



ANALYSE TECHNIQUE



BORDEAUX
SCIENCES
AGRO



iSPA
Interactions Sol Plante Atmosphère



AgroTIC



L'INSTITUT
agro Montpellier

AMEIL, BEILLE, BOUSQUET, COLOMBET, DALES, DELBECQ, FAURE,
GERMAIN, JEAUMART, JOLI, MILAZZO, PRESSÉ, RÉGNIER, SPINNER & THOMAS

2023-24

Sommaire

Analyse technique

Glossaire	3
Introduction.....	4
1. Description du projet	5
1.1. Contexte du projet	5
1.2. Les besoins et objectifs	5
1.3. Les fonctionnalités principales	6
1.4. Les différents profils utilisateurs	7
1.5. L'arborescence du site	8
1.6. Contraintes du projet	9
2. Les données	10
2.1. Les données parcellaires	10
2.2. Les données de sortie du modèle	11
2.3. Les données liées à l'utilisateur	12
2.4. Le dictionnaire de données	13
2.5. Le modèle de données	16
3. Environnement technique	17
3.1. <i>Back-end</i> (côté serveur)	17
3.1.1. Système de gestion de base de données	18
3.1.1.1. Besoins de la solution	18
3.1.1.2. Outils existants	19
3.1.1.3. Comparaison des outils (définition des critères, poids et valeurs)	19
3.1.1.4. Choix pour la solution.....	21
3.1.2. Langages et technologies de développement	23
3.1.2.1. Besoin de la solution	23
3.1.2.2. Outils existants	23
3.1.2.3. Comparaison des outils (définition des critères, poids et valeurs)	25
3.1.2.4. Choix pour la solution.....	26
3.1.3. Serveur web.....	26
3.1.3.1. Besoins de la solution	26
3.1.3.2. Outils existants	27
3.1.3.3. Comparaison des outils (définition des critères, poids et valeurs)	27
3.1.3.4. Choix pour la solution.....	28
3.1.4. Serveur cartographique	28
3.2. <i>Front-end</i> (côté client)	29
3.2.1. Langages et technologies de développement	29
3.2.1.1. Besoins de la solution	29
3.2.1.2. Outils existants	29
3.2.1.2.1. Langage de programmation	29
3.2.1.2.2. <i>Framework</i> /bibliothèque JavaScript	31
3.2.1.2.3. <i>Framework</i> /bibliothèque CSS.....	33
3.2.1.3. Comparaison des outils (définition des critères, poids et valeurs)	35
3.2.1.4. Choix pour la solution.....	36
3.2.2. Cartographie interactive	36
3.2.2.1. Besoins de la solution	36
3.2.2.2. Outils existants	36
3.2.2.3. Comparaison des outils (définition des critères, poids et valeurs)	37
3.2.2.4. Choix pour la solution.....	40
3.2.3. Visualisation des données graphiques	40

3.2.3.1.	Besoins de la solution	40
3.2.3.2.	Outils existants	41
3.2.3.3.	Comparaison des outils (définition des critères, poids et valeurs)	42
3.2.3.4.	Choix pour la solution.....	43
4.	Choix d'architecture technique	43
5.	Choix finaux pour le démonstrateur	44
5.1.	Fonctionnalités développées.....	44
5.2.	Les données	44
5.2.1.	Les données parcellaires	44
5.2.2.	Les données de sorties du modèle	45
5.2.3.	Les données liées à l'utilisateur	45
5.3.	SGBD.....	45
5.4.	Environnement de développement <i>back-end</i>	46
5.4.1.	Langage de programmation du <i>back-end</i>	46
5.4.2.	Serveur web.....	46
5.5.	Environnement de développement <i>front-end</i>	46
5.5.1.	Langage et technologies de développement	46
5.5.2.	Cartographie interactive	46
5.5.3.	Visualisation des données graphiques	47
5.6.	Schéma de l'architecture technique du démonstrateur	47
6.	Bibliographie.....	48

Glossaire

Besoin	Nécessité dans l'utilisation de l'application, exprimée par un ou plusieurs utilisateur(s) potentiel(s).
Équipe de travail	Ensemble d'utilisateurs connectés (membres), associé à des parcelles partagées. Cette équipe est composée d'un utilisateur "Branche" administrateur, qui peut être accompagné d'un ou plusieurs utilisateurs "Feuille".
Fonctionnalité	Service qu'offre une application à ses utilisateurs. <i>Ici, les fonctionnalités sont issues des besoins (cf. Besoin).</i>
Persona	Personnage fictif réaliste, illustrant un profil utilisateur et traduisant des motivations d'un comportement d'utilisation.
Profil métier	Catégorie de personnes et caractéristiques associées en fonction de leur métier.
Profil utilisateur	Statut d'un utilisateur dans une application, auquel est associé un accès à un ensemble de fonctionnalités disponibles et des droits d'utilisation. <i>Différents profils utilisateurs ont accès à différentes fonctionnalités (cf. Fonctionnalité).</i>
Scenario	Script d'une navigation fictive à travers l'application par un <i>persona</i> pour accéder à un service.
User story	Courte narration décrivant une fonctionnalité d'une application sous le format : En tant que [utilisateur], je veux [faire une action], afin de [obtenir ce résultat].
Utilisateur	Personne réelle utilisant une application.
Utilisation	Manière d'utiliser l'application par un utilisateur. <i>Ces utilisations sont rendues possibles par les fonctionnalités de l'application (cf. Fonctionnalité).</i>
IdO	Internet des Objets
ITS	Itinéraire technique sylvicole
RU	Réserve utile
SGBD	Système de gestion de base de données

Introduction

Ce document prend place dans le cadre du projet AgroTIC, *e-Sylve*. Ce projet répond à la demande de l'INRAE et donne une suite au projet *Forêts-21*, portail web mettant à disposition les simulations de différents modes de gestion forestière dans un contexte de changement climatique. Le projet *e-Sylve* vise à proposer un nouveau site web multi-plateforme exploitant les modèles et les données de *Forêts-21*, mais proposant une ergonomie et des interfaces plus adaptées à différents profils d'utilisateurs. Le projet est réalisé par les 15 étudiants de la promotion AgroTIC 2023-2024, en dernière année des écoles d'ingénieur en agronomie de Bordeaux Sciences Agro et de l'Institut Agro Montpellier.

Ce document correspond à l'analyse technique et succède à une première phase d'étude des besoins et d'analyse fonctionnelle. Il vise à préciser les réponses techniques aux fonctionnalités identifiées. L'analyse a pour objectifs de préciser l'architecture technique de la plateforme, de définir les critères susceptibles de guider les choix techniques et de soulever des questions nécessaires au bon déroulement de la phase de développement. Aussi, après un rappel des besoins et des fonctionnalités identifiés, l'analyse sera organisée en grandes sections relatives aux données à gérer, à l'environnement technique et à l'architecture à mettre en place. Dans le cadre du projet *e-Sylve* un démonstrateur de la solution est à produire. La dernière partie du document sera consacrée aux choix techniques retenus pour ce démonstrateur. Ils pourront être différents des préconisations données pour la plateforme qui sera mise en place à l'issue du projet.

1. Description du projet

1.1. Contexte du projet

La forêt en France occupe 31 % de la superficie du territoire et la filière bois est un secteur économique qui représente 378 000 emplois (DGPE, 2019). Le dérèglement climatique modifie les contextes de production, ce qui en fait un enjeu majeur pour la filière sylvicole. L'émergence de logiciels et applications prenant en compte le dérèglement climatique est essentielle pour les acteurs de la filière souhaitant s'appuyer sur ces outils pour adapter leurs pratiques. Le site *Forêts-21* s'inscrit dans cette démarche. L'unité mixte de recherche (UMR) ISPA en collaboration avec l'INRAE et Bordeaux Sciences Agro a créé la plateforme *Forêts-21* disponible depuis 2021 avec l'objectif de faciliter la planification de la production sylvicole à l'échelle régionale et nationale (INRAE, 2023). *Forêts-21* simule l'évolution des forêts entre 2006 et 2100 en prenant en compte différents scénarios climatiques et les caractéristiques des parcelles.

Le projet *e-Sylve*, initié par le réseau mixte technologique (RMT) AFORCE, ambitionne d'améliorer l'ergonomie de l'affichage des résultats observables sur *Forêts-21*. Cela s'inscrit dans une volonté de valorisation des données de *Forêts-21*, par la facilitation de l'accès à celles-ci à destination des acteurs de la forêt. Par ailleurs, le site web aura une ambition pédagogique, en mettant à la disposition du public des données sélectionnées pertinentes dans une interface accessible.

Dans le cadre de l'analyse réalisée par les étudiants AgroTIC, le projet *e-Sylve* se concentre sur le contexte de deux domaines, la forêt des Agreux gérée par la coopérative ALLIANCE Forêts Bois et la propriété forestière de la Caisse des Dépôts gérée par la Compagnie des Landes.

1.2. Les besoins et objectifs

L'analyse des besoins précédemment effectuée a permis d'identifier les besoins du projet *e-Sylve*. Elle s'appuie sur des entretiens semi-directifs auprès de potentiels futurs utilisateurs du site, une analyse concurrentielle et une analyse de l'existant.

Les 20 besoins identifiés pendant cette première phase d'analyse ont été regroupés en 3 catégories (Général, Visualisation, Export), récapitulés ci-dessous :

- **Général** : le produit final a besoin d'être multiplateforme et donc adaptable à n'importe quelle taille de terminal, et posséder une page informative. Il devra être accessible à tout public, avec des fonctionnalités en plus pour les personnes possédant un compte sur l'application. Pour les utilisateurs connectés, il y a possibilité de rejoindre ou quitter une équipe de travail. Les utilisateurs ont besoin de pouvoir choisir leurs paramètres d'entrée ou leurs variables de sortie, et de pouvoir les placer en tant que favoris. Les requêtes doivent pouvoir être gardées en mémoire ;

- **Visualisation** : le produit final nécessite d'afficher des données sous différentes formes, à savoir graphique et cartographique. Les utilisateurs ont besoin de pouvoir comparer les différents graphiques entre eux. Il doit aussi être possible, selon les droits d'accès, d'ajouter des parcelles, de les visualiser et de les commenter. En ce qui concerne les cartes, il y a besoin de pouvoir (dé)zoomer, s'y déplacer, choisir une zone d'étude ou utiliser une fonction de recherche pour localiser la zone d'étude. Différents fonds de cartes sont également souhaités ;
- **Export** : les cartes, les graphiques, les tables et les rapports générés ont besoin de pouvoir être exportés sous différents formats.

Les différents besoins ont ensuite été regroupés par profils métiers et hiérarchisés par ordre d'importance *via* la prise en compte de la fréquence de mention lors des entretiens et l'importance que les interviewés ont exprimée vis-à-vis du besoin. Une restitution orale a permis d'ajuster ces différents besoins.

1.3. Les fonctionnalités principales

L'analyse fonctionnelle a permis de traduire les besoins recensés ci-dessus, en les exprimant sous forme de *user stories*. Ces dernières sont de courtes narrations décrivant une des fonctionnalités principales sous un format spécifique : "En tant que [type d'utilisateur], je veux pouvoir [fonctionnalité souhaitée], afin de [objectif à atteindre]."

Les fonctionnalités ont également été classées en trois catégories issues des catégories de besoins : Général, Visualisation, et Export. En voici un bref récapitulatif dans le Tableau 1 ci-dessous :

Tableau 1 : Récapitulatif des grandes fonctionnalités énoncées dans l'analyse fonctionnelle

Catégorie	Fonctionnalités
Général	Adaptation de la fenêtre en fonction de la taille du terminal.
	Création de compte , avec des profils adaptés aux besoins. Il est possible d'accéder à des données aussi bien sans connexion à un compte , que connecté . La récupération d'un mot de passe oublié est possible.
	Page d'accueil informative avec un accès au site de <i>Forêts-21</i> .
	Paramètres d'entrées pré-sélectionnés en fonction de la zone étudiée et variables de sortie pouvant être ajoutés et retirés des favoris . La dernière requête réalisée est gardée en mémoire. Il est possible de choisir les variables de sortie proposées dans la

	visualisation des données. Il y a une interface différenciée selon les profils utilisateurs. Des conseils d'utilisation sont présents sur les différentes pages et pour préciser les différents champs et variables.
	Des équipes de travail sont disponibles pour les utilisateurs connectés. De ce fait, ce groupe peut s'éditer en ajoutant ou en retirant des personnes. Il est également possible de demander à en faire partie ou à le quitter. Un gestionnaire pourra contrôler les fonctionnalités disponibles aux membres de son groupe de travail.
Visualisation	Différentes données pourront être visuellement observées , selon les paramètres d'entrée et les variables choisis, soit sur un ou des graphique(s) , soit sur une carte . En sélectionnant une parcelle, les paramètres d'entrée peuvent être pré-remplis pour éviter de les remplir à chaque utilisation. Il sera possible de comparer plusieurs graphiques, en fonction des variables de sorties sélectionnées.
	Les parcelles sylvicoles peuvent être ajoutées au site. Il sera alors possible de les visualiser et d'y éditer des commentaires .
	Une carte interactive affichera des informations que l'on souhaite observer spatialement. Le zoom et le déplacement sur la carte seront possibles. Pour être plus efficace, la définition d'une zone d'étude et la recherche seront deux outils à disposition. Le fond de carte pourra être modifié.
Export	Les tables , cartes et graphiques pourront être exportés. Des rapports pourront être générés.




Chaque fonctionnalité a été priorisée selon le besoin métier (voir analyse des besoins) et la priorité de développement (voir analyse fonctionnelle). Les notes attribuées ont ensuite été ajustées par les personnes ressources afin que cela soit au plus proche de leurs besoins.

1.4. Les différents profils utilisateurs

Des profils utilisateurs ont également été définis dans l'analyse fonctionnelle. Ces profils sont au nombre de 3. Des profils connectés avec différentes fonctionnalités disponibles (appelés "Branche" et "Feuille") et un profil non-connecté ("Bourgeon") ayant un accès restreint à l'application.

Il y a un droit d'accès spécifique à chaque fonctionnalité en fonction du profil utilisateur. Le Tableau 2 établit les différences entre les 3 profils utilisateurs.

Tableau 2 : Aperçu des fonctionnalités disponibles pour les 3 profils utilisateurs.

