

# CAHIER DES CHARGES

## Application Restaurant Mobile Flutter: CHELSY

### 1. INFORMATIONS GÉNÉRALES

#### 1.1. Identification du Projet

- **Nom du projet** : Application Restaurant Mobile
- **Plateforme** : Flutter (Android & iOS)
- **Type de projet** : Application mobile de commande et livraison de repas pour un restaurant unique
- **Date de création** : 24 novembre 2025
- **Version du document** : 1.0

#### 1.2. Contexte et Objectifs

Ce projet s'inscrit dans le cadre du cours de Flutter Development d'application. L'objectif est de développer une application mobile dédiée à un restaurant unique permettant aux utilisateurs de :

- Découvrir le restaurant et son histoire
- Parcourir le menu du restaurant
- Commander des plats en ligne (livraison ou à emporter)
- Gérer leurs adresses de livraison

Ce projet permet de mettre en pratique les compétences acquises en développement mobile avec Flutter et de découvrir les concepts avancés de développement d'applications.

#### 1.3. Portée du Projet

Cette application vise à créer un lien direct entre le restaurant et ses clients, en facilitant la commande de repas et en offrant une expérience utilisateur optimale avec des fonctionnalités de commande à emporter, livraison et réservation.

**Contexte** : Projet scolaire réalisé dans le cadre du cours de **Flutter Development d'application**.

### 2. ÉQUIPE PROJET

#### 2.1. Composition de l'Équipe

- **LAWINGNI Abdoul Rachard** - Développeur
- **SEHLIN Divin** - Développeur
- **AHOUCANDJINO Chelsy** - Développeur et Designer UI/UX

**Total** : 3 membres

## **2.2. Rôles et Responsabilités**

L'équipe travaille de manière collaborative dans le cadre d'un projet d'apprentissage. Chaque membre participe activement au développement et à l'apprentissage de Flutter et GetX.

- **LAWINGNI Abdoul Rachard :**

- Développement Flutter
- Contribution à l'architecture et structure du projet
- Partage de connaissances avec l'équipe

- **SEHLIN Divin :**

- Apprentissage et développement de Flutter
- Implémentation de fonctionnalités
- Tests et débogage

- **AHOUANDJINO Chelsy :**

- Conception de l'interface utilisateur (UI/UX)
- Création des maquettes et wireframes (si nécessaire)
- Élaboration du design system
- Apprentissage et développement de Flutter

## **3. BESOINS FONCTIONNELS**

### **3.1. Authentification et Gestion de Compte**

#### **3.1.1. Inscription**

- Création de compte avec email/mot de passe
- Vérification de l'email
- Récupération de mot de passe
- Connexion avec Google/Facebook (optionnel)

#### **3.1.2. Connexion**

- Authentification par email/mot de passe
- Mémorisation de session
- Déconnexion

#### **3.1.3. Profil Utilisateur**

- Visualisation du profil
- Modification des informations personnelles
- Gestion des adresses de livraison (multiple)
- Historique des commandes
- Plats favoris

### **3.2. Découverte du Restaurant**

#### **3.2.1. Page "À Propos"**

- Histoire et valeurs du restaurant
- Présentation du chef et de l'équipe
- Horaires d'ouverture détaillés
- Coordonnées (téléphone, email, réseaux sociaux)
- Zone de livraison

### **3.3. Menu et Catalogue**

#### **3.3.1. Page d'Accueil / Menu Principal**

- Accès direct au menu complet
- Catégories de plats (Entrées, Plats, Desserts, Boissons)
- Menu du jour / Suggestions du chef
- Promotions en cours
- Nouveautés
- Recherche de plats

#### **3.3.2. Catégories de Plats**

- Filtres par catégorie
- Tri par prix, popularité
- Affichage des plats avec photo, nom, prix
- Indication de disponibilité (disponible/épuisé)
- Badge "Nouveauté", "Spécialité", "Végétarien", etc.

