**Projet :**

Conception d’un POC pour le développement d’une application de gestion immobilière

**Client :**

Accord entre plusieurs agences immobilières

**Nom du Site :**

A définir

**Logo du Site :**

A définir

Sommaire

Table des matières

[1. Cadre du projet 3](#_Toc128951558)

[1.1. présentation de la société 3](#_Toc128951559)

[1.1.1. Commanditaire 3](#_Toc128951560)

[1.1.2. Prestataire 3](#_Toc128951561)

[1.2. présentation du projet 3](#_Toc128951562)

[1.3. Public cible 3](#_Toc128951563)

[1.4. CONCURRENCE-BENCHMARK 4](#_Toc128951564)

[1.5. conception graphique et ergonomique 6](#_Toc128951565)

[1.6. budget 7](#_Toc128951566)

[1.7. délai 7](#_Toc128951567)

[1.8. exigences fonctionnelles 7](#_Toc128951568)

[2. Spécification fonctionnelle 8](#_Toc128951569)

[2.1. Diagramme fonctionnelle 8](#_Toc128951570)

[2.2. Représentation MCD 10](#_Toc128951571)

[2.3. Représentation UML 10](#_Toc128951572)

[3. Spécifications techniques 11](#_Toc128951573)

[3.1. Test de l’application 11](#_Toc128951574)

[3.1.1. Test Driven Development 11](#_Toc128951575)

[3.1.2. Test de la fonctionnalité 13](#_Toc128951576)

[3.1.3. Test de l'utilisabilité 14](#_Toc128951577)

[3.1.4. Test de l'interface 15](#_Toc128951578)

[3.1.5. Test de compatibilité 15](#_Toc128951579)

[3.1.6. Test des performances 16](#_Toc128951580)

[3.1.7. Test de SECURITE 16](#_Toc128951581)

[3.2. Choix technologiques 17](#_Toc128951582)

[3.2.1. Versionning 17](#_Toc128951583)

[3.2.2. GitHub 18](#_Toc128951584)

[3.2.3. Base de données 18](#_Toc128951585)

[3.2.4. Visual Studio 19](#_Toc128951586)

[3.2.5. Domaine et hébergement 19](#_Toc128951587)

[4. Conclusion : 20](#_Toc128951588)

# Cadre du projet

## présentation de la société

## Commanditaire

La présente demande a été réalisée suite à un appel d’offre réalisé par un accord entre plusieurs agence immobilière.

## Prestataire

La société Infeco est une entreprise qui allie informatique et écologie, elle possède plusieurs équipes dont une s’appelant Immo, qui sera en charge de ce projet.

Pour la réalisation de ce POC, l’équipe Immo sera constitué comme suit :

## présentation du projet

Le projet consistera à réaliser le développement d’un outil de gestion de location immobilière. Ce développement sera réalisé sous forme de POC (PREUVE DE CONCEPT).

## Public cible

L’application sera essentiellement destinée aux personnes de l’agence. Étant donné que plusieurs gestionnaires peuvent intervenir, une connexion par nom d’utilisateur est conseillée.

## CONCURRENCE-BENCHMARK

De nombreuses applications similaires existent, en comparant ces différents concurrents, on peut noter les points positifs et négatifs, afin de proposer des améliorations

La réalisation d’un benchmark permet ainsi :

* De définir les critères qui favorisent la performance et la réussite
* D’identifier les meilleures entreprises dans votre domaine d’activité
* D’avoir une vision claire du marché et de ses tendances
* De maîtriser l’univers concurrentiel
* De connaître le positionnement, les points forts et les points faibles de chacun de vos concurrents
* D’effectuer une synthèse des bonnes pratiques
* De mettre en place un plan d’action au niveau de l’entreprise, pour s’approprier ces Best practices et optimiser la performance de l’entreprise (ou réussir son lancement produit).