	Branche 	Feuille 	Bourgeon 
Compte	Oui	Oui	Non

ANALYSE TECHNIQUE

Changement de profil utilisateur	Oui	Oui	Non
Visualisation des données : cartographiques et graphiques	Oui <i>Toutes les variables (accès aux Favoris et Choix des variables visibles)</i>	Oui <i>Toutes les variables (accès aux Favoris et Choix des variables visibles)</i>	Oui <i>Sélection imposée de variables</i>
Administrer une équipe	Oui	Non	Non
Être membre d'une équipe	Non	Oui	Non
Parcelles : Importation, Suppression, Commentaires	Oui	<i>Selon choix de l'administrateur Branche</i>	Non
Exportation	Oui	Oui	Oui

En outre, la création d'un compte "Branche" nécessitera d'être acceptée par un 'super-administrateur'. De même, le changement de profil utilisateur, de "Feuille" à "Branche", doit être validé par une personne qui confirmera que le Siret de l'entreprise est correct et que la Branche qui demande à être créée et bien rattaché à ce Siret. Cette personne sera appelée le "super administrateur". Si le compte "Branche" souhaite revenir vers un profil utilisateur "Feuille" ou "Bourgeon", un message d'alerte informe l'utilisateur qu'il entraîne la suppression de son équipe de travail.

Il faut donc que cette analyse technique propose des outils informatiques qui répondent au mieux à chacune de ces fonctions par utilisateur en détail.

1.5. L'arborescence du site

Finalement, le site, que les étudiants ont nommé *Tree 2 1*, sera constitué d'une page Accueil. Un lien vers le site *Forêts-21* est accessible.

- Un onglet Connexion est disponible permettant d'avoir accès à une page Equipe où il est possible de choisir les variables qui seront visibles, modifier ses informations personnelles associées au compte, modifier son profil utilisateur, se déconnecter ;
- Un onglet Données permet d'accéder à deux interfaces de visualisation (cartographiques et graphiques), accessibles pour les 3 profils utilisateurs.
- Un onglet Parcelle ("Branche", "Feuille" selon les droits accordés par la Branche de son équipe de travail) permet d'importer, supprimer et commenter des parcelles ;

ANALYSE TECHNIQUE

- Enfin, l'onglet Équipe permettra d'administrer (dans le cas du compte "Branche") et de consulter l'équipe de travail ("Branche" et "Feuille"), la quitter et en rejoindre une autre ("Feuille"). La Figure 1, ci-dessous, illustre l'arborescence générale de l'application web.

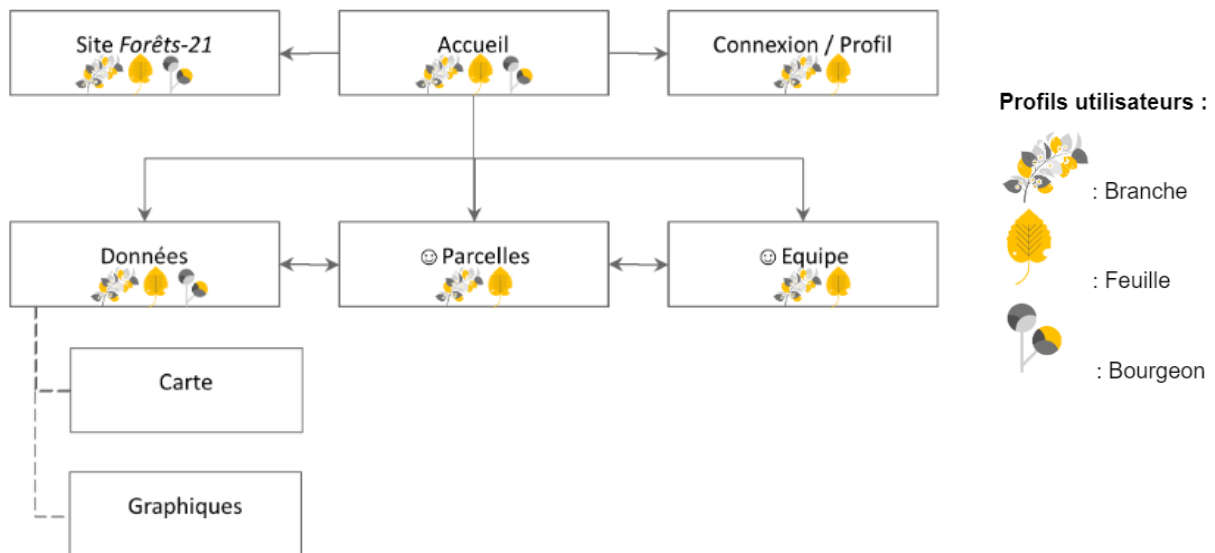


Figure 1 : Arborescence générale du site Tree 2 1

1.6. Contraintes du projet

Le produit final doit prendre en compte les contraintes imposées par les commanditaires. En effet, il faut considérer les problématiques suivantes:

- Qui va reprendre le projet ? ;
- Quel langage de programmation sera utilisé ? ;
- Sur quel serveur sera hébergé le site final ? ;
- Comment seront hébergées/stockées les données ? ;
- Comment intégrer la base de données de *Forêts-21* à *Tree 2 1* ? ;
- Quel est le niveau d'ergonomie à choisir pour le produit final ?

Pour répondre à ces contraintes, il sera pertinent de réaliser des veilles qui permettent de lister les solutions et les langages existants en insistant sur les avantages et inconvénients de ces solutions dans le but d'éclaircir les choix à faire.

Après plusieurs entretiens avec les commanditaires, il existe à ce jour de nombreuses incertitudes quant au développement de la solution (et l'organisme qui l'assurera), suite au projet des étudiants. Aucune contrainte liée aux compétences et habitudes de développement de l'équipe qui

prendra en charge le développement n'est donc identifiée à ce jour. Les préconisations présentées dans ce document se basent donc uniquement sur le résultat des veilles et mettent en avant les solutions qui présentent le plus d'avantages et qui sont disponibles en *open source*. Néanmoins, le modèle économique n'est pas totalement défini, mais prévoit *a priori* la gratuité pour les utilisateurs.

Après une description générale du projet et de la future application web, il est primordial de présenter les données qui seront échangées et manipulées par les différents composants de l'application.

2. Les données

2.1. Les données parcellaires

Les données parcellaires correspondent au parcellaire géoréférencé et ses données associées, que les utilisateurs importeront dans le site web *Tree 2 1*. Les fichiers des parcellaires qui seront importés, après validation par un processus d'importation, par les utilisateurs sont au format Shapefile (shp), GeoPackage (gpkg) ou GeoJSON (geojson). Néanmoins, le site web *Tree 2 1* sera potentiellement utilisé par un certain nombre d'utilisateurs (estimé à pas plus de 300) provenant d'entreprises ou structures différentes. Ainsi les champs des colonnes des fichiers ne seront pas les mêmes pour le parcellaire de tous les utilisateurs, et n'auront pas nécessairement les mêmes systèmes de coordonnées. Or certaines caractéristiques de parcelles sont nécessaires en paramètre d'entrée du modèle utilisé. La prise en compte de l'hétérogénéité des données parcellaires serait donc à envisager dans le développement du site *Tree 2 1*, pour pouvoir extraire correctement les bonnes données pour le modèle, à savoir : la localisation, la RU, l'essence, l'ITS, et l'année de plantation.

Les données parcellaires de la Compagnie des Landes ont permis d'avoir une idée générale des informations se trouvant dans un fichier de données parcellaires.

Mis à part la maille SAFRAN qui est obtenue grâce à l'emprise des parcelles, et l'essence d'arbre qui est le pin maritime, les autres caractéristiques du modèle n'apparaissent pas dans leur fichier. En se basant sur cet exemple, il est donc possible que les données parcellaires que les utilisateurs importeront ne contiennent pas les données des paramètres d'entrée. Ces informations auraient permis de renseigner les variables d'entrée du modèle automatiquement pour une parcelle donnée dans les onglets visualisations des données. Sans cela, ce seront d'autres valeurs par défaut qui seront sélectionnées pour les parcelles, et modifiables par l'utilisateur.

Pour conclure, les données parcellaires à importer seront donc au format Shapefile, GeoPackage ou GeoJSON, et seront hétérogènes dans les informations qu'elles contiendront en fonction des structures et entreprises. Elles peuvent donc ne pas contenir les caractéristiques d'entrée du modèle : la RU, l'essence, l'ITS, et l'année de plantation. Seule la maille SAFRAN semble pouvoir

être trouvée car les parcelles sont géoréférencées (shp, gpkg, geojson). Ces paramètres ne seront donc pas automatiquement renseignés dans le modèle pour la visualisation des données.

2.2. Les données de sortie du modèle

Le modèle de *Forêts-21* fournit en sortie des fichiers csv. Les fichiers correspondent à une association variable-simulation, sachant qu'une simulation correspond à un scénario climatique, une essence, un ITS, une RU et un âge de peuplement donné. Il y a donc environ 176 400 fichiers csv à traiter.

Chaque fichier csv (schématisé en Tableau 3) permet d'illustrer pour le couple variable et simulation voulue, l'ensemble des années (sur la période 2006-2100) en colonnes et l'ensemble des mailles SAFRAN (8 km x 8 km à l'échelle de la France entière) en lignes. Pour chaque maille, les coordonnées du point SAFRAN qui lui sont associés sont aussi données en colonnes pour 4 systèmes de coordonnées de référence (SCR).

Tableau 3 : Schéma d'un fichier csv de Forêts-21 pour un couple variable-simulation

	Coordonnée X SCR 1	Coordonnée Y SCR 1	...	2006	2007	...	2100
Maille 1	<i>Coordonnée X</i>	<i>Coordonnée Y</i>	...	<i>valeur de variable</i>	<i>valeur de variable</i>	...	<i>valeur de variable</i>
Maille 2	<i>Coordonnée X</i>	<i>Coordonnée Y</i>	...	<i>valeur de variable</i>	<i>valeur de variable</i>	...	<i>valeur de variable</i>
...	<i>Coordonnée X</i>	<i>Coordonnée Y</i>	...	<i>valeur de variable</i>	<i>valeur de variable</i>	...	<i>valeur de variable</i>
Maille N	<i>Coordonnée X</i>	<i>Coordonnée Y</i>	...	<i>valeur de variable</i>	<i>valeur de variable</i>	...	<i>valeur de variable</i>

De plus, le même système de nomenclature s'applique à l'ensemble de ces fichiers. Il est généré par le modèle, ce qui permet de réduire le nombre d'erreurs.

Le nom du fichier se compose de trois parties. Par exemple : *vecteur__400000_CO2.csv*.

Seules les deux dernières parties sont intéressantes pour le projet, à savoir le code à 6 chiffres et le nom de la variable représentée. Ce code permet de savoir à partir de quelle simulation le fichier a été généré.

- Le premier chiffre correspond au scénario climatique : **3** pour le RCP 2.6, **4** pour le RCP 4.5 et **8** pour le RCP 8.5 ;
- Le deuxième chiffre représente l'essence d'arbre : **0** pour le pin maritime, **1** pour le hêtre, **2** pour le Douglas et **3** pour le chêne ;
- Le troisième chiffre équivaut à l'ITS : de **0 à 6** ;
- Le quatrième chiffre symbolise la maladie : pour le moment il est toujours égal à **0** car cette option n'a pas encore été modélisée ;
- Le cinquième chiffre désigne la RU : **0** pour 25 mm, **1** pour 75 mm et **2** pour 125 mm ;
- Le dernier chiffre traduit l'âge du peuplement : chaque chiffre correspond à une classe d'âge, qui varie suivant l'essence d'arbre.

2.3. Les données liées à l'utilisateur

Comme il a été précisé précédemment, les utilisateurs de la plateforme relèvent de 3 types de profil : "Branche", "Feuille" et "Bourgeon". Le profil "Bourgeon" est un profil par défaut qui ne nécessite pas de création de compte. Aucune information à leur sujet n'est conservée. Les profils "Branche" et "Feuille" sont, quant à eux, accessibles *via* la création d'un compte. Les données relatives à l'utilisateur connecté sont les suivantes :

- Son nom ;
- Son prénom ;
- Son adresse mail ;
- Sa fonction au sein de sa structure ;
- Son profil utilisateur "Feuille" ou "Branche" ;
- Son mot de passe.

De plus, les données des utilisateurs "Branche" et "Feuille" sont étroitement liées à la dynamique des équipes de travail au sein d'une structure sylvicole. Chaque structure (par exemple une compagnie forestière) se compose d'une à plusieurs "équipes de travail". À la création d'une "équipe de travail", il est demandé de renseigner le SIRET de la structure sylvicole, dans le but de connaître les groupes appartenant à une même structure. La supervision des équipes de travail est assurée par un utilisateur de type "Branche". Les membres d'une équipe de travail sont des utilisateurs de type "Feuille". Chaque équipe possède ses propres données parcellaires. Les utilisateurs "Branche" ont la capacité d'accorder des droits de gestion des parcelles aux utilisateurs "Feuilles" de leur équipe. Ces droits sont :

- La visualisation des parcelles, qui est nécessairement octroyée aux membres de l'équipe ;
- La possibilité de commenter les parcelles ;
- L'import et la suppression de parcelles.

Pour les utilisateurs de type "Feuille" ou "Branche", les informations sur les variables qu'ils ont choisies en favoris et celles qu'ils ont choisies de rendre accessibles dans la visualisation des données sont stockées.