#### **3.3.3. Détails du Plat**

- Galerie de photos
- Description détaillée
- Prix et disponibilité en temps réel
- Temps de préparation estimé
- Allergènes et informations nutritionnelles
- Avis et notes clients
- Options personnalisables (ingrédients, taille, cuisson, etc.)
- Ajout au panier

### **3.4. Panier et Commande**

#### **3.4.1. Gestion du Panier**

- Ajout/retrait de plats
- Modification des quantités
- Personnalisation des plats
- Instructions spéciales pour la cuisine
- Calcul automatique du total (frais de livraison inclus)
- Code promo/réduction
- Sauvegarde du panier
- Minimum de commande (si applicable)

#### **3.4.2. Processus de Commande**

- Choix du mode de réception :

- Livraison (avec vérification de la zone de livraison)
- À emporter
- Sélection de l'adresse de livraison (si livraison)
- Validation de la zone de livraison
- Calcul des frais de livraison selon la distance
- Sélection de l'heure de livraison/récupération
- Choix du mode de paiement
- Récapitulatif de commande
- Confirmation de commande
- Envoi de SMS/email de confirmation

### **3.5. Suivi de Commande en Temps Réel**

#### **3.5.1. Statut de la Commande**

- Commande reçue
- Commande confirmée par le restaurant
- En préparation (avec temps estimé)
- Prête pour livraison/retrait
- En livraison (avec suivi GPS du livreur - optionnel)
- Livrée / Récupérée
- Annulée (avec motif)

#### **3.5.2. Notifications Push**

- Confirmation de commande
- Commande acceptée par le restaurant
- Début de préparation
- Commande prête
- Livreur en route (avec nom et contact)
- Arrivée imminente
- Commande livrée
- Rappel d'évaluation

### **3.7. Historique et Favoris**

#### **3.7.1. Historique des Commandes**

- Liste de toutes les commandes
- Détails d'une commande passée
- Recommander (commande rapide)
- Téléchargement de facture (PDF)
- Évaluation du plat et du service

#### **3.7.2. Plats Favoris**

- Liste des plats favoris
- Accès rapide pour commander
- Commandes récurrentes

### **3.8. Paiement**

### **3.8.1. Méthodes de Paiement**

- Carte bancaire ()
- Paiement à la livraison/au retrait
- Mobile Money

## **3.9. Avis et Notations**

### **3.9.1. Système d'Avis**

- Notation du restaurant (1-5 étoiles)
- Notation des plats individuels
- Notation du service de livraison
- Commentaires avec photos
- Réponse du restaurant aux avis
- Modération des avis

## **3.10. Support Client**

### **3.10.1. Communication avec le Restaurant**

- Appel téléphonique direct
- Email
- FAQ
- Réclamation / Retour
- Suivi des réclamations

# **4. BESOINS NON FONCTIONNELS**

## **4.1. Performance**

- Application fonctionnelle et fluide
- Optimisation raisonnable des images et performances
- Temps de chargement < 3 secondes
- Temps de réponse acceptable pour les actions utilisateur
- Gestion efficace du cache

## **4.2. Sécurité**

- Sécurisation des données utilisateur (conformément aux bonnes pratiques)
- Authentification fonctionnelle et sécurisée
- Protection des données sensibles (paiement)
- Chiffrement des communications
- Conformité RGPD (protection des données personnelles)

## **4.3. Compatibilité**

- Android : API 21+ (Android 5.0+)
- iOS : iOS 12.0+ (si développement iOS disponible)
- Support smartphone (tablette optionnel)
- Mode portrait principalement (paysage optionnel)

#### **4.4. Expérience Utilisateur**

- Interface intuitive et appétissante
- Design cohérent avec l'identité visuelle du restaurant
- Photos de haute qualité des plats
- Navigation claire et logique
- Feedback utilisateur (chargement, erreurs, succès)
- Accessibilité (contraste, taille de texte)
- Mode sombre (optionnel)

#### **4.5. Disponibilité**

- Application fonctionnelle en ligne
- Mise en cache locale des données essentielles
- Mode hors ligne limité (consultation du menu)
- Synchronisation automatique