***Numéro 1 :***



Source : <https://www.emjysoft.com/logiciel-gestion-locative>

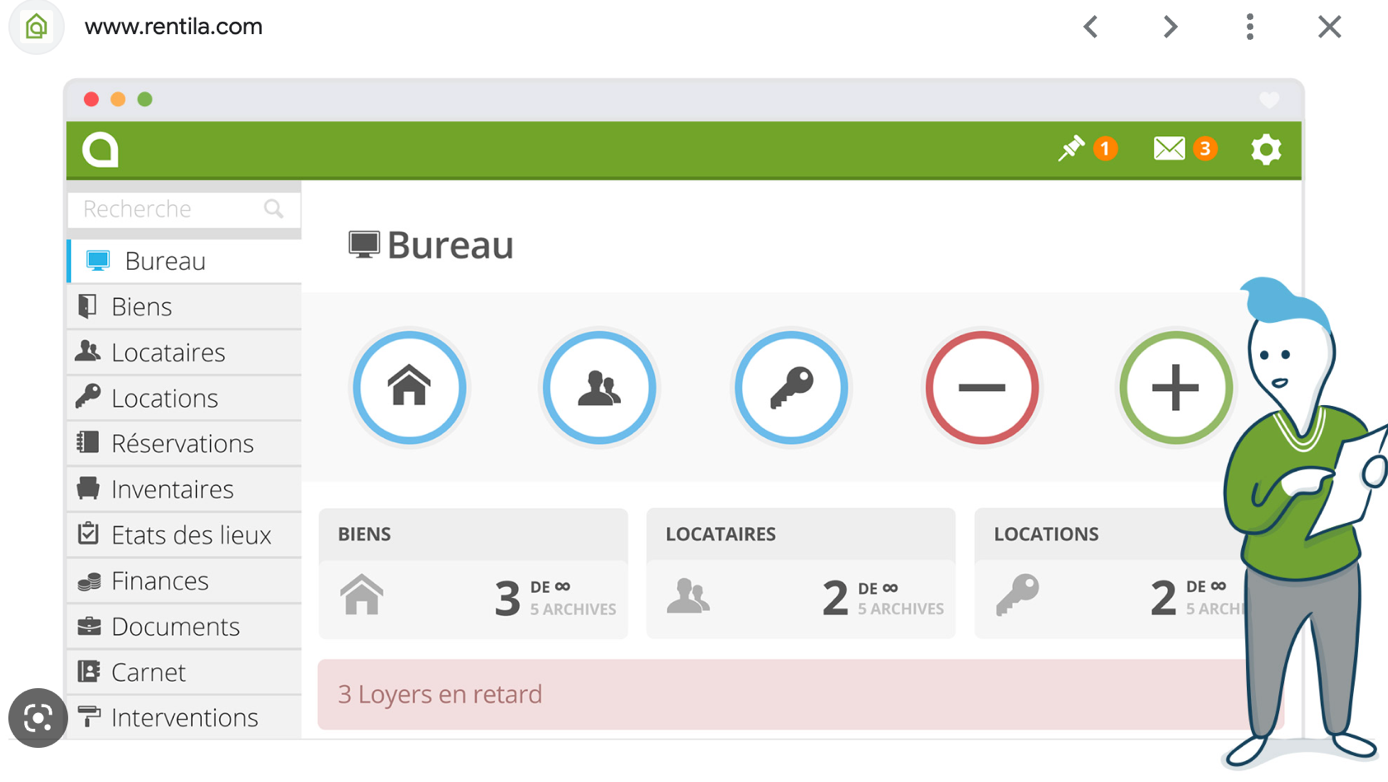
Points positifs :

* Écran d’accueil simple et intuitif
* Nombreuses fonctionnalités

Points négatifs :