2.4. Le dictionnaire de données

Les choix et contraintes mentionnés précédemment conduisent à l'élaboration du dictionnaire de données (Tableau 4) :

Tableau 4 : Dictionnaire de données des tables permanentes

Table	Colonne	Type	Commentaire
categories	idcategories	int	
	nom_categorie	varchar	catégorie de la variable
comptes	idcomptes	int	
	nom_compte	varchar	
	prenom_compte	varchar	
	mail_compte	varchar	
	mdp_compte	varchar	
	action_parcelle	int	possibilité, pour les feuilles, de visualiser les parcelles (niveau 1), de les commenter (niveau 2), ou de les éditer (niveau 3)
	valide	bool	défini si le compte "branche" a été validé par un super administrateur
essences_arbre	idessences_arbre	int	
	nom_essences_arbre	varchar	
fonctions	idfonctions	int	
	nom_fonction	varchar	fonction de l'utilisateur au sein de la structure
equipes_travail	idequipes_travail	int	
its	idits	int	
	nom_its	varchar	
mailles_projet_tree21	gid	int	identifiant
	cell	double	numéro SAFRAN de la maille
	geom	geometry	géométrie de la maille
parcelles	idparcelles	varchar	

ANALYSE TECHNIQUE

	id_parcelle_structure	varchar	
	nom_parcelle	varchar	
	date_plantation	int	
	commentaire	text	
	geom_parcelle	geometry	géométrie de la parcelle
profils	idprofils	int	
	nom_profil	varchar	profil "feuille" ou "branche"
ru	idru	int	
	nom_ru	int	réserve utile
scenarios_climatiques	idscenarios_climatiques	int	
	nom_scenarios_climatiques	varchar	
structures	idstructures	int	
	siret	varchar	
variables	idvariables	int	
	nom_variables	varchar	
	unite	varchar	unité de la variable
	info_bulle	text	explication de la variable en info-bulle
	calcule	bool	indique si la variable est calculée à partir d'autres variables
	formule	text	formule de calcul de la variable si elle est calculée
	cliquable_bourgeon	bool	indique si la variable est ou non cliquable pour les bourgeons
	nom_fichier	varchar	
variables_selectionnees	idvariables_selectionnees	int	
	visible	bool	indique si la variable a été choisi est présente dans la liste déroulante
	favoris	bool	indique si la variable a été enregistrée en favoris par l'utilisateur

ANALYSE TECHNIQUE

Aussi, une table temporaire, associée à un utilisateur, va être créée lors de l’affichage d’une carte ou d’un graphique par l’utilisateur. Cette table est remplie avec les informations présentes dans le fichier csv correspondant à la ou les variables sélectionnées par l’utilisateur. Son remplissage est permis grâce à l’utilisation d’un JOB Talend, entre les fichiers csv et la base de données, qui permet de récupérer les données d’un fichier csv précis (grâce à sa nomenclature) et de bien les mettre en forme pour l’insertion dans la base de données.

Cette table temporaire est nécessaire car 176 400 fichiers csv sont présents. Chacun de ces fichiers contient 790 900 données soit près de 139 milliards de valeurs à traiter. Au vu de cette volumétrie, il ne paraît pas cohérent de charger l’ensemble de ces données dans la base de données. Il a donc été décidé d’importer uniquement les données nécessaires au moment où elles sont nécessaires. Le dictionnaire de cette table se trouve au Tableau 5.

Tableau 5 : Dictionnaire de données de la table temporaire

Table temporaire	Colonne	Type	Commentaire
temp_annees	idtemp_annees	int	
	annee_valeur	real	
	nom_fichier	varchar	

2.5. Le modèle de données

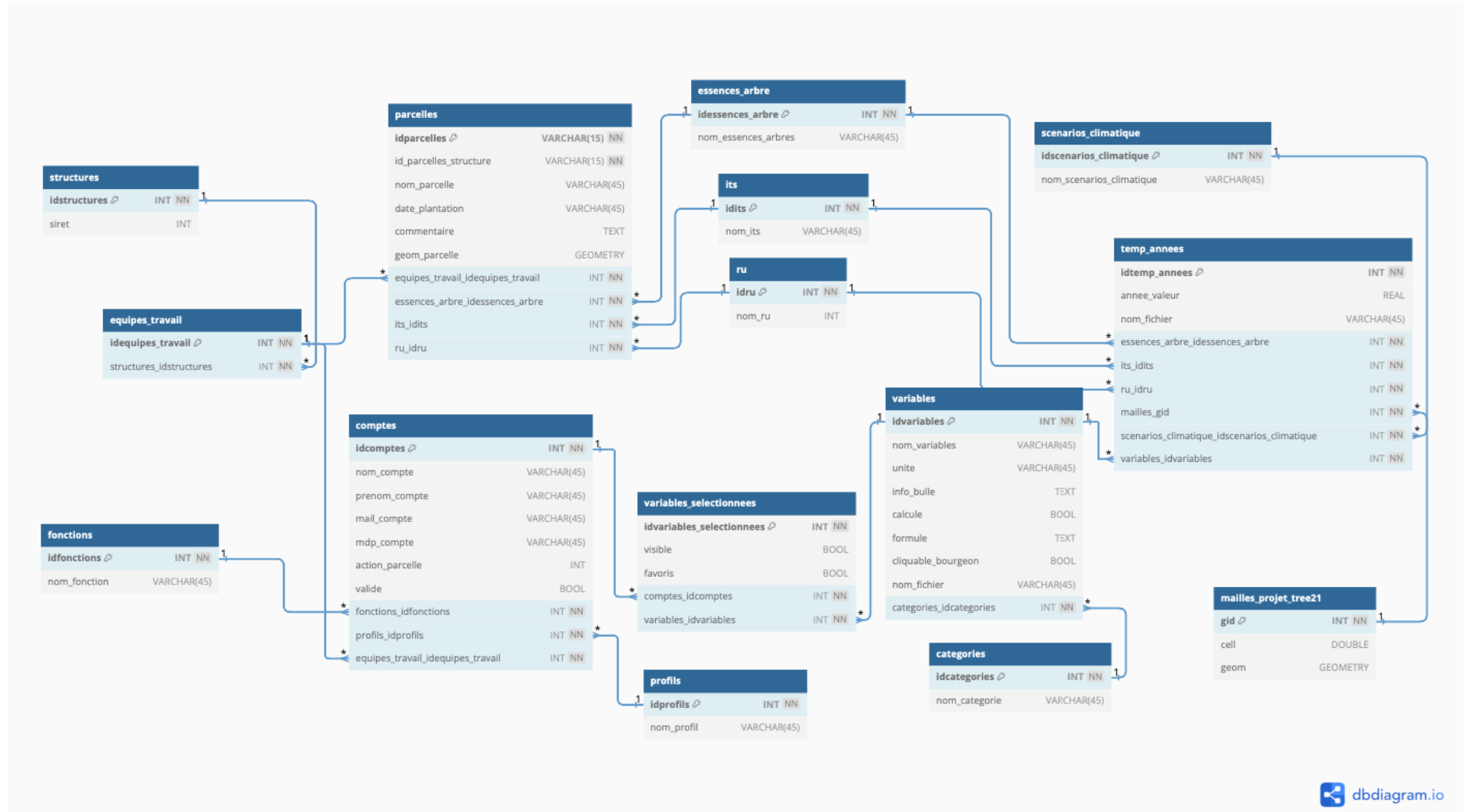


Figure 2 : Modèle de la base de données

3. Environnement technique

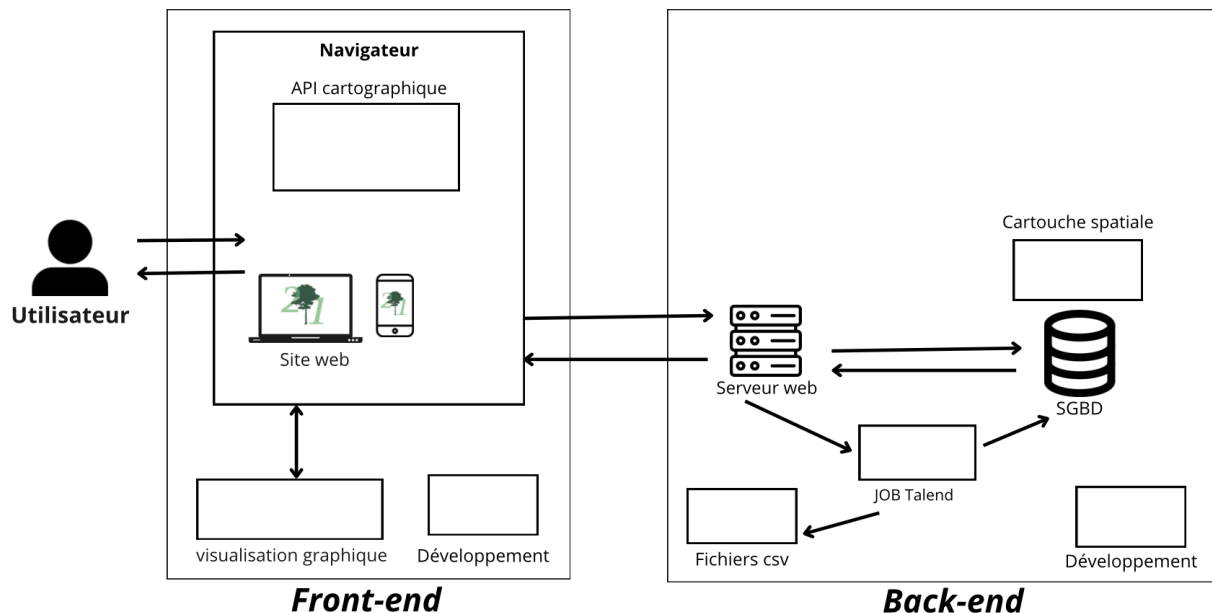


Figure 3 : Schéma de l'architecture technique

L'environnement technique regroupe tous les éléments techniques nécessaires à la réalisation du produit final. Il s'agit des différents langages de programmation et logiciels utilisés pour créer et gérer les différents éléments de l'architecture technique (Figure 3). Afin de les choisir, des veilles technologiques ont été réalisées sur le système de gestion de base de données et sur les éléments techniques côté serveur (*back-end*) et côté utilisateur/client (*front-end*).

3.1. *Back-end* (côté serveur)

Le *back-end* a plusieurs rôles, cités ci-dessous :

- **Gestion de la base de données** : responsable du stockage et de la récupération des données à partir de la base de données. Inclut la création, modification et suppression des données ;
- **Gestion des utilisateurs et de l'authentification** : garantit que seules les personnes autorisées peuvent accéder à certaines fonctionnalités ;
- **Optimisation des performances** : peut être chargé d'optimiser les performances du site en gérant efficacement les requêtes ;
- **Intégration de services tiers** : API externes, données cartographiques, base de données.

En résumé, le *back-end* correspond au bon fonctionnement du site sans que cela soit visible par l'utilisateur.

3.1.1. Système de gestion de base de données

Le SGBD est un logiciel qui gère les aspects primaires d'une base de données. Le SGBD est ainsi ce qui permet la manipulation des données, mais aussi la gestion des utilisateurs et de leurs droits sur la base de données. Le choix d'un SGBD adapté est essentiel pour la pérennité du produit final.

3.1.1.1. Besoins de la solution

Le choix du SGBD pour le projet *e-Sylve* doit répondre aux contraintes de l'application, tout en anticipant la reprise future du projet par une équipe.

Les critères sur le choix d'un gestionnaire vont dépendre des caractéristiques de l'application. Les aspects de volumétrie de la base de données (quantité de données) et de connexions simultanées peuvent demander l'utilisation d'un SGBD optimisé pour ces contraintes.

Il est possible d'estimer la volumétrie de la base de données par le volume des fichiers csv qui stockent les sorties du modèle de *Forêts-21*, qui représente la majorité des données stockées dans la base de données. Les sorties du modèle se font sur les 8 602 mailles SAFRAN de la France métropolitaine pour les 94 années que couvre le modèle (une valeur par année).

Le SGBD ne stockera pas l'ensemble des fichiers csv, mais uniquement quelques fichiers csv, 36 au maximum, sous la forme de table temporaire. Sachant qu'un fichier csv contient 8 602 lignes environ, et que chaque csv contient 94 années. Si les fichiers sont mis en forme de telle sorte qu'il y ait une ligne par maille et par année (et non plus une ligne par maille), la table contiendrait $36 \times 8\,602 \times 94 = 29\,109\,168$ lignes au maximum. En considérant que toutes les valeurs sont de type *float* et ainsi stockées sur 4 octets, la taille finale obtenue est de 116 Méga octets. Par conséquent, le volume de la base est estimé à moins de 1 Giga octet. Ce volume ne contraint pas le choix du SGBD.

En tant qu'application web, *e-Sylve* demandera la connexion simultanée des utilisateurs. Dans le cadre de l'édition des données, cela peut être coûteux en performance pour le SGBD. Les opérations d'édition seront circonscrites à l'importation des parcelles et leur suppression. Les entretiens avec les utilisateurs finaux ont permis de préciser l'usage qu'il y aura de l'application. Il en ressort que l'application ne sera pas utilisée quotidiennement et que les importations de parcelles associées au compte "Branche" ne seront pas systématiques à chaque utilisation. Aussi, le nombre maximum de comptes "Branche" est estimé à quelques centaines d'après les commanditaires. Il s'agit donc d'une petite application avec peu de contraintes sur le SGBD.

L'application permettra la manipulation d'objets géographiques qui seront stockés sur la base de données. Par conséquent, le SGBD choisi devra avoir un moteur géographique.

Enfin, le choix du SGBD se doit d'être cohérent avec l'environnement final de l'application (système d'information et compétences pour l'administration du SGBD de la structure hébergeuse). Ces critères devront être considérés une fois la nature du repreneur éclaircie.

3.1.1.2. Outils existants

Plusieurs SGBD existent avec un fonctionnement et des spécifications variées. Afin de sélectionner les plus pertinents à étudier dans cette veille, l'étude réalisée par le site db-engines (*DB-Engines Ranking - Method*, s. d.), sur la popularité des SGBD en janvier 2024, a été utilisée. Cette étude prend en compte, entre autres, le nombre de mentions du système sur les sites internet, l'intérêt général pour le système (basé sur la fréquence de recherche sur Google), la fréquence des discussions techniques sur le système sur le web (et donc la documentation disponible), le nombre d'offres d'emploi et le nombre de profils LinkedIn dans lesquelles le système est mentionné. Seuls les quatre SGBD les plus populaires selon le site db-engines sont considérés :

- Oracle ;
- MySQL ;
- Microsoft SQL server ;
- PostgreSQL.

3.1.1.3. Comparaison des outils (définition des critères, poids et valeurs)

Les quatre outils sont comparés selon les critères d'évaluation présentés dans le Tableau 6 . Ce sont les critères les plus souvent examinés par les sites de comparaison des SGBD. Néanmoins, les deux critères que sont la volumétrie/performance et la cohérence du système d'information (cohérence avec le système de gestion de base de données utilisé par la structure qui reprendra le projet), ne seront pas pris en compte dans les tableaux suivants. En effet, la taille de la base de données (estimée à 1 Go) n'est pas limitante dans notre cas, et aucune information n'a été fournie sur la structure qui reprendra le projet, ce qui n'en fait donc pas un critère.

