

Contents

[Introduction 1](#_Toc170416299)

[Contexte 1](#_Toc170416300)

[Objectif 1](#_Toc170416301)

[Périmètre du Projet 1](#_Toc170416302)

[Analyse des Besoins 2](#_Toc170416303)

[Utilisateurs Cibles 2](#_Toc170416304)

[Fonctionnalités Principales 2](#_Toc170416305)

[Pour les Clients 2](#_Toc170416306)

[Exigences Fonctionnelles 2](#_Toc170416307)

[Exigences Non-Fonctionnelles 3](#_Toc170416308)

[Architecture du Système 3](#_Toc170416309)

[Vue d'Ensemble de l'Architecture 3](#_Toc170416310)

[Technologies Utilisées 3](#_Toc170416311)

[Conception du Backend – Laravel 3](#_Toc170416312)

[Modèle de Données 3](#_Toc170416313)

[API RESTful 4](#_Toc170416314)

[Inscription et Gestion des Rafistoleurs 4](#_Toc170416315)

[Inscription et Gestion des Clients 4](#_Toc170416316)

[API pour la Recherche des Rafistoleurs Basée sur la Localisation 4](#_Toc170416317)

[Sécurité 5](#_Toc170416318)

[Conception du Frontend Desktop – Angular 5](#_Toc170416319)

[Structure de l'Application 5](#_Toc170416320)

[Composants Principaux 5](#_Toc170416321)

[Services et Communication avec l'API 5](#_Toc170416322)

[Conception du Frontend Mobile – Floater 6](#_Toc170416323)

[Structure de l'Application 6](#_Toc170416324)

[Composants Principaux 6](#_Toc170416325)

[Services et Communication avec l'API 6](#_Toc170416326)

[Déploiement 6](#_Toc170416327)

[Environnement de Développement 6](#_Toc170416328)

[Environnement de Production 7](#_Toc170416329)

[Gestion de Projet 7](#_Toc170416330)

[Planification 7](#_Toc170416331)

[Gestion des Risques 7](#_Toc170416332)

[Annexes 8](#_Toc170416333)

[Glossaire 8](#_Toc170416334)

[Références 8](#_Toc170416335)

# Introduction

## Contexte

En Côte d'Ivoire, les rafistoleurs, communément appelés "toclo toclo", consacrent leurs journées à marcher sous un soleil brûlant, la pluie à la recherche de clients. Ce mode de fonctionnement traditionnel est non seulement éprouvant mais aussi inefficace, limitant les opportunités pour les rafistoleurs de trouver rapidement des clients et pour les clients de localiser des rafistoleurs disponibles.

## Objectif

Le projet vise à transformer ce processus en développant une application mobile innovante. Cette application permettra aux rafistoleurs de se connecter plus facilement avec des clients potentiels dans leur zone géographique immédiate. En digitalisant ce service, l'application facilitera la mise en relation rapide et efficace entre rafistoleurs et clients, améliorant ainsi les conditions de travail des rafistoleurs et l'accès aux services pour les clients.

## Périmètre du Projet

* Création d'une application web pour desktop utilisant Angular.
* Création d'une application mobile utilisant Floater.
* Développement d'un backend commun avec Laravel pour gérer les API.

# Analyse des Besoins

## Utilisateurs Cibles

* Rafistoleurs (Toclo Toclo) : Cherchent à trouver des clients.
* Clients : Cherchent à trouver des rafistoleurs disponibles.

## Fonctionnalités Principales

* Inscription
* Collecte des informations personnelles : nom, prénom, numéro de téléphone, photo, localisation, disponibilité.
* Gestion de Profil
* Mise à jour des informations personnelles et de la disponibilité.

## Pour les Clients

* Inscription
* Collecte des informations personnelles : nom, prénom, numéro de téléphone, photo, localisation.
* Recherche de Rafistoleurs
* Recherche basée sur la localisation pour trouver les rafistoleurs disponibles.
* Affichage des résultats avec les informations nécessaires pour contacter les rafistoleurs.

## Exigences Fonctionnelles

* Système d'authentification sécurisé.
* Interface utilisateur intuitive pour la recherche et la gestion des demandes.

## Exigences Non-Fonctionnelles

* Performance et rapidité de l'application.
* Scalabilité pour gérer un grand nombre d'utilisateurs.
* Sécurité des données utilisateur.

# Architecture du Système

## Vue d'Ensemble de l'Architecture

* Frontend Desktop : Angular
* Frontend Mobile : Flutter
* Backend : Laravel
* Base de Données : MySQL

## Technologies Utilisées

* Angular : Framework pour l'application web desktop.
* Flutter : Framework pour l'application mobile.
* Laravel : Framework pour le backend et les API.
* MySQL : Base de données relationnelle.
* Git Refferenciel : <https://github.com/alsondab/rafistoleurs-app.git>
* Canva : Pour la presentation du projet
* StarUml : Pour la conception des Diagrame Uml
* Swagger UI : Pour les API [Swagger UI (miniprojectideas.com)](https://freeapi.miniprojectideas.com/index.html)

# Conception du Backend – Laravel

## Modèle de Données

* Table rafistoleurs : iD, nom, prenom, téléphone, photo, localisation.
* Table clients : iD, nom, prenom, téléphone, photo, localisation.