* Application certainement non responsive

***Numéro 2 :***



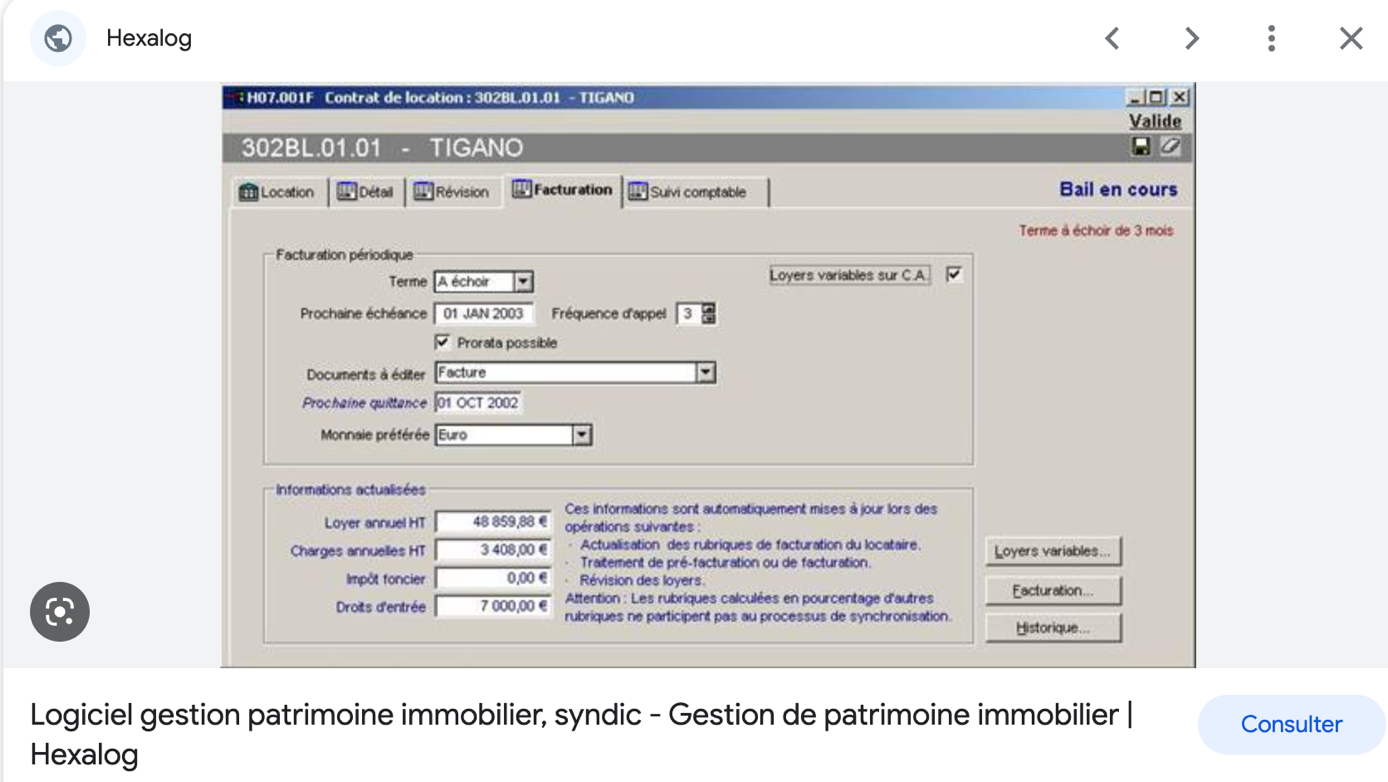
Source : <https://www.rentila.com/>

Points positifs :

* Écran d’accueil simple et intuitif
* Nombreuses fonctionnalités
* Menu en liste permettant une vision rapide

Points négatifs :

***Numéro 3 :***



Points positifs :

Points négatifs :

* Interface simple et non intuitive

***Conclusion :***

Afin de pouvoir se démarquer, une application simple et intuitive sera privilégiée . Un menu en liste semble le mieux adapter pour une application responsive. Pour une application bureau, un menu en onglet donnerait une application conviviale et intuitive.

Un choix au niveau d’une application responsive devra être fait.

De nombreuses fonctionnalités peuvent être réalisé afin de créer une application complète.

## conception graphique et ergonomique

Le graphisme du site se doit de retranscrire l’image, l’identité de l’agence immobilière.

La page d’accueil, contiendra le logo de l’agence et une image de fond représentant l’entreprise ou le personnel de l’agence.

La charte graphique devra être finalisé entre le designer et l’agence immobilière.

Le site respectera les [80 bonnes pratiques SEO du référentiel OPQUAST.](https://checklists.opquast.com/seo/)

## budget

A déterminer

## délai

Un planning prévisionnel sera mis en place afin de déterminer les différents jalons et durée de chaque étape.

Il sera mis à jour régulièrement lors de réunion régulière afin de valider et modifier les différentes étapes

Ce planning sera géré par le responsable du projet en utilisant Trello.



## exigences fonctionnelles

Listes des différentes fonctions souhaitées :

* Stocker les coordonnées des locataires
* Le locataire peut louer 1 ou plusieurs appartements
* Un dépôt de garantie doit être versé à chaque nouveau bail
* Un appartement a
  + Une adresse
  + Un complément d’adresse
  + Une ville
  + Un code postal
  + Un prix de charges
  + Un prix de loyer
  + Un prix de dépôt de garantie
* Tous les mois le locataire doit payer un loyer
  + Il faut conserver la date de paiement
  + Le paiement émit peut provenir soit du locataire et/ou de la CAF
* Un état des lieux est obligatoire lors de l’entrée ou de la sortie de l’appartement
* L’état des lieux doit mentionner la date d’entrée, de sortie ainsi que les remarques observées.
* Il faut connaitre le solde du locataire à tout moment
* L’agence prélèvent 8% sur les loyers
* Génération des quittances à destination des locataires, seulement si le locataire est en règle :
  + Mail
  + PDF
* Bilan lorsque le locataire quitte le logement

Listes des fonctions pouvant être ajoutés :