Tableau 6 : Critères d'évaluation des principaux SGBD

Critères d'évaluation	Description
Moteur géographique. ²	Il s'agit d'une extension ou non, proposée par le SGBD, permettant la manipulation d'informations géographiques (spatiales) sous forme de géométries (points, lignes, polygones).
Sécurité de la Base. ⁴	Il s'agit des fonctionnalités de sécurité du SGBD et de la documentation disponible pour paramétrer la base dans l'objectif de sécuriser la base de données.

ANALYSE TECHNIQUE

Compatibilité système d'exploitation. ²	Compatibilité avec les principaux systèmes d'exploitation qui sont : Windows, Linux & macOS X.
Fiabilité. ⁵	Capacité d'un logiciel de rendre des résultats corrects quelles que soient les conditions d'exploitation (notamment lors de la panne de composants).
Capacité d'administration. ^{5,6}	Cela correspond à la facilité à reprendre le projet et manipuler la base de données, ce qui dépend entre autres du nombre d'administrateurs disponibles pour reprendre le projet.
Évolutivité. ³	Disponibilité en fonctionnalités et en extensions pour adapter le SGBD à l'évolution de la base de données

L'analyse des différents outils à l'aide des critères précédemment détaillés est effectuée dans les Tableaux 7 à 10.

Tableau 7 : Analyse du SGBD Oracle

Avantages	Limitations
<p><u>Fiabilité</u> : ce SGBD est extrêmement robuste et fiable.</p> <p><u>Évolutivité</u> : les nombreuses fonctionnalités proposées permettent de faire évoluer facilement les bases de données.</p>	<p><u>Payant</u> : les licences et maintenances sont assez chères⁷.</p> <p><u>Capacité d'administration</u> : ce SGBD nécessite un haut niveau d'administration en lien avec la difficulté à le prendre en main.</p> <p><u>Prise en main</u> : cet outil est difficile à prendre en main notamment à cause des nombreuses fonctionnalités avancées qu'il propose.</p>

Tableau 8 : Analyse du SGBD MySQL

Avantages	Limitations
<p><u>Open source</u> : cette solution est gratuite.</p> <p><u>Compatibilité OS</u> : MySQL peut aussi bien s'utiliser sous Windows que sous macOS X ou Linux.</p> <p><u>Prise en main simple</u> : la syntaxe simple de MySQL en fait un outil assez facile à utiliser.</p> <p><u>Capacité d'administration</u> : cette solution ne nécessite pas beaucoup d'administrateurs pour fonctionner, ce qui rend l'utilisation simple.</p>	<p><u>Efficacité du moteur géographique</u> : ce moteur géographique est moins performant que les autres solutions présentées.</p> <p><u>Évolutivité</u> : cette solution ne présente pas d'avantages de fonctionnalités en cas d'évolution de la base de données.</p>

Tableau 9 : Analyse du SGBD Microsoft SQL Server

Avantages	Limitations
<p><u>Variété de versions</u> : une version gratuite existe avec les fonctionnalités de base et suffisantes pour les petits serveurs.</p>	<p><u>Payant</u> : l'utilisation de ce SGBD nécessite plus de fonds pour réaliser le projet.</p> <p><u>Compatibilité OS</u> : cette solution peut s'utiliser sous Windows ou Linux mais n'est pas adaptée au fonctionnement macOS X. De plus, l'utilisation est la plus efficace et facile à prendre en main sous Windows.</p> <p><u>Capacité d'administration élevée</u> : ce SGBD nécessite des compétences importantes pour fonctionner, ainsi que des ordinateurs puissants. Il fonctionne bien pour des gros projets mais trop complexe pour le projet <i>e-Sylve</i>.</p>

Tableau 10 : Analyse du SGBD PostgreSQL

Avantages	Limitations
<p><u>Open source</u> : PostgreSQL est totalement gratuit.</p> <p><u>Fiabilité</u> : il possède une grande robustesse et fiabilité.</p> <p><u>Capacité d'administration</u> : les bases de données peuvent être facilement reprises par un faible nombre d'administrateurs.</p> <p><u>Évolutivité</u> : de nombreux outils gratuits peuvent être intégrés à PostgreSQL afin de faire évoluer les bases de données.</p>	<p><u>Prise en main</u> : la documentation est parfois incohérente³, ce qui rend la prise en main parfois difficile.</p>

3.1.1.4. Choix pour la solution

À l'aide des différentes analyses des outils réalisées dans la partie précédente, leur comparaison, selon les critères de sélection déterminés plus haut, est présentée dans le Tableau 11.

Des poids ont ensuite été attribués aux critères en fonction de leur importance dans le projet et pour l'équipe de programmation.

La compatibilité du système d'exploitation (OS) et la prise en main ne sont pas des critères primordiaux. En effet, la plupart des systèmes sont compatibles sur tous les OS et une équipe de programmeurs professionnels pourra avoir le temps de se former à un outil si c'est le plus approprié pour le projet.

Les deux critères jugés les plus importants sont le coût et la capacité d'administration. L'équipe qui reprendra le projet sera certainement une petite équipe, liée aux organismes de recherche associés.

ANALYSE TECHNIQUE

Les fonds alloués ne pourront sûrement pas être aussi importants que pour une grande entreprise. Le système choisi ne doit donc pas être trop onéreux. Concernant la capacité d'administration, la base de données développée doit pouvoir être facilement reprise si les équipes sont amenées à changer.

Enfin, chaque outil a reçu une note, comprise entre 1 et 5, pour chacun des critères et la note finale est déterminée par la moyenne pondérée.

Tableau 11 : Bilan de comparaison des différents SGBD

Tableau de performance							
Solution	C1	C2	C3	C5	C6	C7	Note
	Coût	Moteur géographique	Compatibilité OS	Capacité d'administration	Prise en main	Évolutivité	
PostgreSQL	4	3	5	4	3	4	3,83
MySQL	4	1	5	4	5	2	3,33
Oracle	1	3	5	2	2	4	2,63
Microsoft SQL Server	2	3	3	2	3	3	2,55
Poids critère	0,2	0,17	0,1	0,25	0,1	0,18	

En absence de contraintes imposées par la structure qui reprendra le projet, les quatre SGBD répondent à la demande. Néanmoins sur les critères considérés, MySQL et PostgreSQL semblent plus adaptés. En effet, Oracle est une solution plus adaptée pour les grandes entreprises (prix élevé, fonctionnalités avancées, nécessite des administrateurs formés sur la solution) et Microsoft SQL Server apparaît moins intéressant au regard des critères d'évaluation évoqués. MySQL et PostgreSQL sont gratuits et largement suffisants pour les besoins de l'application. Néanmoins, PostgreSQL présente une extension pour manipuler les données à références spatiales (PostGis) mieux documentée que celle de MySQL. En effet, PostGis a été créé en avril 2001. En revanche, pour MySQL, la gestion des données à références spatiales n'a été intégrée que dans la version 8.0.0 datant de septembre 2016.

De ce fait, PostgreSQL est préconisé pour le produit final.

3.1.2. Langages et technologies de développement

3.1.2.1. Besoin de la solution

Les besoins du produit final englobent une intégration fluide avec des API cartographiques, des requêtes efficaces vers une base de données, et une gestion sécurisée des profils et des sessions. Pour répondre à ces exigences, un langage de programmation polyvalent est recherché pour le *back-end*, qui soit à la fois simple d'utilisation et capable de gérer les interactions entre la cartographie, la base de données, et la gestion des utilisateurs.

Après discussion avec les commanditaires, il n'y a en revanche pas de contraintes quant au langage de programmation des requêtes ou au langage de programmation du *back-end*. Les objectifs de cette veille sont donc de trouver :

- Le langage de programmation pour les requêtes réalisées vers les serveurs et les bases de données des commanditaires ;
- Le langage de programmation du *back-end* des commanditaires.

3.1.2.2. Outils existants

Il existe de nombreuses technologies de base, chacune ayant ses forces et ses faiblesses. Toutefois, pour les besoins de cette analyse, seules les plus populaires ont été sélectionnées. Les différents outils qui sont détaillés sont :

- JavaScript ;
- Python ;
- Ruby ;
- PHP.

JavaScript

Grâce à des technologies telles que Node.js, JavaScript n'est plus seulement un langage de script côté client (*front-end*), il permet d'effectuer des opérations côté serveur (*back-end*) grâce à des *frameworks* appelés Express.js. (Tableau 12)

Tableau 12 : Analyse du langage JavaScript

Avantages	Limitations
<u>Développement rapide</u> : possibilité de développer dans le même langage le <i>back-end</i> et le <i>front-end</i> .	<u>Anciennes versions peu intuitives</u> : fonctions événements parfois compliquées (résolu dans les dernières versions).

ANALYSE TECHNIQUE

<p><u>Moins de frais de codage</u> : certaines fonctions intégrées (DOM, hoops...) augmentent l'efficacité et les performances.</p> <p><u>Avantage de l'open source</u> : communautés actives de développeurs <i>back-end</i> et <i>front-end</i>.</p>	<p><u>Frameworks et MySQL</u> : l'hébergement des <i>frameworks</i> de base de JavaScript avec la base de données MySQL est compliqué.</p>
--	--

Python

Il s'agit de l'un des langages de programmation les plus polyvalents, grâce à ses nombreuses bibliothèques. C'est le troisième langage de programmation le plus apprécié dans l'enquête annuelle de StackOverflow Developer Survey en 2020 (*Stack Overflow Developer Survey 2020*, s. d.). (Tableau 13)

Tableau 13 : Analyse du langage Python

Avantages	Limitations
<p><u>Relativement facile à apprendre</u> : coder et lire des codes Python est relativement facile pour les programmeurs débutants et expérimentés.</p> <p><u>Bibliothèques</u> : les énormes bibliothèques disponibles avec Python réduisent la nécessité d'écrire des codes manuellement.</p> <p><u>Rentabilité</u> : offre de nombreux outils et ressources gratuits qui améliorent les projets de développement d'applications.</p> <p><u>Caractéristiques de l'IdO</u> : les développeurs peuvent s'appuyer sur les fonctionnalités modernes de Python pour créer des objets physiques Raspberry Pi.</p>	<p><u>Rapidité d'exécution</u> : l'exécution du code Python devient lente lorsqu'elle est interrompue. Cette lacune nuit à l'ensemble du projet de développement de l'application.</p> <p><u>Accès BDD</u> : la couche d'accès à la base de données Python est moins développée que les autres langages de programmation.</p> <p><u>Détection des bogues</u> : des tests approfondis sont nécessaires pour détecter les erreurs et les bogues dans les codes Python.</p> <p><u>Bibliothèques</u> : il est fortement dépendant des <i>frameworks</i> et des bibliothèques de tiers.</p>

Ruby

Ce langage possède une syntaxe similaire à celle de Python ou encore Java. Il s'agit d'un langage apprécié pour le développement d'applications web et pour le prototypage. (Tableau 14)

Tableau 14 : Analyse du langage Ruby

Avantages	Limitations
-----------	-------------

<p><u>Hautement productif</u> : il s'agit d'un langage ayant une très grande bibliothèque, donc beaucoup de tâches peuvent être réalisées via divers outils.</p> <p><u>Rapide et fiable</u> : la méthode de programmation orientée objet rend ce langage plus résistant et facilite les requêtes et le traitement des résultats.</p>	<p><u>Temps d'exécution</u> : ce langage de programmation a un temps d'exécution plus long que la majorité des autres langages.</p> <p><u>Apprentissage</u> : Ruby est un langage de programmation difficile à apprendre.</p>
--	---

PHP

C'est un langage de programmation polyvalent et simple à utiliser. Il s'agit de l'un des principaux langages utilisés pour le côté serveur. En effet, une étude du site w3techs.com (*Usage Statistics and Market Share of PHP Version 7 for Websites, March 2024, s. d*) montre que 76,5 % des sites web dans le monde sont codés en PHP. (Tableau 15)

Tableau 15 : Analyse du langage PHP

Avantages	Limitations
<p><u>Populaire</u> : un des principaux langages de programmation côté serveur du monde actuel.</p> <p><u>Open source et polyvalent</u> : il existe de nombreuses bibliothèques PHP gratuites en ligne exploitables.</p> <p><u>Facile à utiliser</u> : la simplicité de PHP en fait un langage idéal pour les nouveaux développeurs.</p> <p><u>Automatisation des fonctions</u> : la fonction de script de PHP le rend utile pour créer des automatismes tels que l'authentification, le mappage d'URL ou encore la gestion de session.</p>	<p><u>Baisse de popularité</u> : l'influence du PHP en tant que technologie de développement s'amenuise. Aujourd'hui, les gens incluent à peine le PHP dans leurs compétences de développement.</p> <p><u>Bibliothèques</u> : le PHP ne peut pas concurrencer efficacement les technologies de développement modernes comme Ruby et Python en raison du nombre limité de bibliothèques.</p>

3.1.2.3. Comparaison des outils (définition des critères, poids et valeurs)

D'après l'analyse de l'existant réalisée ci-dessus, plusieurs critères ressortent :

- Facilité d'utilisation ;
- Rapidité d'exécution ;
- Fiabilité/sécurité ;
- Bibliothèques.

Un score a été attribué à chaque solution pour chaque critère selon la bibliographie. Il varie de 0 à 5. Celui-ci est couplé avec le poids du critère ce qui a permis d'obtenir une note finale afin de classer les différentes solutions.

ANALYSE TECHNIQUE

3.1.2.4. Choix pour la solution

La comparaison des différents outils est réalisée à l'aide des critères et des analyses effectuées.

Tableau 16 : Critères de sélection des différents langages de programmation *back-end* et leurs notes finales

Tableau de performance					
Solution	C1	C2	C3	C4	Note
	<i>Facilité d'utilisation</i>	<i>Rapidité d'exécution</i>	<i>Fiabilité/sécurité</i>	<i>Bibliothèques</i>	
<i>JavaScript</i>	4	4	4	3	3,85
<i>PHP</i>	5	3	4	3	3,75
<i>Ruby</i>	4	2	5	4	3,55
<i>Python</i>	3	3	4	4	3,4
Poids critère	0,25	0,35	0,25	0,15	

D'après le Tableau 16, la meilleure solution pour le langage du *back-end* serait JavaScript, qui est adapté à la fois pour le *back-end* mais également pour le *front-end*. PHP serait le deuxième choix pour le *back-end*. JavaScript arrive en première position du fait de la rapidité d'exécution, critère au poids plus important que celui de la facilité d'utilisation.

