Table des matières

[Remerciements 3](#_Toc30268154)

[Introduction 3](#_Toc30268155)

[I Contexte du projet 3](#_Toc30268156)

[1) Diagramme de Gantt 3](#_Toc30268157)

[2) Activité du client 3](#_Toc30268158)

[3) Analyse du besoin 3](#_Toc30268159)

[II Réalisation et conception 4](#_Toc30268160)

[1) Choix technologiques 4](#_Toc30268161)

[2) Description du projet 4](#_Toc30268162)

[a) Côté backend 4](#_Toc30268163)

[b) Côté frontend 4](#_Toc30268164)

[III Résultats et perspectives 4](#_Toc30268165)

[1) Utilisation de la solution 4](#_Toc30268166)

[a) Gestion des locataires 4](#_Toc30268167)

[b) Gestion des locaux 4](#_Toc30268168)

[c) Gestion des contrats 4](#_Toc30268169)

[d) Gestion des documents 4](#_Toc30268170)

[2) Perspectives d’amélioration 4](#_Toc30268171)

[Conclusion 4](#_Toc30268172)

[Références bibliographiques 4](#_Toc30268173)

[Glossaire 4](#_Toc30268174)

# Remerciements

Nous tenons à remercier tout particulièrement Mme Kang pour l’aide qu’elle a pu nous fournir lors de nos rencontres de projet. Elle a pu nous guider et nous orienter vers les bonnes technologies à utiliser.

# Introduction

# I Contexte du projet

## Diagramme de Gantt

## Activité du client

Le client est un professionnel qui gère un parc immobilier. Il s’occupe de la location d’appartements. Jusqu’à présent, il gère tout cela à la main avec des papiers. Cependant, il perd un temps précieux à remplir à chaque fois des documents qui pourraient se remplir automatiquement. Il faut aussi qu’il envoit des mails avec des documents remplis à faire signer par le locataire ainsi que des documents à faire remplir.

## Analyse du besoin

Dans un souci de gain de temps et de pénibilité, le client aimerait avoir un logiciel capable de gérer des locataires, des locaux et des contrats de location.

Ce logiciel doit donc capable de créer des locataires à partir d’un formulaire, de les modifier, de les afficher et de leur assigner un contrat.

De plus, le logiciel soit aussi gérer les locaux, pouvoir en ajouter et en supprimer au besoin.

Le logiciel gère aussi les contrats de bail, c’est-à-dire leur création en assignant un local et un locataire. Une fonction principale du logiciel est de pouvoir remplir automatiquement des documents templates à partir des informations renseignées ainsi que d’envoyer des emails automatiquement aux locataires en joignant des documents.

Enfin, le logiciel doit être rapide à prendre en main, facile d’utilisation, et, une demande explicite du client, le logiciel doit être très agréable à regarder.

# II Réalisation et conception

## Choix technologiques

Durant le projet, nous avons dû faire des choix technologiques afin de mener à bien notre projet.

Tout d’abord, nous étions partis sur une solution desktop, c’est-à-dire un logiciel programmé en C++ ou Java. Mais une des contraintes importantes du projets était qu’il faille que l’apparence du logiciel soit jolie. En utilisant ces technologie-là, il aurait été très compliqué et long de faire quelque chose de beau. Ainsi, nous nous sommes orientés vers les technologies web. En effet, celles-ci sont plus faciles à implémenter, plus facile à maintenir, et il est très simple de faire quelque chose de visuellement jolie.

Au niveau du backend, côté serveur, nous utilisons le langage PHP avec le Framework Symfony. Symfony nous évite ainsi de passer du temps sur des parties ennuyantes du logiciel, pour nous concentrer sur la partie métier.

Au niveau du frontEnd, côté client, nous utilisons les langages Javascript, CSS et HTML afin d’avoir un rendu agréable à regarder pour l’utilisateur.

Entre le front et le back, nous passons les paramètres et éléments dans des fichiers Twig qui sont ensuite transformés en fichier html interprétables par les navigateurs actuels.

Au niveau des données, nous les stockons dans une base de données, afin que celles-ci ne se perdent pas. Au début nous avons utilisés le server de base de donnée MySQL. Puis, nous voulions que notre projet se suffise à lui-même, c’est-à-dire que lorsque nous téléchargeons celui-ci, il n’y a pas besoin d’utiliser d’autres outils ou librairies externes au projet. Ainsi, nous sommes passé sur une base de donnée SqLite, dont le fichier se trouve directement dans notre projet. Ceci permet un meilleur déploiement de notre solution.

Au niveau du serveur, nous avions besoin d’un serveur web afin d’exécuter le code PHP. Pour se faire, nous avions commencé par utiliser le serveur Apache. C’est un serveur très connu et qui fonctionne très bien. Cependant, toujours dans un souci de déploiement, nous préférions un serveur qui soit intégré à notre dossier de projet. Comme le logiciel est utilisé en local, c’est-à-dire qu’il n’est pas connecté à Internet, nous avons pu nous servir du server web intégré à Symfony. Ce qui nous a permis de nous passer d’Apache et ainsi supprimer une dépendance de taille au projet.

De plus, au niveau de la gestion des documents génériques, nous pensions utiliser des fichier PDF remplissable. Cependant, cette solution nous contraignait à devoir acheter une license Adobe afin de pouvoir remplir les fichiers PDF. Nous avons donc cherché d’autres solutions, et nous avons décidé d’utiliser plutôt des fichiers word (DOCX). Ceux-ci sont en effet remplissable directement par le code PHP et les modules Symfony.

## Description du projet

### Côté backend

Le backend, c’est tout ce que l’utilisateur du logiciel ne voit pas. Il contient le code métier.

C’est lui qui va fournir les différents éléments au frontend. Par exemple, les locataires, les contrats, les locaux ou les documents.

Ces differents éléments sont accessibles grâce à des chemins, que l’on appelle des routes. (par exemple : /locataires/12 ou /contrats/1/addDocument)

C’est aussi le backend qui va permettre d’envoyer des emails automatiquement lorsqu’un contrat est ajouté et c’est aussi lui qui va permettre de remplir les documents templates avec les informations fournies lors de la création d’un contrat.

Les documents ainsi que les templates sont stockés selon une arborescence de dossiers bien précise :

* users
* default
* generated Dossier des templates générés automatiquement
* record1 Document templates pour le contrat numéro 1
* record2 Document templates pour le contrat numéro 2
* ……
* templates Dossier des contenant les templates vides
* general Contient les templates généraux
* typed Contient les templates typés
* apartments Contient les templates pour les appartements
* hangars Contient les templates pour les hangars
* unique Contient documents pour les locaux
* room1 Contient les documents relatifs au local numéro 1
* room2 Contient les documents relatifs au local numéro 2
* …..

### Côté frontend

# II Résultats et perspectives

## Utilisation de la solution

### Gestion des locataires

### Gestion des locaux

### Gestion des contrats

### Gestion des documents

## Perspectives d’amélioration

Utilisation de Jhipster afin de bien séparer les parties frontend et backend, avec l’utilisation d’API REST. Jhiipster est bien adpaté pour ce genre de projet qui utilise des formulaires et des entités. De plus, il gère aussi l’intégration et le déploiement continu. Et il est très facile de lancer le site sur le cloud.

# Conclusion

# Références bibliographiques

# Glossaire