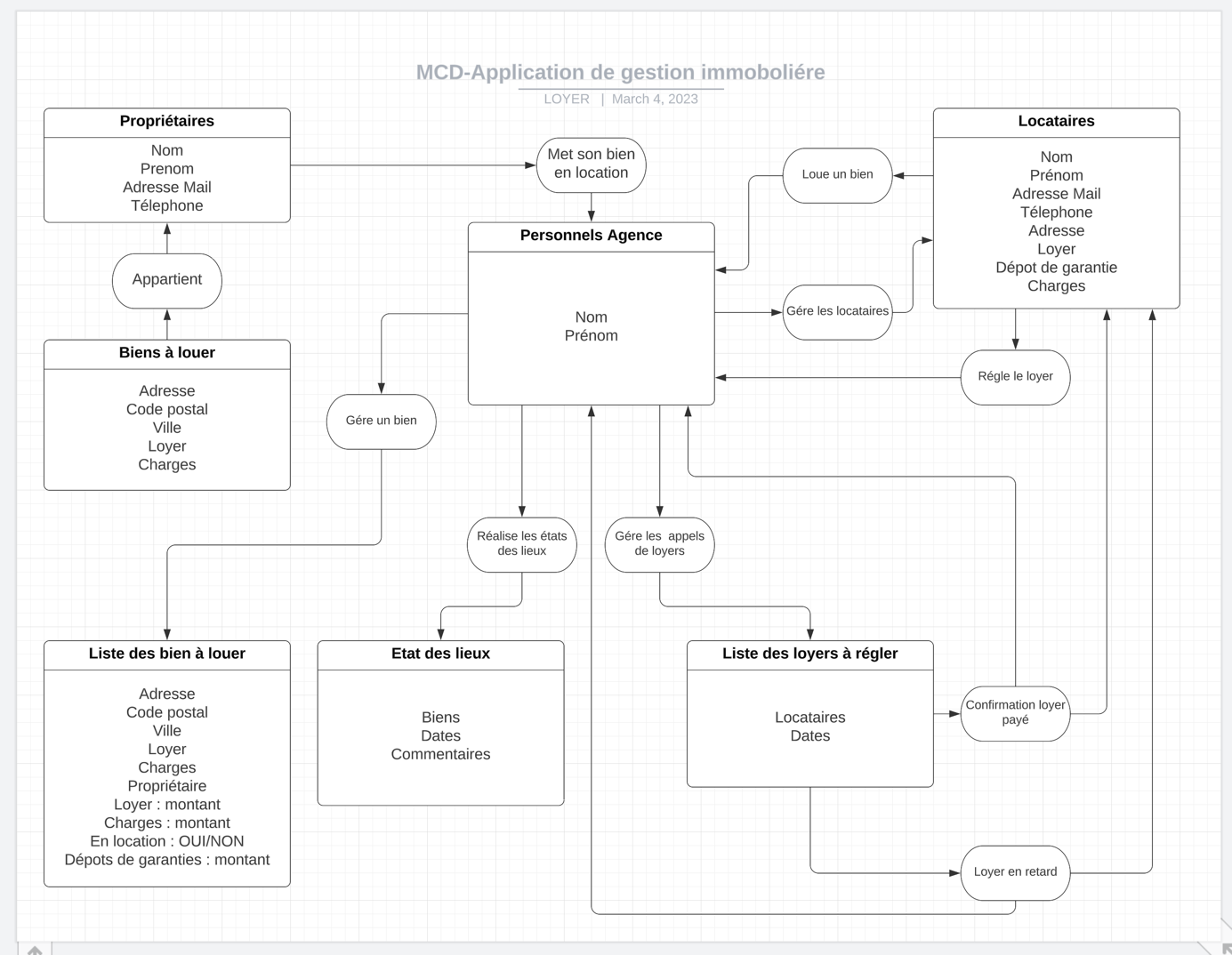
* Stocker les coordonnées des propriétaires
* Bilan des loyers perçus de locataires
* Bilan des prélèvements de l’agence

# Spécification fonctionnelle

## Diagramme fonctionnelle

## Représentation MCD

Le schéma conceptuel de données, également appelé MCD pour modèle conceptuel de données, **est une représentation claire des données du système d'information à concevoir**. Cette représentation en outre figure les relations entre ces données.



## Représentation UML

# Spécifications techniques

## Test de l’application

## Test Driven Development

Lors du développement de l’application nous utiliserons le Test Driven Development.

Plus concrètement le Test Driven Development est composé de plusieurs étapes :

* Décrire un problème par l’écriture d’un test.
* Exécuter le test et s’assurer qu’il échoue ; vu que le code à tester n’existe pas encore.
* Coder juste ce qu’il faut pour assurer la réussite du test
* Vérifier la réussite du test en question avec l’exécution d’autres tests.
* Remanier (refactor) le code ; améliorer sa qualité sans modifier le comportement, en d’autres termes retravailler le code source en conservant les mêmes fonctionnalités.