Le JavaScript est préconisé pour le produit final.

3.1.3. Serveur web

3.1.3.1. Besoins de la solution

Un serveur web est un programme qui utilise le protocole HTTP pour fournir les fichiers qui constituent les pages web. Ces fichiers sont ceux que les utilisateurs ont demandés via des requêtes transmises par les clients HTTP (logiciel conçu pour se connecter à un serveur HTTP) de leurs ordinateurs. Dans le cas d'un serveur local, c'est l'ordinateur qui fait office de serveur.

Pour choisir un serveur web pour le projet, il y a plusieurs critères à prendre en compte :

- La compatibilité avec le système d'exploitation ;
- La capacité à prendre en charge la programmation côté serveur ;

- La sécurité ;
- Le nombre de connexions simultanées ;
- Les outils particuliers fournis (pour la publication, le moteur de recherche, la création de site).

3.1.3.2. Outils existants

Les principaux serveurs web sont :

- Apache (le plus répandu) ;
- NGINX ;
- IIS (Internet Information Server) ;
- Netware ;
- Google Web Server.

Les deux premiers sont les deux plus importants serveurs web, sollicités par plus de la moitié des sites. La comparaison portera donc sur ces deux solutions possibles.

3.1.3.3. Comparaison des outils (définition des critères, poids et valeurs)

La comparaison des deux serveurs web, Apache et NGINX, suivant différents critères est illustré dans le Tableau 17.

Tableau 17 : Comparaison des deux serveurs web Apache et NGINX

Critères d'évaluation	Apache	NGINX
Compatibilité OS	Multiplateforme	Multiplateforme, performances limitées sur Windows
Coût de la licence	Gratuit	Gratuit
Programmation côté serveur	PHP, Python, Perl, etc.	PHP, Python, Perl, etc.
Sécurité	Bonne	Élevée
Modules	50 modules intrinsèques chargés dynamiquement	Configuration légère : 100 modules chargés non dynamiquement
Connexions simultanées	Difficultés en grand nombre	Gestion efficace
Traitement d'applications dynamiques	Bon	Besoin de logiciels externes
Empreinte mémoire	Moyenne	Faible

ANALYSE TECHNIQUE

3.1.3.4. Choix pour la solution

De même que pour le choix du langage de programmation, un score variant de 0 à 5 a été attribué à chaque solution pour chaque critère. L'utilisation du poids critère permet de donner la note finale. (Tableau 18)

Tableau 18 : Critères de sélection des serveurs web et leurs notes finales

Tableau de performance									
Solutions	C1	C2	C3	C4	C5	C6	C7	C8	Note
	<i>Compatibilité OS</i>	<i>Coût</i>	<i>Programmation côté serveur</i>	<i>Sécurité</i>	<i>Modules</i>	<i>Connexions simultanées</i>	<i>Traitement d'applications dynamiques</i>	<i>Empreinte mémoire</i>	
NGINX	4	5	5	4	4	5	2	4	4,34
Apache	5	5	5	3	5	3	4	3	4
Poids critère	0,14	0,17	0,14	0,17	0,03	0,14	0,05	0,17	

Bien que NGINX soit plus compliqué à prendre en main qu'Apache, il semble être le meilleur candidat pour servir de serveur web au projet *Tree 2 1*.

NGINX est donc préconisé pour le produit final.

3.1.4. Serveur cartographique

Un serveur cartographique est un logiciel qui traduit des données cartographiques en un format lisible par un serveur web (SOGEFI). Il est à l'interface entre les données cartographiques aux formats SIG et le serveur web. À chaque requête normalisée (WMS par exemple) du navigateur, le serveur web la transfère au serveur cartographique. Il transmet une requête à la base de données pour récupérer les données brutes et construire la carte ou la couche demandée. Il la traduit alors en un format lisible par le serveur web pour que ce dernier l'affiche sur le navigateur.

Plusieurs fonctionnalités de la solution nécessitent l'intégration de cartes. Ces cartes doivent être interactives et évolutives.

- La partie interactive est réalisée à l'aide d'une API cartographique détaillée ultérieurement dans la partie 3.3 *front-end*. Le projet doit, pour répondre aux fonctionnalités de

visualisation des sorties du modèle au format cartographique, être capable d'afficher à chaque modification de paramètres d'entrée ou de variable à visualiser, une nouvelle couche ;

- La solution a un important besoin dans la dynamique de son affichage. Il est possible d'utiliser un serveur cartographique pour le développement du produit final mais il ne sera pas forcément nécessaire. En effet, l'utilisation d'un serveur cartographique sera justifiée si la quantité de données à afficher simultanément est très importante et si elle entraîne une perte de performance dans l'affichage cartographique. Actuellement, l'affichage de l'ensemble des mailles SAFRAN sur une carte est suffisamment rapide pour se passer d'un serveur cartographique.

Le produit final ne nécessite pas de serveur cartographique.

3.2. *Front-end* (côté client)

Le *front-end* est l'ensemble des éléments que l'utilisateur voit et avec lesquels il peut interagir.

3.2.1. Langages et technologies de développement

3.2.1.1. Besoins de la solution

L'un des objectifs du projet est d'améliorer l'ergonomie du site web *Forêts-21*. Ainsi, les outils pour développer le *front-end* doivent permettre de pouvoir proposer un site moderne, convivial et ergonomique pour les utilisateurs, qui présente de manière efficace l'accès aux données et aux fonctionnalités. Il sera également nécessaire de pouvoir gérer la responsivité (entre ordinateur et smartphone), c'est-à-dire que l'interface doit s'adapter aux différentes tailles d'écrans .

Concernant le langage de programmation et le *framework* utilisé, les habitudes de l'équipe qui reprendra le projet (non connue actuellement) devraient être déterminantes. Cependant, les commanditaires n'ont fait remonter aucune contrainte.

3.2.1.2. Outils existants

Pour la veille du *front-end*, les langages de programmation *front-end* et des *frameworks* ou bibliothèques JS et CSS seront étudiés.

3.2.1.2.1. Langage de programmation

De nombreux langages de programmation *front-end* existent : Javascript, CSS, HTML, SASS, Swift et Elm. Actuellement, les trois langages les plus couramment utilisés sont HTML, JavaScript et CSS, bien que les autres soient de plus en plus utilisés. Ces trois langages ont été sélectionnés pour la veille.

HTML (*Hypertext Markup Language*)

HTML est un langage de balisage standard utilisé dans la création de sites et de pages internet. (12 *Avantages et inconvénients du HTML*, 2023) (Tableau 19)

Tableau 19 : Récapitulatif du langage HTML

Avantages	Limitations
<p><u>Prise en main</u> : l'HTML est relativement facile à apprendre.</p> <p><u>Compatibilité</u> : ce langage est compatible avec la grande majorité des navigateurs web.</p> <p><u>Chargement des pages</u> : HTML permet une charge rapide des pages.</p>	<p><u>Anciennes versions peu intuitives</u> : fonctions événements parfois compliquées (résolu dans les dernières versions).</p> <p><u>Frameworks et MySQL</u> : l'hébergement des <i>frameworks</i> de base de JavaScript avec la base de données MySQL est compliqué.</p> <p><u>Insuffisant pour des visuels dynamiques</u> : pour ce genre de visuels stylisés, ou encore pour les mesures de sécurité, ce langage doit être combiné à d'autres langages.</p>

CSS (*Cascading Style Sheets*)

CSS ou feuille de style en cascade est un langage descriptif qui permet de gérer la représentation d'autres langages comme HTML ou XML. (« Feuilles de style en cascade », 2023) (Tableau 20)

Tableau 20 : Récapitulatif du langage CSS

Avantages	Limitations
<p><u>Personnalisation</u> : CSS permet une grande personnalisation de la mise en page.</p>	<p><u>Complexe</u> : ce langage est difficile à apprendre et la création de styles avancés est difficile car la structuration du code est difficile.</p>

JavaScript

JavaScript permet de créer des fonctionnalités interactives et dynamiques sur une page ou site web. Il est à l'origine utilisé pour le développement du *front-end* mais est désormais aussi utilisé pour le développement *back-end* depuis plusieurs années. (« Coding », 2021) (Tableau 21)

Tableau 21 : Récapitulatif du langage JavaScript

Avantages	Limitations
<p><u>Compatibilité</u> : ce langage est compatible avec la</p>	<p><u>Apprentissage</u> : ce langage peut être difficile à</p>

plupart des navigateurs.

Interopérabilité : ce langage est interopérable avec d'autres langages de *front-end* et de *back-end*.

apprendre.

Sécurité : une mauvaise utilisation de JavaScript entraîne de grands problèmes de sécurité.

3.2.1.2.2. Framework/bibliothèque JavaScript

Le développement *front-end* s'appuie souvent sur un *framework* ou des bibliothèques spécifiques, qui permettent d'aider les développeurs à créer des fonctionnalités et des éléments visuels avancés de manière plus efficace. La littérature fait la distinction entre les outils JS et CSS.

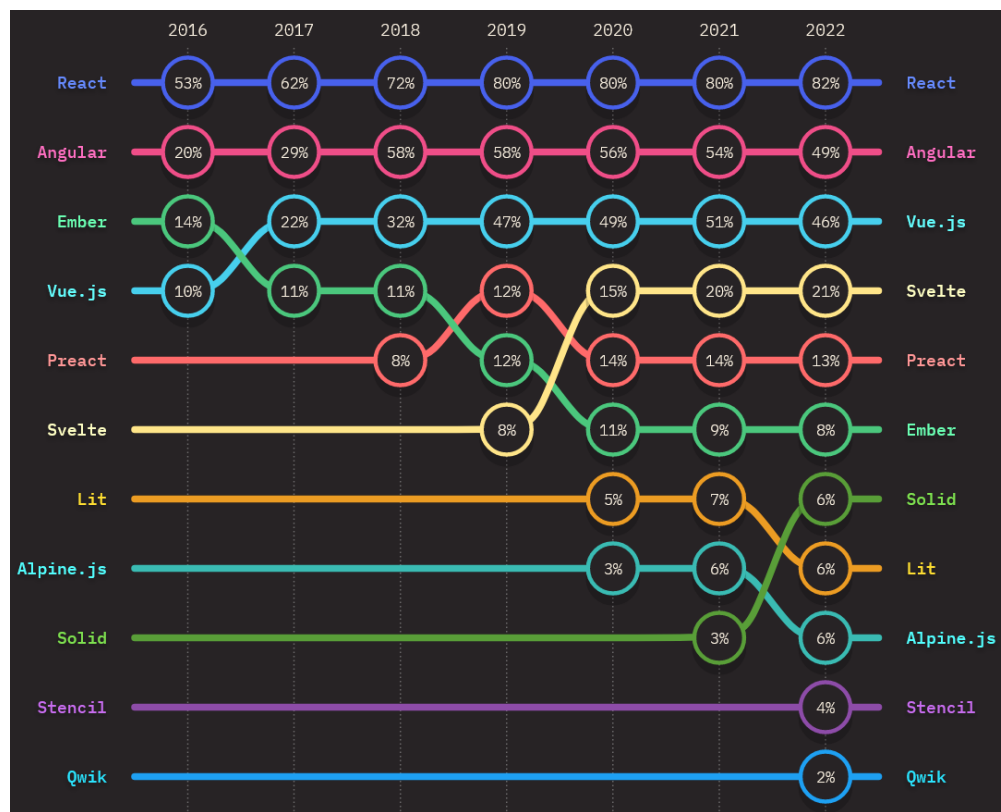


Figure 4 : Usage ratio of front-end frameworks and libraries over time (State of JavaScript 2022, s. d.)

D'après la Figure 4 de StateOfJS qui porte sur la satisfaction de l'utilisation du JavaScript en 2022, les trois *frameworks* ou bibliothèques JavaScript les plus appréciés sont React (82 %), Angular (49 %) et Vue.js (46 %) (les résultats de l'étude de 2023 ne sont pas encore disponibles). Les autres technologies étant beaucoup moins appréciées (moins de 25 %), elles ne seront pas développées lors de cette veille. (State of JavaScript 2022, s. d.). Il a donc été décidé de porter les recherches sur ces trois *frameworks*.