#### **4.6. Géolocalisation**

- Localisation GPS pour la livraison
- Calcul de la distance depuis le restaurant
- Vérification de la zone de livraison
- Calcul des frais de livraison selon la distance
- Affichage du restaurant sur une carte

### **5. CONTRAINTES TECHNIQUES**

#### **5.1. Technologies**

##### **5.1.1. Framework et Langage**

- Framework : Flutter (Dart)
- Version Flutter : 3.0+
- Gestion d'état : GetX (obligatoire pour ce projet)

##### **5.1.2. Base de Données**

- Locale : Hive / SQLite (cache)
- Backend : À définir (PostgreSQL, MongoDB, Firebase Firestore)

##### **5.1.3. Services**

- Authentification : Firebase Auth
- Stockage cloud : Firebase Storage (pour images)
- API : REST ou GraphQL
- Paiement : Stripe SDK / Mobile Money API
- Géolocalisation : Google Maps API / Mapbox
- Push Notifications : Firebase Cloud Messaging
- WebSocket : Pour le suivi en temps réel (optionnel)
- Analytics : Firebase Analytics

#### **5.2. Architecture**

### 5.2.1. Structure du Projet

- Architecture modulaire (Clean Architecture recommandée)
- Séparation des couches :
  - Présentation : UI et widgets
  - Logique métier : Contrôleurs GetX
  - Données : Repositories et sources de données

### 5.2.2. GetX

- Utilisation de GetX pour :
  - Gestion d'état (GetXController)
  - Navigation (Get.to, Get.off, etc.)
  - Injection de dépendances (Bindings)
  - Gestion de l'état global et local
- Contrôleurs GetX pour chaque fonctionnalité
- Bindings GetX pour l'injection de dépendances

## 5.3. Outils de Développement

- IDE : Android Studio / VS Code
- Versioning : Git / GitHub
- Gestion de projet : GitHub Projects / Trello (optionnel)
- Tests : Flutter Test Framework

# 6. INTERFACES ET DESIGN

## 6.1. Design System

### 6.1.1. Identité Visuelle

- Identité visuelle spécifique au restaurant
- Logo du restaurant
- Couleurs de la marque
- Typographie personnalisée
- Éléments graphiques distinctifs

### 6.1.2. Design Guidelines

- Material Design 3 ou Cupertino (iOS)
- Palette de couleurs appétissante et cohérente
- Typographie lisible et moderne
- Composants réutilisables
- Design centré sur les images de plats de haute qualité
- Animations fluides et subtiles

## 6.2. Écrans Principaux

1. Écran de démarrage / Splash Screen
2. Onboarding (première utilisation)
3. Authentification (Login/Register)
4. Accueil - Menu principal du restaurant

5. À propos du restaurant
6. Catégorie de plats
7. Détails du plat
8. Panier
9. Checkout / Commande
10. Suivi de commande en temps réel
11. Profil utilisateur
12. Historique des commandes
13. Plats favoris
14. Événements et actualités

### **6.3. Wireframes / Maquettes**

- [À ajouter si éventuellement on fait une maquette]
- Liens vers les maquettes Figma/Adobe XD

## **7. INTÉGRATIONS ET API**

### **7.1. API Backend**

#### **7.1.1. Configuration**

- Base URL : [À définir]
- Authentification : JWT / OAuth 2.0
- Format : JSON
- Versioning : /api/v1

### **7.2. Services Externes**

#### **7.2.1. Firebase**

- Authentication : Gestion des utilisateurs
- Cloud Firestore : Base de données (optionnel)
- Storage : Stockage des images
- Cloud Messaging : Notifications push
- Analytics : Suivi de l'usage

#### **7.2.2. Paiement**

- Stripe : Paiement par carte (si possible)
- Mobile Money : MTN, Moov, etc.