Le Test Driven Development se déroule par cycle de trois phases : red (rouge), green (vert) et refactor (remaniement).

**TDD Phase Red :**

En amont, le développeur doit commencer par écrire un test unitaire. Celui-ci va obligatoirement échouer puisque le code à tester n’a pas encore été créé. Ce test unitaire est donc symbolisé par la couleur rouge.

Dans cette phase, vous vous mettez à la place de l’utilisateur. Celui-ci veut pouvoir utiliser simplement le code. Vous écrivez donc un test, qui contient des composants qui n’ont pas encore été implémentés. Vous choisissez les éléments indispensables pour que le code soit fonctionnel.

**TDD Phase Green :**

Dans un second temps, le développeur écrit le code suffisant pour que le test unitaire réussisse et passe au vert.

Dans cette phase, le développeur essaie de trouver une solution simple en n’écrivant que le strict nécessaire puis l’intègre dans le code de production afin que le test soit marqué en vert.

**TTD Phase refactor :**

Enfin, l’étape du refactoring, elle, consiste comme son nom l’indique à remanier le code en contrôlant que tous les tests réalisés restent bien en vert.

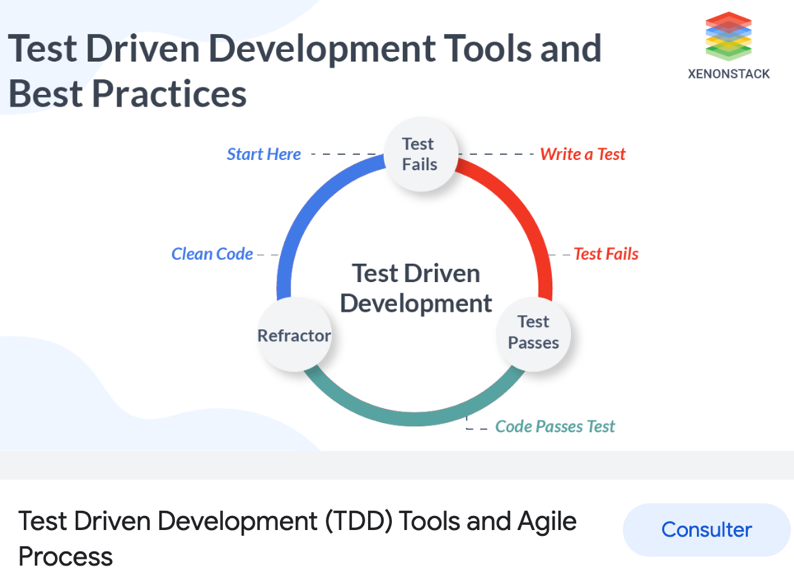
Dans cette étape, vous n’avez plus qu’à structurer et éventuellement compléter le code de production pour qu’il soit lisible et propre en supprimant la duplication des codes par exemple.

 Quels sont les bénéfices du Test Driven Development ?

* Le TDD permet de s’assurer que la conception d’un logiciel est bien pensée avant de commencer à écrire le code.
* Écrire les tests en amont du code permettent une couverture de test unitaire élevé mais aussi une pertinence de ces tests, dans la mesure où ils répondent parfaitement aux besoins métiers. On évite ainsi les pertes de temps à réaliser des fonctionnalités dont le résultat ne sera pas satisfaisant !
* Avec pour objectif de détecter les bugs le plus en amont possible, le test-driven développent, minimise les erreurs et autres problèmes une fois le déploiement réalisé et notamment les régressions. La maintenance s’en trouve facilitée avec, à la clé, des gains de temps potentiellement substantiels en production et une meilleure qualité des développements
* La suppression des redondances dans le code permet d’éviter également la suringénierie

Quels sont les inconvénients du Test Driven Development ?

* Nécessite la compréhension du code et exige peut-être une période de formation plus longue.
* Teste uniquement l’exactitude du code et non la facilité d’utilisation du logiciel.
* Doit éventuellement être complété par d’autres procédures de test.