React

ANALYSE TECHNIQUE

React est, depuis plusieurs années, la technologie la plus utilisée (StateOfJS) pour gérer le *front-end*. C'est une bibliothèque de construction d'interface utilisateur, développée en open source par Meta depuis 2013. (Tableau 22)

Tableau 22 : Récapitulatif de la bibliothèque React

Avantages	Limitations
<p><u>Open source</u> : développée par Meta depuis 2013.</p> <p><u>Flexible et adaptable</u> : du code HTML peut être utilisé en même temps que JavaScript. React ne possède pas de règles strictes de convention de codes, il est donc utilisable pour de nombreux projets et habitudes de programmation (<i>Démarrer avec React - Apprendre le développement web</i> MDN, 2023).</p> <p><u>Responsive</u> : celle-ci peut être gérée via des packages disponibles (<i>React-Responsive</i>, 2022).</p>	<p><u>Documentation obsolète</u> : React évolue très vite et la documentation devient très vite obsolète (Maslov, 2023).</p> <p><u>Outil de compilation nécessaire</u> : la syntaxe JSX utilisée par React ne peut pas être lue par un navigateur et nécessite donc ce genre d'outil.</p>

Angular

Angular est un *framework open source* développé et maintenu par Google depuis 2010. Il utilise TypeScript, un sur-ensemble de JavaScript permettant la programmation orientée objet et le repérage d'erreurs courantes de code. (*Pourquoi utiliser Angular ?* | Angular, s. d.) (Tableau 23)

Tableau 23 : Récapitulatif du framework Angular

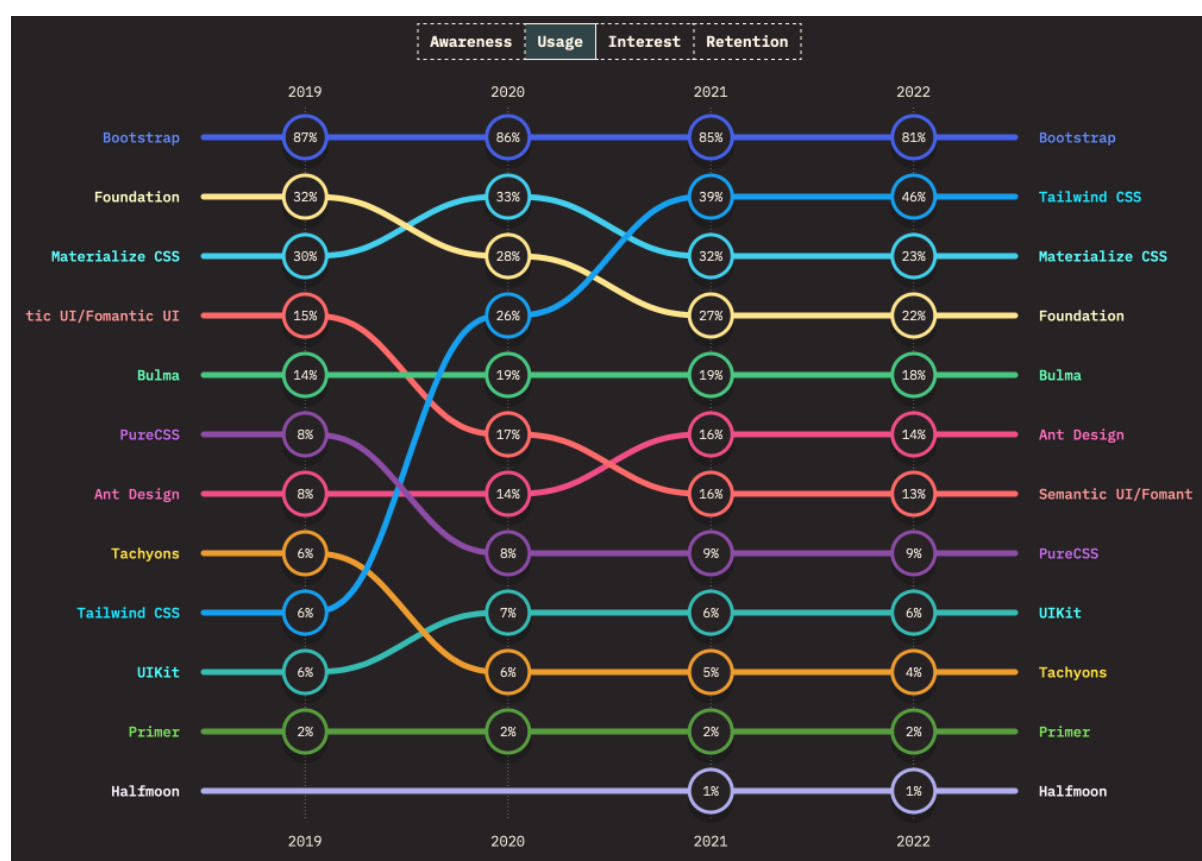
Avantages	Limitations
<p><u>Open source</u> : développé et maintenu par Google depuis 2010.</p> <p><u>Grande communauté</u> : très utilisé par les développeurs professionnels, Angular possède de nombreuses ressources en ligne pour apprendre à l'utiliser.</p> <p><u>Responsivité adaptable</u> : la responsivité des sites développés avec Angular peut être gérée de différentes façons notamment avec des bibliothèques.</p>	<p><u>Complexe</u> : c'est un <i>framework</i> complexe le rendant difficile à apprendre pour les développeurs débutants d'autant plus qu'il utilise le TypeScript.</p> <p><u>Peu flexible</u> : ce <i>framework</i> est très structuré, le rendant pas assez flexible pour certains projets.</p> <p><u>Volumineux</u> : cela rallonge les temps de chargements des pages développées.</p>

Vue.js

Vue.js est un *framework* lancé en 2013. (Chahine, 2020) (Tableau 24)

Tableau 24 : Récapitulatif du *framework* Vue.js

Avantages	Limitations
<p><u>Léger et performant</u> : Vue.js est bien plus léger (58,8 kB) que ses concurrents Angular (143 kB) et React (97,5 kB) et très performant.</p> <p><u>Prise en main</u> : ce <i>framework</i> est facile à prendre en main malgré le fait qu'il ait été initialement développé en mandarin, laissant ainsi encore de nombreux éléments et descriptions en mandarin au lieu de l'anglais.</p> <p><u>Responsive design</u> : celle-ci peut être gérée avec des bibliothèques ou des packages.</p>	<p><u>Documentation</u> : la communauté est assez petite et présente donc peu de documentation.</p>

3.2.1.2.3. *Framework/bibliothèque CSS***Figure 5 : Usage ratio of CSS Frameworks over time (State of CSS 2022, s. d.)**

D'après la Figure 5 les deux *framework*/bibliothèques CSS les plus appréciés en 2022 (données non disponibles pour 2023) sont Bootstrap et Tailwind CSS. Les *frameworks* ou bibliothèques qui les suivent sont bien moins appréciés, avec une appréciation inférieure à 23 %, ils ne seront donc pas étudiés dans cette veille.

Bootstrap

Bootstrap est une bibliothèque CSS gratuite et *open source* destinée au développement *front-end web responsive*. (Quels sont les meilleurs frameworks CSS en front-end ?, 2022) (Tableau 25)

Tableau 25 : Récapitulatif de la bibliothèque Bootstrap

Avantages	Limitations
<p><u>Open source</u> : cette bibliothèque est gratuite.</p> <p><u>Modèle de design</u> : disponible en CSS, HTML, JavaScript (optionnel) pour les différents éléments de l'interface utilisateur.</p> <p><u>Développement facilité</u> : Bootstrap permet de développer rapidement une interface visuelle de qualité, <i>responsive</i> et hautement personnalisable.</p> <p><u>Rapport temps de formation/rendu</u> : un temps de formation est nécessaire pour la prise en main, mais il est raisonnable compte tenu du temps ensuite gagné pour réaliser un design propre et joli de l'interface.</p> <p><u>Personnalisation</u> : il permet de personnaliser des modèles d'éléments déjà existants ce qui permet de gagner du temps.</p> <p><u>Grande communauté active</u> : étant l'une des bibliothèques CSS les plus utilisées, elle possède une large documentation.</p>	<p><u>Apparence générique</u> : l'utilisation répandue de composants et styles par défaut de Bootstrap peut conduire à une apparence générique des sites web, rendant complexe la création d'une identité visuelle unique.</p> <p><u>Taille du fichier</u> : la richesse des fonctionnalités de Bootstrap entraîne une taille de fichier importante lors de son intégration, ce qui peut nuire aux performances du site, surtout sur des connexions Internet plus lentes.</p> <p><u>Personnalisation limitée</u> : bien que Bootstrap offre une certaine personnalisation, ses limites peuvent se manifester face à des besoins de personnalisation approfondis, nécessitant parfois des surcharges CSS substantielles pour obtenir le design et l'ergonomie désirés.</p> <p><u>Dépendance JavaScript</u> : la dépendance à JavaScript de certains composants de Bootstrap peut poser un inconvénient, en particulier dans des projets où l'utilisation de JavaScript est limitée ou non désirée.</p>

Tailwind CSS

Tailwind CSS est une bibliothèque CSS *open source*. L'une de ses particularités est de ne pas proposer une série de classes prédéfinies pour les éléments mais une liste de classes CSS "utilitaires" qui peuvent être utilisées pour chaque élément, à la différence de Bootstrap. (« 10 frameworks CSS à utiliser en 2024 - Codeur Blog », 2022) (*Deux années avec Tailwind CSS, quel bilan?*, 2022). (Tableau 26)

Tableau 26 : Récapitulatif de la bibliothèque Tailwind CSS

Avantages	Limitations
<p><u>Open source</u> : cette bibliothèque est gratuite.</p> <p><u>Personnalisation</u> : tout peut être personnalisé.</p> <p><u>Formation relativement rapide</u> : elle dure environ une demi-journée et est relativement facile à prendre</p>	<p><u>Personnalisation</u> : elle se fait de zéro, ce qui prend du temps et elle alourdit le fichier HTML.</p> <p><u>Modèles payants</u> : pour la personnalisation, des modèles sont disponibles mais à acheter.</p>

en main.	<u>Bibliothèque volumineuse</u> : cette bibliothèque est assez lourde lors de l'import dans le projet.
----------	--

3.2.1.3. Comparaison des outils (définition des critères, poids et valeurs)

En tenant compte des informations trouvées lors de la veille et des besoins de la solution, des critères de sélection ont été déterminés qui sont présentés dans le Tableau 27.

Tableau 27 : Critères d'évaluation pour l'analyse des outils de développement front-end

Critères d'évaluation	Description
Flexibilité	Personnalisation du rendu final et de la structure de programmation, pour s'adapter le plus possible au projet et aux habitudes de l'équipe de programmation.
Performance	Temps de chargement des pages notamment pour celles qui contiendront les éléments graphiques et cartographiques.
Responsive design	Capacité à s'adapter à différentes tailles d'écran, doit être facilement mise en place pour que la solution soit utilisable sur le plus grand nombre de terminaux.
Facilité de prise en main ou apprentissage	Critère qui prend en compte le temps et la complexité à se former pour utiliser correctement l'outil. Le temps et la complexité étant liés, plus l'outil sera complexe plus cela prendra du temps pour s'y former.
Disponibilité des ressources	Nombre et qualité des documentations explicatives de l'outil, mais aussi la taille et l'activité des communautés des utilisateurs de l'outil, permettant de fournir une aide et un support en cas de questionnement ou d'apprentissage pour l'équipe de programmation.

Des poids ont ensuite été attribués aux critères en fonction de leur importance dans le projet et pour l'équipe de programmation. Ainsi, pour une équipe de programmation professionnelle, la facilité de prise en main d'un outil n'est pas le critère principal, les programmeurs pourront prendre le temps de se former si l'outil est celui qui correspond le plus au projet. En revanche, la flexibilité de l'outil et sa performance sont primordiales pour la réalisation du projet.

Enfin, chaque outil a reçu une note, comprise entre 1 et 5, pour chacun des critères et la note finale est déterminée par la moyenne pondérée. (Tableau 28)

Tableau 28 : Critères de sélection des frameworks et bibliothèques front-end et leurs notes finales

Tableau de performance

ANALYSE TECHNIQUE

Solution	C1	C2	C3	C4	C5	Note
	<i>Facilité de prise en main</i>	<i>Flexibilité</i>	<i>Ressources disponibles</i>	<i>Responsive design</i>	<i>Performance</i>	
Bootstrap	4	4	5	5	3	4,15
Tailwind CSS	3	5	3	5	4	4,15
Vue.js	4	3	2	3	4	3,15
React	2	3	2	3	4	2,95
Angular	1	2	5	3	3	2,95
Poids critère	0,1	0,25	0,2	0,2	0,25	

3.2.1.4. Choix pour la solution

Pour une équipe n'ayant pas de préférence particulière en développement *front-end*, **l'utilisation de Bootstrap ou Tailwind CSS est préconisée pour le produit final**. Ils permettent une grande personnalisation de l'interface utilisateur et une grande flexibilité d'utilisation et les langages utilisés sont les plus couramment utilisés.

3.2.2. Cartographie interactive

3.2.2.1. Besoins de la solution

Concernant la cartographie graphique sur le web, le choix de la solution et de son architecture dépend des besoins d'utilisation, notamment au vu de l'interaction envisagée, l'affichage dynamique d'éléments ainsi que l'ajout de données.

Il a été vu précédemment que le site doit présenter une interactivité, notamment au niveau de la navigation sur la carte, ainsi que de l'affichage d'informations au survol de la souris ou à un appui long avec le doigt. De fait, les solutions non-interactives ne sont pas envisagées dans la comparaison des outils existants. En effet, il s'agit d'un critère discriminant.

3.2.2.2. Outils existants

Au-delà des outils existants, il existe différentes architectures qui peuvent être constituées :

- D'une simple API cartographique ;
- De systèmes intégrés avec le SGBD qui communique avec un serveur ou une API cartographique au niveau du *back-end*.

Ces modèles plus complexes sont étudiés dans le *back-end*.

Au niveau du *front-end*, l'interactivité peut être prise en compte avec des API qui intègrent des fonctions JavaScript telles que :

- Leaflet (*Leaflet — an Open-Source JavaScript Library for Interactive Maps*, n.d.) ;
- Google Maps JavaScript API (*Google Maps Platform Documentation | Maps JavaScript API*, n.d.) ;
- OpenLayers (*OpenLayers - Welcome*, n.d.) ;
- ArcGIS Maps SDK for JavaScript (*ArcGIS Maps SDK for JavaScript | Overview | ArcGIS Maps SDK for JavaScript 4.28 | ArcGIS Developers*, n.d.).

Ces différents outils vont être comparés selon les besoins du site.

3.2.2.3. Comparaison des outils (définition des critères, poids et valeurs)

D'après les analyses détaillées précédemment, les principaux besoins en termes de fonctionnalités interactives sont les suivants :

- Une interactivité de « base » qui contient notamment le zoom/dézoom avec deux boutons « + » et « - », deux doigts sur un pavé tactile, un écran ou encore la molette d'une souris, la navigation sur la carte en appuyant sur celle-ci ;
- Une emprise de la carte sur une certaine zone lors de l'affichage ;
- Un traçage de zone d'intérêt, donc un mode « traçage » ;
- Une barre de recherche pour se situer au niveau d'une commune ;
- Un choix de fond de carte.

Par ailleurs, au-delà des critères de sélection fonctionnels spécifiques au site, des critères (*non-*)fonctionnels non spécifiques aux besoins du site sont à considérer. En particulier,

- La taille du code et la performance qui en suit ;
- La disponibilité d'une documentation et éventuellement de communauté pour le support ;
- La compatibilité avec les principaux navigateurs web ;
- Le type de licence et le coût associé à l'utilisation de l'outil ;
- La facilité d'utilisation au niveau du développement et de la configuration.

Ainsi, les critères de sélection sont résumés dans le Tableau 29.

Tableau 29 : Récapitulatif des critères de sélection des outils de cartographie interactive

Critères d'évaluation	Description
Interactivité de « base »	Niveau d'interactivité simple, détaillé précédemment.

ANALYSE TECHNIQUE

Traçage	Option de traçage sur la carte.
Barre de recherche	Option de recherche de commune.
Choix du fond de carte	Option qui permet de choisir le fond de carte.
Taille du code et performance	Taille du code généré et impact sur la performance du site.
Documentation et support	Disponibilité d'une documentation et de support.
Compatibilité	Compatibilité avec les principaux navigateurs web.
Type de licence et coût	Le type de licence et le coût associé à l'utilisation de l'outil.