## API RESTful

* Endpoints :

1. API pour la Gestion des Utilisateurs (Rafistoleurs et Clients).

## Inscription et Gestion des Rafistoleurs

POST /api/rafistoleurs : Créer un nouveau rafistoleur.

GET /api/rafistoleurs/{id} : Récupérer les informations d'un rafistoleur par son ID.

PUT /api/rafistoleurs/{id} : Mettre à jour les informations d'un rafistoleur par son ID.

DELETE /api/rafistoleurs/{id} : Supprimer un rafistoleur par son ID.

GET /api/rafistoleurs : Récupérer la liste de tous les rafistoleurs.

## Inscription et Gestion des Clients

POST /api/clients : Créer un nouveau client.

GET /api/clients/{id} : Récupérer les informations d'un client par son ID.

PUT /api/clients/{id} : Mettre à jour les informations d'un client par son ID.

DELETE /api/clients/{id} : Supprimer un client par son ID.

GET /api/clients : Récupérer la liste de tous les clients.

## API pour la Recherche des Rafistoleurs Basée sur la Localisation

GET /api/rafistoleurs/search : Rechercher des rafistoleurs basés sur la localisation. Le paramètre de recherche inclut localisation.

## Sécurité

* Utilisation de JWT pour l'authentification.
* Validation des données d'entrée.
* Protection contre les attaques CSRF.

# Conception du Frontend Desktop – Angular

## Structure de l'Application

* Modules principaux : AuthModule, clientsModule, rafistoleursModule.
* Composants : LoginComponent, clientsProfileComponent, rafistoleursProfileComponent, rafistoleursListComponent.

## Composants Principaux

* LoginComponent : Formulaire de connexion.
* clientsProfileComponent et rafistoleursProfileComponent : Gestion du profil clients et rafistoleurs.
* rafistoleursListComponent. : Affichage les rafistoleur dans le voisinage.

## Services et Communication avec l'API

* ApiService : Service pour communiquer avec les endpoints Laravel.
* getClients(), createClients(), getRafistoleurs(), createRafistoleurs().

# Conception du Frontend Mobile – Floater

## Structure de l'Application

* Pages principales : LoginPage, ClientsProfilePage, RafistoleursProfilePage, RafistoleursListPage.
* Composants : LoginForm, ClientsProfileForm, RafistoleursProfileForm, RafistoleursCard.

## Composants Principaux

* LoginPage : Page de connexion.
* ClientsProfilePage et RafistoleursProfilePage : Page de gestion du profil Clients et Rafistoleurs.
* RafistoleursListPage : Page d'affichage des rafistoleur dans le voisinage.

## Services et Communication avec l'API

* ApiService : Service pour communiquer avec les endpoints Laravel.
* `getClients()`, `createClients()`, `getRafistoleurs()`, `createRafistoleurs()`.

# Déploiement

## Environnement de Développement

* Utilisation de Docker pour les environnements de développement.
* Configuration de l'environnement local avec Laravel, Angular, et Floater.

## Environnement de Production

* Déploiement de l'application Laravel sur Heroku.
* Déploiement de l'application Angular sur Netlify.
* Déploiement de l'application mobile sur Firebase App Distribution.

# Gestion de Projet

## Planification

* Phase de Conception : 4 Jours.
* Phase de Développement : 2 semaines.
* Phase de Test : 3 Jours.
* Phase de Déploiement 3 Jours.

## Gestion des Risques

* Risques Techniques : Problèmes d'intégration, bugs critiques, etc.
* Risques de Planning : Retards dans le développement, changements de spécifications.
* Plan de Contingence : Mise en place de solutions alternatives, extensions de délai si nécessaire.

# Annexes

## Glossaire

* **Rafistoleur (Toclo Toclo**) : Artisan qui répare ou coud des vêtements, localement appelé "toclo toclo" en Côte d'Ivoire.
* **API** : Interface de Programmation d'Application.
* **JWT** : JSON Web Token, utilisé pour l'authentification sécurisée.
* **Client** : Personne qui cherche à faire réparer ou ajuster ses vêtements par un rafistoleur.
* **Application mobile** : Programme installé sur un smartphone ou une tablette permettant aux rafistoleurs et aux clients de se connecter.
* **Disponibilité** : Statut indiquant si un rafistoleur est prêt à accepter des travaux ou non.
* **Digitalisation** : Processus de conversion de services ou de données en format numérique.
* **Interface utilisateur (UI)** : Ensemble des éléments visuels avec lesquels un utilisateur interagit sur une application.
* **Expérience utilisateur (UX)** : Perception et réaction d'un utilisateur face à l'utilisation d'une application.

# References

Documentation officielle d'Angular : [Angular Documentation](https://angular.io/docs)

Documentation officielle de Laravel : [Laravel Documentation](https://laravel.com/docs)

Documentation officielle de Floater : [Floater Documentation]( https://docs.flutterflow.io/)