## Test de la fonctionnalité

(Lien : <https://comparium.app/fr/blog/web-test-tools/website-testing-checklist/#functionality>)

La partie la plus importante d'une liste de contrôle de test de site Web est la fonctionnalité de ciblage. Bien sûr, la conception et l'affichage sont également importants. Cependant, il est vital de s'assurer que chaque lien, formulaire et base de données fonctionne correctement, sans erreur. Pour arriver à ce point, vous devez ajouter des exigences fonctionnelles telles que :

* Cliquer sur les boutons et les liens pour vérifier si la bonne page s'ouvre
* Vérifier qu'aucun lien ne mène à une page vide
* Vérifier à nouveau qu'aucune entrée invalide n'est acceptée
* Passer en revue les champs obligatoires et facultatifs pour assurer une fonctionnalité appropriée
* Longueur et limites des caractères
* Test des boutons radio
* Définition des valeurs par défaut lors du chargement / rechargement de la page
* Vérifier qu'une barre de défilement est présente en cas de besoin

## Test de l'utilisabilité

Qu'il s'agisse de cas de test pour une application Web dans le cadre de tests manuels ou de processus automatisés, la convivialité est la prochaine chose importante à vérifier. Cela peut être fait sur des publics ciblés ou des utilisateurs généraux et cela confirme que le site offre une interface conviviale, que les instructions fournies sont claires et que la navigation est intuitive. Les tests d'utilisabilité comprennent également l'examen du contenu pour identifier les fautes d'orthographe ou de grammaire, ainsi que la vérification des polices et des couleurs. Voici un exemple de liste de contrôle de test d'utilisabilité Web:

* Vérifier le contenu du site Web et la navigation générale
* Les liens, boutons et menus vers d'autres pages doivent être facilement accessibles et cohérents dans tous les cas
* Tout le contenu doit être lisible, exempt d'orthographe ou de grammaire
* Toutes les images du site doivent avoir une description "texte alternatif"
* L'application Web offre-t-elle une idée claire de son public cible?
* L'application offre-t-elle une interface et une interaction reconnaissables et cohérentes?
* Cohérence de la police et du style dans toute l'application
* Alignement du texte sur toutes les pages
* Standardisation du format et de la taille des boutons et des champs dans l'application Web
* Champs grisés qui ne sont pas obligatoires ou désactivés
* La présence du titre sur toutes les pages
* Les boutons radio, listes déroulantes, zones de texte et autres champs doivent également être accessibles à partir du clavier
* Vérification de tout ce qui précède dans différentes résolutions d'écran sur le même navigateur Internet

## Test de l'interface

L'interface est la suivante pour une liste de contrôle de test d'application Web appropriée. En commençant par l'interface côté serveur, il est également essentiel de tester la communication et la compatibilité entre le serveur, les logiciels, le matériel, la base de données et le réseau.

**Concentrez-vous sur les éléments suivants**

* Serveur d'applications et interface de serveur Web
* Interface de base de données et de serveur d'applications

Si vous vous demandez comment écrire des cas de test pour une application Web avec un exemple d'efficacité prouvée dans la vérification d'interface, nous avons la réponse pour vous. Commencez toujours par vérifier les interactions entre la manière dont le serveur d'applications et le serveur Web / serveur de base de données et le serveur d'applications sont gérés. Si, au cours d'une interrogation, la base de données ou le serveur Web délivre un message d'erreur, ce message doit être correctement affiché aux utilisateurs via le serveur d'applications.

Une fois la fonctionnalité vérifiée de toutes les communications avec le serveur, il est temps de passer à la partie visuelle du processus de test - Compatibilité.

## Test de compatibilité

Même si cela se trouve plus bas dans la liste de notre exemple de cas de test pour la vérification des applications Web, la compatibilité est également essentielle pour que l'ensemble de l'application fonctionne correctement. Cela garantit que, quel que soit le navigateur ou le système d'exploitation des utilisateurs sur leurs appareils, l'application Web ressemble et fonctionne correctement. Il contient plusieurs étapes :

* Compatibilité sur plusieurs navigateurs
* Compatibilité sur divers systèmes d'exploitation
* Compatibilité de la navigation mobile

Compatibilité du navigateur

La mise en place d'un script de test de site Web qui traite de la compatibilité du navigateur n'est pas la procédure la plus simple. Cette section peut être la partie la plus influente de l'ensemble du processus de test. Étant donné que certaines applications dépendent trop des navigateurs, différentes configurations peuvent renvoyer des résultats incohérents.

## Test des performances

Lorsque nous parlons de performances dans les tests d'applications Web, nous faisons référence aux temps de chargement, aux tests de résistance et à d'autres fonctionnalités qui font référence à la façon dont votre application interagit avec divers navigateurs et systèmes d'exploitation de ce point de vue. Il est préférable de créer plusieurs exemples de cas de test pour tester les performances des applications Web, en utilisant différentes vitesses Internet.