Les différents outils énoncés précédemment, à savoir Leaflet, Google Maps JavaScript API, OpenLayers et ArcGIS Maps SDK for JavaScript, sont analysés individuellement au vu des différents critères de sélection préalablement décrits. Les analyses sont résumées dans le Tableau 30.

Tableau 30 : Comparaison des outils de cartographie interactive

Critères d'évaluation	Leaflet	Google Maps JavaScript API	OpenLayers	ArcGIS Maps SDK for JavaScript
Interactivité de « base »	Intégré. (<i>Leaflet Features — an Open-Source JavaScript Library for Interactive Maps</i> , n.d.)	Intégré. (<i>Overview Maps JavaScript API</i> , n.d.)	Intégré. (<i>OpenLayers Examples Control</i> , n.d.)	Intégré. (<i>ArcGIS Maps SDK for JavaScript Overview ArcGIS Maps SDK for JavaScript 4.28 ArcGIS Developers</i> , n.d.)
Traçage	Via l'utilisation de <i>plug-in</i> . (West, 2018)	Disponible sous le nom <i>overlays</i> . (<i>Drawing on the Map Maps JavaScript API</i> , n.d.)	Intégré. (<i>OpenLayers Examples Draw</i> , n.d.)	Intégré. (<i>Draw API Reference ArcGIS Maps SDK for JavaScript 4.28 ArcGIS Developers</i> , n.d.)
Barre de recherche	Via l'utilisation de <i>plug-in</i> . (<i>Plugins - Leaflet - a JavaScript Library for Interactive Maps</i> , n.d.)	Intégré. (<i>Places Search Box Maps JavaScript API</i> , n.d.)	Via l'utilisation de <i>plug-in</i> . (Cavailhez, 2023)	Intégré. (<i>Find Places or Search for an Address ArcGIS Maps SDK for Unity ArcGIS Developers</i> , n.d.)

Choix de fond de carte	Via l'utilisation ou non de <i>plug-in</i> . (Shramov & Bergot, 2021) & (Seelmann, 2021)	Disponible sous l'option <i>map type</i> . (<i>Cloud-Based Maps Styling Overview</i> <i>Maps JavaScript API</i> , n.d.)	Via l'appel d'API. (<i>OpenLayers Examples Tile</i> , n.d.)	Intégré. (<i>Featured Samples Map View</i> <i>Sample Code</i> <i>ArcGIS Maps SDK for JavaScript 4.28</i> <i>ArcGIS Developers</i> , n.d.)
Taille du code et performance	Taille hors <i>plug-in</i> additionnels est seulement estimée à 42 kb. (<i>Leaflet — an Open-Source JavaScript Library for Interactive Maps</i> , n.d.)	Niveau de performance annoncé comme acceptable avec guide de performance. (<i>Support</i> <i>Maps JavaScript API</i> , n.d.) & (<i>Optimization Guide</i> <i>Google Maps Platform</i> , n.d.)	Performance acceptable au-dessous d'un très grand nombre de polygones affichés. (<i>OpenLayers Performance</i> , n.d.)	Performance acceptable au-dessous d'un très grand nombre de polygones affichés et variable selon le terminal. (Rubalcava & Powell, 2022)
Documentation et support	Tutoriels et guides d'utilisation existant ainsi qu'une communauté active. (<i>Tutorials - Leaflet - a JavaScript Library for Interactive Maps</i> , n.d.), (<i>Documentation - Leaflet - a JavaScript Library for Interactive Maps</i> , n.d.) & (<i>Newest "leaflet" Questions</i> , n.d.)	Documentation complète assortie d'exemple et d'option de support. (<i>Google Maps Platform Documentation</i> , n.d.)	Documentation et exemples explicatifs. (<i>OpenLayers - Welcome</i> , n.d.)	Documentation complète disponible sur le site du vendeur. (<i>Documentation</i> <i>ArcGIS Developers</i> , n.d.)
Compatibilité	Absence de dépendances donc disponible sur les principaux moteurs de recherche et appareils. (<i>Leaflet — an Open-Source JavaScript Library for Interactive Maps</i> , n.d.)	Accessible sur les principaux moteurs de recherche et appareils. (<i>Browser Support</i> <i>Maps JavaScript API</i> , n.d.)	Accessible sur les principaux moteurs de recherche et appareils. (<i>openlayers collective</i> , 2023)	Accessible sur les principaux moteurs de recherche et appareils (pour l'affichage 2D). (<i>Requirements—ArcGIS Web AppBuilder</i> <i>ArcGIS Developers</i> , n.d.)
Type de licence et coût	Licence <i>BSD 2-Clause "Simplified" License</i> (<i>open source</i> permissive). (Agafonkin, 2024)	Outil propriétaire et payant au-delà d'un certain niveau d'utilisation. (<i>Maps JavaScript API Usage and Billing</i> , n.d.)	Licence <i>2-clause BSD</i> qui renseigne une liberté d'utilisation. (<i>OpenLayers - Welcome</i> , n.d.)	Outil propriétaire et payant au-delà d'un certain niveau d'utilisation. (<i>Pricing</i> , n.d.)

Étant donné les résultats des veilles réalisées, le critère **C7** "Compatibilité" ne discrimine pas les solutions comparées. De ce fait, il n'apparaît pas dans le Tableau 31 ci-dessous.

Tableau 31 : Comparaison des outils de cartographie interactive à la lumière des critères de sélection associés

Tableau de performance								
Solution	C1	C2	C3	C4	C5	C6	C8	Note
	<i>Interactivité</i>	<i>Traçage</i>	<i>Recherche</i>	<i>Fond</i>	<i>Performance</i>	<i>Support</i>	<i>Licence</i>	
Leaflet	5	3	4	4	5	5	5	4,1
Google Maps JavaScript API	5	3	5	4	4	5	3	4
ArcGIS Maps SDK for JavaScript	5	4	3	4	1	3	3	3,5
OpenLayers	5	3	3	3	4	2	5	3,25
Poids critère	0,10	0,30	0,10	0,20	0,10	0,15	0,05	

3.2.2.4. Choix pour la solution

Ainsi, d'après les résultats de comparaison du Tableau 31, **l'outil préconisé est Leaflet**. Il est convenable de noter que l'outil Google Maps JavaScript API est presque autant à préconiser, sous réserve d'évaluer son coût d'usage.

3.2.3. Visualisation des données graphiques

L'intégration des graphiques joue un rôle de dynamisme dans un support visuel. Dans le cas du projet, il s'agit plutôt d'infographie animée qui vise à mettre en forme efficacement des éléments statistiques comme des graphiques.

3.2.3.1. Besoins de la solution

La solution ayant pour but de rendre accessible des données techniques, il y a un vrai besoin de lisibilité et de compréhension des différents éléments graphiques tels que les tableaux ou les

graphiques qui peuvent être assez complexes. Il faut des éléments clairs et faciles à comprendre pour tous les niveaux de technicité et accessibles quel que soit le matériel.

L'utilisateur aura la possibilité sur une même page web de choisir des paramètres d'entrée afin de visualiser des graphiques. Il est donc nécessaire, afin de limiter le temps pour afficher les données, d'actualiser certaines parties de la page sans avoir à recharger entièrement l'ensemble à chaque nouvelle requête de l'utilisateur (nouveau choix de paramètres).

3.2.3.2. Outils existants

3.2.3.2.1. Les bibliothèques utiles

Les bibliothèques suivantes ont comme point commun de créer des schémas, des graphiques ou des interfaces cartographiques interactives c'est-à-dire que l'utilisateur interagit avec en passant sa souris dessus. De plus, elles permettent de mettre à jour les graphiques au fur et à mesure que de nouvelles requêtes sont réalisées sans avoir à recharger la page.

Après plusieurs recherches, la plupart des outils utilisent des bibliothèques JavaScript. Cependant il en existe plusieurs, une sélection est détaillée ci-dessous dans les tableaux 32 à 34.

D3.js

Tableau 32 : Récapitulatif de la bibliothèque D3.js

Avantages	Limitations
<p><u>Open source</u> : D3.js est une librairie JavaScript gratuite permettant de visualiser des données.</p> <p><u>Grande communauté</u> : l'ancienneté de cette solution fait que la documentation est importante.</p> <p><u>Grande flexibilité</u> : il est possible de composer plusieurs visuels avec l'ajout de composants simples.</p>	<p><u>Prise en main difficile</u> : Malgré de nombreux exemples disponibles, son utilisation peut être complexe voire surdimensionnée pour des schémas simples.</p>

Chart.js

Chart.js est une librairie créée en 2013 par Nick Downie. Cette librairie est la seconde plus populaire après D3.js.

Tableau 33 : Récapitulatif de la bibliothèque Chart.js

Avantages	Limitations
<p><u>Open source</u> : Chart.js est une librairie gratuite utilisant des fonctions JavaScript.</p> <p><u>Communauté active</u> : cette solution possède une grande communauté, offrant ainsi de la</p>	<p><u>Bibliothèque limitée</u> : cette solution n'est pas adaptée aux graphiques complexes. En effet il n'existe que 8 types de graphiques : linéaire, à barre, à bulle, nuage de point, de zone, en étoile,</p>

documentation. <u>Adaptée aux débutants</u> : cette bibliothèque possède de nombreux schémas pré-faits, ce qui la rend très accessible.	diagramme circulaire en 2 styles différents. <u>Graphiques limités</u> : cette bibliothèque donne des graphiques peu interactifs, même si l'interactivité n'est pas un point crucial ici, et très dépendants de JavaScript.
--	--

Plotly**Tableau 34 : Récapitulatif de la bibliothèque Plotly**

Avantages	Limitations
<u>Polyvalence</u> : elle permet de faire beaucoup de graphiques et schémas simples. <u>Interactivité</u> : elle permet de rendre les interfaces interactives. <u>Facile à prendre en main</u> : accessible même aux personnes peu habituées à créer des graphiques.	<u>Sous licence</u> : la librairie est sous licence MIT donc payante. <u>Interopérabilité</u> : les visuels dépendent du navigateur. <u>Personnalisation</u> : il est assez difficile de sortir des schémas pré-faits de la librairie donc compliqué de créer des graphiques complexes.

3.2.3.3. Comparaison des outils (définition des critères, poids et valeurs)

D'après l'analyse réalisée ci-dessus, les critères suivants semblent ressortir :

- Flexibilité ;
- Interopérabilité ;
- Facilité d'utilisation ;
- Coût.

L'équipe qui reprendra le projet est normalement formée à utiliser ces outils et a surtout besoin de flexibilité pour atteindre le meilleur rendu possible, ce qui en fait le critère le plus important. Cependant, l'interopérabilité représente quelque chose d'important car tous les utilisateurs doivent y avoir accès de la même façon. Enfin, les coûts ne sont pas vraiment un problème en vue du prix des licences, ce qui fait qu'il s'agit du critère au poids le plus faible.

Tableau 35 : Critères de sélection des différentes bibliothèques d'intégration graphique et leurs notes finales

Tableau de performance bibliothèque					
Solution	C1	C2	C3	C4	Note
	<i>facilité</i>	<i>Flexibilité et</i>	<i>Interopérabilité</i>	<i>Coût</i>	

	<i>d'utilisation</i>	<i>personnalisation</i>			
D3.js	3	5	5	5	4.4
Chart.js	4	2	5	5	3.5
Plotly	4	2	1	1	2.3
Poids critère	0.3	0.4	0.2	0.1	

3.2.3.4. Choix pour la solution

D'après le Tableau , la meilleure solution pour l'intégration des graphiques au produit final est la bibliothèque D3.js, qui fait partie de la méthode AJAX. En effet, elle arrive en première place grâce à sa grande flexibilité, son interopérabilité et le fait qu'il s'agisse d'une solution *open source*.

D3.js est donc la solution préconisée pour le produit final.

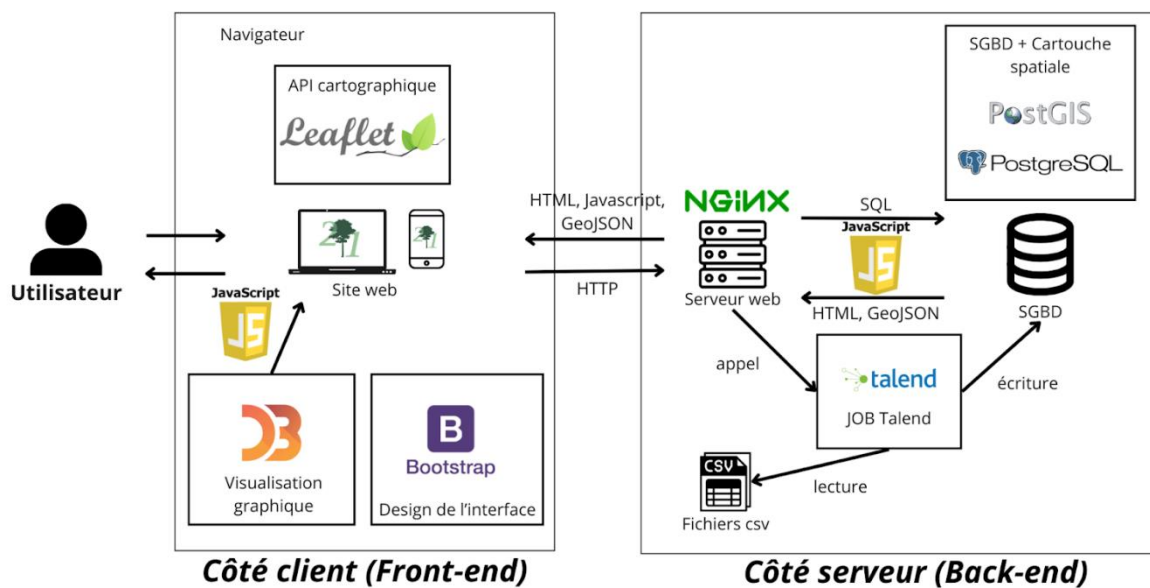
4. Choix d'architecture technique

Les veilles nous ont permis de comparer différents outils permettant la mise en place des parties *front-end* et *back-end*, ainsi que leur articulation. À l'issue de chaque veille, des préconisations d'outils ont été faites. Ces outils devraient permettre de répondre au mieux aux besoins de notre projet et de mettre en place toutes les fonctionnalités prévues.

