et services web permettants la gestion d'annotations côté serveur Back-end.



Création des packages de données et Interface de gestion d'annotations

Résultats attendus

- Le serveur est capable d'envoyer à l'app mobile les données pour de réaliser une visite touristique d'un site patrimonial voulu à l'aide d'images RA,
- Un administrateur de site patrimonial peut ajouter des annotations sur les oeuvres d'arts qu'il possède.



Transfert de données de visites entre le serveur et l'app mobile

Ecole Polytechnique de l'Université François Rabelais de Tours
Département Informatique
64 avenue Jean Portalis, 37200 Tours, France
polytech.univ-tours.fr

POLYTECH TOURS
Informatique

Projet VISIT : Web Application Administration des sites patrimoniaux de la région Centre

Résumé

Le projet VISIT consiste à mettre en place une application à destination de personnes souhaitant visiter un site patrimonial de la région Centre. La solution proposée permettra à l'utilisateur de visualiser une œuvre sous la forme d'un modèle 3D à l'aide de la technologie AR Core. L'application globale est constituée de deux modules à savoir l'application mobile et l'application web. Le projet recherche et développement concerne la mise en place de la partie web application permettant l'administration des sites patrimoniaux et de leurs contenus. Ce projet requière des compétences allant de la modélisation et la création de base de données à la conception d'une interface utilisateur ce qui constitue un large éventail des compétences requises pour un ingénieur.
Le rapport fait part de la mise en place du projet et de la phase de recherche visant à répondre aux besoins exprimés par le client. Il est donc constitué de 5 sections relatives au contexte, à la description générale, à l'état de l'art, à la partie analyse et conception et au bilan semestriel du projet. les descriptions sur les interfaces, les spécifications fonctionnelles et non fonctionnelles et la planification du projet ainsi que des documents relatifs aux conditions de tests et compte rendu de réunion sont joint à ce document sous forme d'annexes.

Mots-clés

API : Interface de Programmation Applicatif, Symfony : Framework PHP, IHM : Interface Homme Machine, ARCore : SDK pour réalité augmentée par Google, Angular : Framework JavaScript Google, SVG : Image Vectorielle (mise à l'échelle), SDK : Kit de développement logiciel.

Abstract

The VISIT project consists in setting up an application for people wanting to visit a heritage site in the Centre region. The proposed solution will allow the user to visualize a work in a 3D model using AR Core technology. The global application is divided into two modules: a mobile application and a web application. The research and development project concerns the implementation of the web application part allowing the administration of heritage sites. This project requires skills ranging from modeling and database creation to user interface design, which is a wide range of skills required for an engineer.
The report describes the implementation of the project and the research part according to the needs expressed by the client. Therefore, it consists of 5 sections relating to the context, the general description, the state of the art, the analysis and design part and the half-yearly assessment of the project. The descriptions on interfaces, functional and non-functional specifications and project planning as well as documents relating to test conditions and meeting minutes are attached to this document in annexes.

Keywords

API : Application Programming Interface, Symfony : Framework PHP, IHM : Human-Machine Interface, ARCore : SDK for augmented reality by Google, Angular : Framework JavaScript Google, SVG : Scalable Vector Graphics, SDK : Software Development Kit.

Tuteur académique
Gilles Venturini

Étudiant
Thomas Charlet (DI5)