Le test de charge consiste essentiellement à tester ce qui se passe lorsque plusieurs utilisateurs accèdent à la même page. Idéalement, vos applications Web n'auront aucun problème à gérer plusieurs demandes en même temps. La conversation est beaucoup plus large que la façon dont votre application réagit à plusieurs visites, car vous devez prendre en compte le service d'hébergement et de nombreux autres éléments. Cependant, vous assurer que tout fonctionne correctement de votre côté facilitera grandement le dépannage.

Tout comme pour la compatibilité des navigateurs, les tests de résistance doivent être effectués sur différents systèmes d'exploitation, résolutions d'écran et versions de navigateur. Les tests de performance vous donneront également une idée claire de l'évolutivité de votre site Web. Il s'agit généralement d'un équilibre entre l'équilibrage de la charge, le temps de réponse entre les demandes des utilisateurs et la réponse du serveur, et la charge qu'une application Web peut supporter avant de tomber en panne

## Test de SECURITE

Enfin, nous examinons le processus de test de sécurité avant de lancer l'application de site Web. La cybersécurité est vitale pour toute entreprise qui souhaite avoir une bonne réputation à l'heure actuelle. La protection de vos données, et surtout des données des utilisateurs, est cruciale. Les tests de sécurité augmenteront également les tractions et vous permettront de mieux dormir la nuit en sachant que le risque de brèches est très faible. Une liste de contrôle de site Web de sécurité à toute épreuve devrait inclure tous les éléments suivants :

* Vérifier que toutes les données importantes telles que les informations financières, les mots de passe et les réponses aux questions de sécurité sont soumises via HTTPS et sont accompagnées d'un cryptage SSL
* Vérifiez que les informations financières et les mots de passe sont affichés via un format crypté
* Vérifiez que toutes les pages qui acceptent les données des utilisateurs exécutent l'injection SQL pour accéder à la base de données
* Vérifiez les messages d'erreur pour vous assurer qu'ils n'affichent pas d'autres données internes
* Test to see what happens when the session expires or the user logs out - there should be no possibility to continue without logging in again
* Votre [liste de contrôle des tests de sécurité des applications](https://comparium.app/fr/blog/web-test-tools/web-application-security-testing-checklist/) doit également contenir la vérification si les pages non sécurisées et sécurisées sont accessibles sans connexion - en fonction de la configuration de la fonctionnalité pour cette fonctionnalité. Vous devez également exécuter cette vérification sur plusieurs navigateurs
* Assurez-vous que "Afficher le code source" n'est pas activé pour tous les utilisateurs
* Les cookies doivent toujours être stockés sous forme cryptée

<https://www.cnil.fr/fr/reglement-europeen-protection-donnees>

## Choix technologiques

## Versionning

Afin de suivre l’évolution du programme, nous mettrons en place un versionning sémantique, une méthode très utilisée pour l’attribution des numéros de version

Les modifications apportées au logiciel sont communément rassemblées et désignées par **des numéros de version sous la forme “x.y.z”.** On formule ainsi des degrés d’importance de ces changements de gauche à droite, du plus significatif au moindre correctif :

**X – Majeur :** changements non rétro-compatibles, suppression d’une fonctionnalité obsolète, modification d’interfaces, renommages…

**Y – Mineur :** changements rétro-compatibles, introduction de nouvelles fonctionnalités, fonctionnalité marquée comme obsolète…

**Z – Correctif :** corrections d’anomalies rétro-compatibles, modification/correction d’un comportement interne, failles de sécurité…

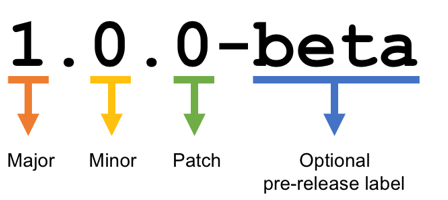


Illustration 1 : Méthode de nomenclature du versioning sémantique

## GitHub

Afin de pouvoir travailler à plusieurs source projet, nous utiliserons GitHub.

GitHub est un service d’hébergement et de développement de logiciels. Il est Actuellement le service d’hébergement de code le plus populaire parmi les développeurs et les programmeurs open source. Il utilise le système de contrôle en version Git, d’où le nom.

## Base de données

Pour le développement des cette application nous utiliserons **PostgreSQL** qui est un système de gestion de base de données relationnelle et objet (SGBDRO).