La solution aura donc la forme d'un site web. Il sera lié au serveur web **NGINX** qui communiquera avec le système de gestion de base de données **PostgreSQL** et sa cartouche spatiale **PostGIS**. Aucun serveur cartographique ne sera nécessaire, ainsi la communication entre serveur web et cartouche spatiale se fera avec un **middleware PHP**. L'API cartographique **Leaflet** permettra la visualisation des cartes sur le navigateur web. La visualisation graphique sera permise par l'utilisation de la bibliothèque **D3.js**. La partie *front-end* pourra être codée à l'aide de **Bootstrap** ou **Tailwind** et la partie *back-end* en utilisant le langage **Javascript**.

L'architecture technique ci-dessous (Figure 6) résume les différents éléments techniques préconisés et schématise l'articulation entre le *back-end*, le *front-end* et l'utilisateur.

Figure 6 : Schéma de l'architecture technique du produit final



5. Choix finaux pour le démonstrateur

Les solutions préconisées pour le produit final peuvent différer des choix réalisés par les étudiants AgroTIC pour le démonstrateur. En effet, des contraintes pédagogiques et de connaissances entraîneront des choix différents. Ces différences seront recensées dans cette partie.

5.1. Fonctionnalités développées

Dans le cadre de la réalisation du démonstrateur, la difficulté de développement (estimée par le temps de développement nécessaire) des fonctionnalités a été estimée au cours d'un processus de vote (*planning poker*) ayant sollicité les étudiants ainsi que l'équipe pédagogique.

5.2. Les données

5.2.1. Les données parcellaires

Dans le démonstrateur, le développement de la fonctionnalité d'importation de parcelles est réalisé. Néanmoins, cette importation se fait à l'aide un fichier "idéal", c'est-à-dire un fichier qui correspond exactement à ce dont la base de données a besoin pour réaliser l'import (comme sur la figure 7).

parcelles-polygon — Total des entités: 3, Filtrées: 3, Sélectionnées: 0

	id	nom	date_plant	comm	essence	its	ru
1	NULL	P1	2012	NULL	0	4	1
2	NULL	P2	2007	NULL	0	4	1
3	NULL	P3	2010	NULL	0	4	1

Figure 7 : Exemple de fichier "idéal" pour l'importation d'un shapefile

Aussi, par soucis de confidentialité, les données parcellaires utilisées dans la base de données ne seront pas des données réelles, mais des données inspirées.

5.2.2. Les données de sorties du modèle

N'ayant pas d'accès direct au serveur des commanditaires, il est nécessaire dans le cadre de ce projet de stocker localement les fichiers csv. Ainsi, seule une partie de ceux-ci sera prise en compte dans le démonstrateur. En effet, si tous les fichiers de variables sont disponibles, ils ne seront pas nécessairement utilisés dans le démonstrateur. De plus, certaines mailles ne seront pas présentes dans les csv : seules des mailles du département des Landes seront disponibles dans le démonstrateur. Ces mailles ont été sélectionnées grâce à l'emprise des parcelles fournies par la Compagnie des Landes. Enfin, seule une espèce sera disponible : le pin maritime. Cette sélection permettra ainsi de limiter la quantité de fichiers csv utilisés, sans restreindre les fonctionnalités du démonstrateur par rapport au produit final.

5.2.3. Les données liées à l'utilisateur

Pour des raisons pratiques, les utilisateurs entrés dans la base de données seront nos *personae*, afin de faciliter la présentation du démonstrateur. Les *personae* inventés lors de l'analyse fonctionnelle représentent l'ensemble des types de profils définis.

Justine PHEVE et Christophe HAIE auront des comptes "Branche". Justine PHEVE intégrera dans son équipe de travail Marc KAKTUS avec son compte "Feuille". Camille CAMOMILLE n'aura, quant à elle, pas de compte, elle accèdera au site *Tree 2 1* avec le profil "Bourgeon".

5.3. SGBD

PostgreSQL a été préconisé pour le produit final. Celui-ci sera aussi utilisé pour le démonstrateur avec la cartouche SIG PostGIS. De plus, les étudiants ont été formés sur ce SGBD. **Le démonstrateur sera donc en cohérence avec le produit final.**

5.4. Environnement de développement *back-end*

5.4.1. Langage de programmation du *back-end*

Le langage de programmation *back-end* préconisé pour le produit final est JavaScript. Cependant, pour des raisons pédagogiques, les langages privilégiés seront **PHP** et **HTML**. **Occasionnellement**, **JavaScript** pourra être utilisé, notamment pour l'interaction cartographique et les tests de cohérence.

Dans ce cas, le démonstrateur sera alors différent du produit final.

5.4.2. Serveur web

Le serveur web préconisé pour le produit final est le serveur NGINX. Cependant, pour des raisons pratiques et de temps, le serveur Apache sera utilisé pour le démonstrateur. En effet, NGINX requiert du temps de prise en main que les étudiants ne disposent pas. De plus, le développement sera fait en local avec l'utilisation de WAMP ou MAMP, qui utilisent Apache, afin de pouvoir visualiser les pages codées en langage PHP.

De ce fait, **Apache sera le serveur web utilisé pour le démonstrateur.**

5.5. Environnement de développement *front-end*

5.5.1. Langage et technologies de développement

Pour le produit final, les bibliothèques CSS Bootstrap ou encore Tailwind CSS ont été préconisées pour leur possibilité de personnalisation de l'interface utilisateur et pour la flexibilité des langages utilisés.

Pour les étudiants AgroTIC, le temps est limité pour réaliser le projet. De ce fait, il s'agit d'une contrainte limitante et l'outil doit être facilement pris en main par l'ensemble des étudiants qui devront l'utiliser dans le cadre du projet. De plus, le responsable web de l'école Bordeaux Sciences Agro est formé à Bootstrap et pourra répondre aux questions et problématiques des étudiants.

Bootstrap sera donc utilisé pour le démonstrateur.

5.5.2. Cartographie interactive

Dans le produit final, l'outil de cartographie préconisé est **Leaflet** ou encore **Google Maps Javascript API** en vue de la proximité des notes attribuées aux deux outils.

Dans le cadre du démonstrateur, l'outil utilisé sera **Leaflet**, pour sa simplicité de mise en place, et parce que les étudiants AgroTIC auront déjà étudié cet outil au cours de leur cursus.

5.5.3. Visualisation des données graphiques

Dans ce cas, **le démonstrateur utilisera la même méthode que pour le produit final, à savoir AJAX**. En effet, l'actualisation des graphiques sans recharger la page en fait un outil idéal pour le projet.

Cependant, en ce qui concerne la bibliothèque utilisée, la bibliothèque **Highcharts** sera utilisée à la place de **D3.js**, du fait que les étudiants l'aient utilisé durant leur formation.

5.6. Schéma de l'architecture technique du démonstrateur

L'architecture technique ci-dessous (Figure 8) résume les différents éléments techniques du démonstrateur et schématise l'articulation entre le *back-end*, le *front-end* et l'utilisateur.

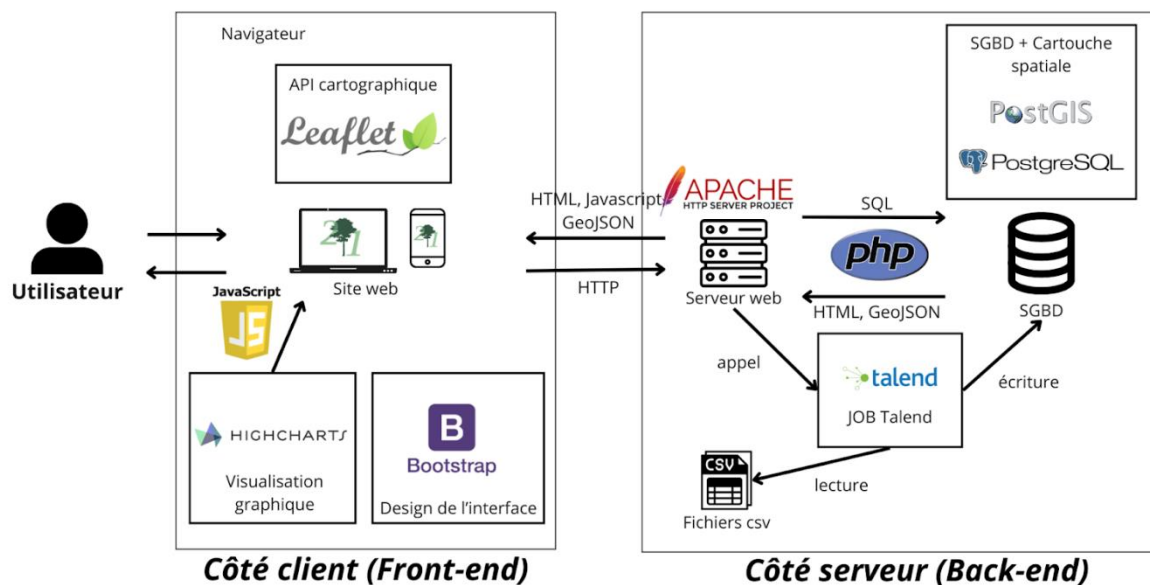


Figure 8 : Schéma de l'architecture technique du démonstrateur

6. Bibliographie

10 frameworks CSS à utiliser en 2024—Codeur Blog. (2022, octobre 18). Codeur.com.

<https://www.codeur.com/blog/frameworks-css/>

12 Avantages et inconvénients du HTML. (2023, septembre 12).

<https://barrazacarlos.com/fr/avantages-et-inconvenients-de-lhtml/>

Bootstrap (front-end framework). (2024). In Wikipedia.

[https://en.wikipedia.org/w/index.php?title=Bootstrap_\(front-end_framework\)&oldid=1198156648](https://en.wikipedia.org/w/index.php?title=Bootstrap_(front-end_framework)&oldid=1198156648)

Chahine, H. (2020a, novembre 19). ReactJS vs Angular vs VueJS : Que choisir en 2024 ? Ambient IT.

<https://www.ambient-it.net/reactjs-vs-angular-vs-vuejs/>

Chahine, H. (2020b, novembre 19). ReactJS vs Angular vs VueJS : Que choisir en 2024 ? Ambient IT.

<https://www.ambient-it.net/reactjs-vs-angular-vs-vuejs/>

Coding : Les avantages et les inconvénients de JavaScript. (2021, mars 23). Geekway Mag.

<https://geekway-mag.fr/coding-les-avantages-et-les-inconvenients-de-javascript/coding/>

Coner, M. (s. d.). Tailwind CSS vs. Bootstrap : Which is better? Consulté 1 février 2024, à l'adresse

<https://prismic.io/blog/tailwind-vs-bootstrap>

Démarrer avec React—Apprendre le développement web | MDN. (2023, août 3).

https://developer.mozilla.org/fr/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started

Deux années avec Tailwind CSS, quel bilan? - Alsacreations. (2022, décembre 20).

<https://www.alsacreations.com/article/lire/1884-deux-annees-avec-tailwind-quel-bilan.html>

Feuilles de style en cascade. (2023). In Wikipédia.

https://fr.wikipedia.org/w/index.php?title=Feuilles_de_style_en_cascade&oldid=210382977

Gupta, A. (2023, juin 21). 24 Best CSS Frameworks To Look Forward In 2023. LambdaTest.

<https://www.lambdatest.com/blog/best-css-frameworks/>

Maslov, Y. (2023, juin 7). When and Why Use React : Pros, Cons, Use Cases | DigitalSuits.

<https://digitalsuits.co/blog/when-and-why-use-react-pros-cons-use-cases/>

Pourquoi utiliser Angular ? | Angular. (s. d.). Consulté 1 février 2024, à l'adresse <https://angular.fr/>

Quels sont les meilleurs frameworks CSS en front-end ? (2022, juin 8). Mobiskill.

<https://mobiskill.fr/blog/conseils-emploi-tech/quels-sont-les-meilleurs-frameworks-css-en-front-end/>

React-responsive. (2022, novembre 29). Npm. <https://www.npmjs.com/package/react-responsive>

State of CSS 2022 : CSS Frameworks. (s. d.). Consulté 1 février 2024, à l'adresse

<https://2022.stateofcss.com/en-US/css-frameworks/>

State of JavaScript 2022 : Front-end Frameworks. (s. d.). Consulté 1 février 2024, à l'adresse

<https://2022.stateofjs.com/en-US/libraries/front-end-frameworks/>

Tailwind CSS. (2024). In Wikipedia.

https://en.wikipedia.org/w/index.php?title=Tailwind_CSS&oldid=1196094850

Veille SGBD

¹ Method of calculating the scores of the DB-Engines Ranking, par solidIT consulting & software development gmbh, https://db-engines.com/en/ranking_definition.

² System Properties Comparison Microsoft SQL Server vs. MySQL vs. Oracle vs. PostgreSQL, par solidIT consulting & software development gmbh, <https://db-engines.com/en/system/Microsoft+SQL+Server%3BMySQL%3BOracle%3BPostgreSQL>

³ Comparaison des systèmes de gestion de bases de données: MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch et autres, par Hello Pomelo, 17 mars 2021, <https://hello-pomelo.com/articles/quel-systeme-de-gestion-de-bases-de-donnees-choisir>

⁴ Alexei Balaganski, Database and Big Data Security, 11 Avril 2023 KuppingerCole Leadership Compass, <https://www.oracle.com/a/ocom/docs/corporate/analystrelations/kuppingercole-leadership-compass-data-security-platforms.pdf>

⁵ Comparatif des 5 plus grands systèmes de gestion de base de données, par Veevo solutions, <https://www.veevo-solutions.fr/blog/comparatif-des-5-plus-grands-systemes-de-gestion-de-base-de-donnees>

⁶ TOP 10 DES MEILLEURES BASES DE DONNÉES À CHOISIR EN 2024, par Ambient IT, Mis à jour le 29/12/2023, <https://www.ambient-it.net/meilleures-base-de-donnees/>

⁷COMPARATIF, PRIX DES LICENCES MICROSOFT SQL SERVER, ORACLE, IBM DB2 UDB, par SQLPro, https://blog.developpez.com/sqlpro/p10447/ms-sql-server/comparatif_prix_des_licences_microsoft_s