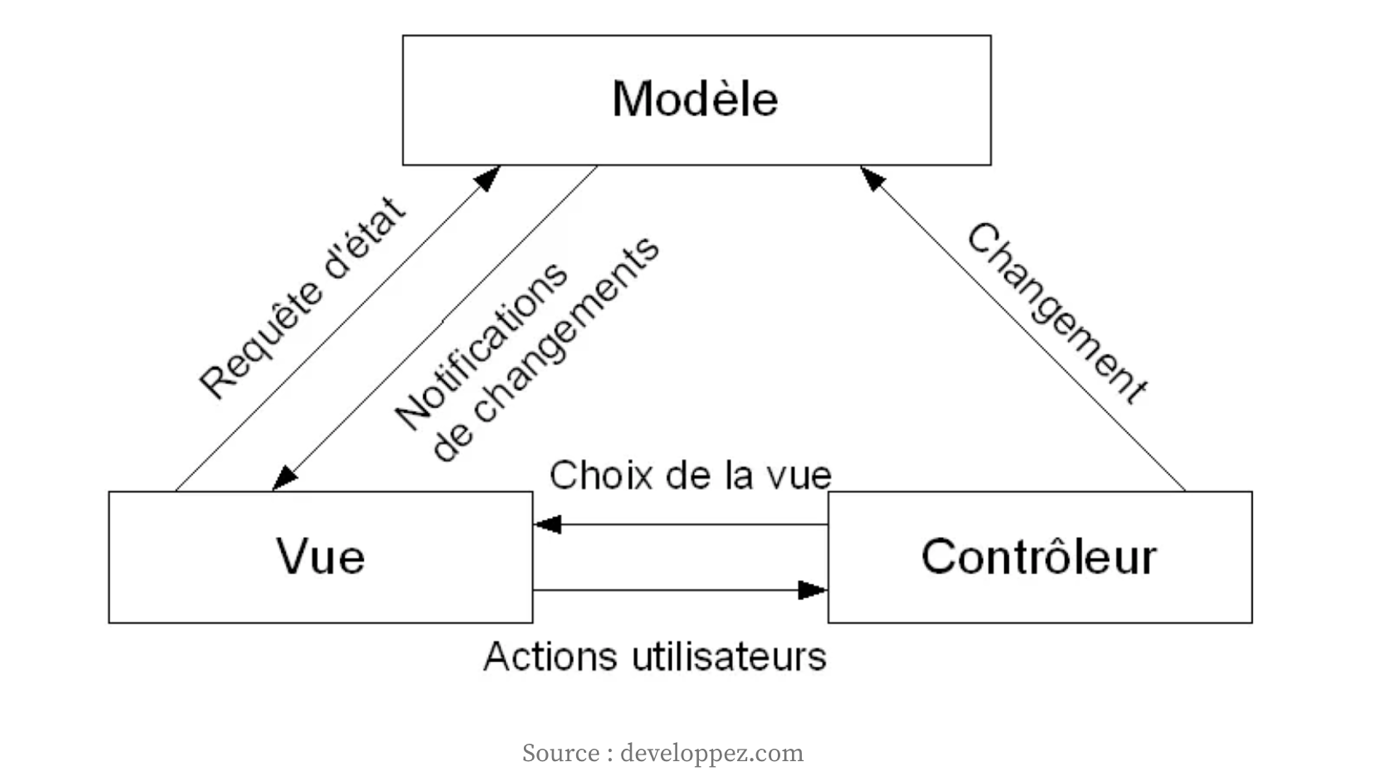
En [informatique](https://fr.wikipedia.org/wiki/Informatique), une base de données relationnelle est une [base de données](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es) où l'information est organisée dans des tableaux à deux dimensions appelés des relations ou tables[1](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es_relationnelle#cite_note-1), selon le modèle introduit par [Edgar F. Codd](https://fr.wikipedia.org/wiki/Edgar_F._Codd) en 1960. Selon ce [modèle relationnel](https://fr.wikipedia.org/wiki/Mod%C3%A8le_relationnel), une base de données consiste en une ou plusieurs relations. Les lignes de ces relations sont appelées des [nuplets](https://fr.wikipedia.org/wiki/N-uplet) ou enregistrements. Les colonnes sont appelées des attributs.

## Visual Studio

L’application sera réalisée en utilisant le langage c#, en utilisant le design pattern MVC, **ce modèle nous apportera :**

* Une meilleure organisation du code.
* La diminution de la complexité lors de la conception
* Une conception claire et efficace grâce à la séparation des données de la vue et du contrôleur.
* La possibilité de réutilisé le code dans d’autres applications.
* Un gain de temps de maintenance et d’évolution du site.
* Une plus grande souplesse pour organiser le développement du site entre différents développeurs.
* Plus de facilité pour les tests unitaires.

Nous utiliserons ce modèle avec l’apport de bibliothèque externe ( Bootstrap



## Domaine et hébergement

# Conclusion :