#### **7.2.3. Géolocalisation**

- Google Maps API : Cartes et géolocalisation
- Calcul de distance
- Affichage du restaurant

## **8. STRUCTURE DE DONNÉES**



## 8.1. Modèles Principaux

- User (Utilisateur)
- Restaurant (Restaurant)
- Menu (Menu) • Dish (Plat)
- Category (Catégorie de plat)
- Cart (Panier)
- Order (Commande)
- OrderItem (Article de commande)
- Address (Adresse)
- Payment (Paiement)
- Review (Avis)
- Réservation (Réservation - optionnel)

## 8.2. Base de Données

### 8.2.1. Backend

- À définir : PostgreSQL, MongoDB, Firebase Firestore
- Schéma relationnel ou NoSQL selon le choix

### 8.2.2. Locale (Cache)

- Hive / SQLite pour le cache local
- Données mises en cache :
  - Menu et plats
  - Informations du restaurant
  - Historique des commandes
  - Favoris
  - Profil utilisateur

## 9. PLANNING ET LIVRABLES

### 9.1. Phases du Projet

Note : Le projet sera développé progressivement au fur et à mesure de l'avancement du cours. Le cours totalise 40h et le planning s'ajustera selon la progression des cours. Les phases suivantes sont indicatives et seront adaptées selon l'avancement réel.

### Phase 1 : Analyse et Conception

#### Objectifs :

- Finalisation du cahier des charges
- Définition de l'identité visuelle du restaurant
- Création des maquettes (si nécessaire)
- Architecture technique

#### Livrables :

- Cahier des charges finalisé
- Charte graphique et identité visuelle
- Maquettes (optionnel)

- Document d'architecture

## **Phase 2 : Développement - Setup et Authentification**

### **Objectifs :**

- Configuration de l'environnement Flutter
- Apprentissage de Flutter et GetX
- Setup du backend / API (selon choix)
- Configuration de GetX dans le projet
- Implémentation de l'authentification
- Setup Firebase

### **Livrables :**

- Environnement configuré
- Structure GetX en place
- Authentification fonctionnelle
- Splash screen et onboarding

## **Phase 3 : Développement - Restaurant et Menu**

### **Objectifs :**

- Page d'accueil avec menu principal
- Page "À propos du restaurant"
- Catégories de plats
- Détails des plats
- Recherche et filtres
- Gestion de la disponibilité en temps réel

### **Livrables :**

- Menu complet fonctionnel
- Navigation fluide
- Affichage des détails du restaurant

## **Phase 4 : Développement - Panier et Commande**

### **Objectifs :**

- Gestion du panier
- Personnalisation des plats
- Processus de commande complet
- Gestion des adresses de livraison
- Vérification de la zone de livraison
- Intégration paiement (Stripe si possible, Mobile Money)

### **Livrables :**

- Système de commande fonctionnel
- Panier avec calculs automatiques
- Paiement sécurisé

## **Phase 5 : Développement - Suivi et Notifications**

### **Objectifs :**

- Suivi de commande en temps réel
- Notifications push
- Historique des commandes
- Système d'avis et notations

**Livrables :**

- Suivi temps réel fonctionnel
- Notifications opérationnelles
- Système d'avis complet

## **Phase 6 : Développement - Événements et Support**

**Objectifs :**

- Page événements et actualités
- FAQ et support client
- Profil utilisateur complet

**Livrables :**

- Système d'événements fonctionnel

## **Phase 7 : Finalisation et Tests**

**Objectifs :**

- Tests fonctionnels complets
- Tests unitaires
- Corrections de bugs
- Optimisations de performance
- Documentation du code
- Guide utilisateur

**Livrables :**

- Application testée et fonctionnelle
- Documentation technique complète
- Guide utilisateur

## **Phase 8 : Déploiement (Optionnel)**

**Objectifs :**

- Compilation APK/IPA
- Publication sur Google Play Store (si déploiement)
- Publication sur Apple App Store (si déploiement)
- Documentation finale du projet

**Livrables :**

- Application publiée (si déploiement)
- Documentation complète
- Présentation du projet

## **9.2. Livrables Finaux**

- Application mobile (APK/IPA)

- Code source (repository Git)
- Documentation technique
- Guide utilisateur
- Présentation du projet

## 10. CRITÈRES D'ACCEPTATION

### 10.1. Fonctionnels

- Toutes les fonctionnalités essentielles sont implémentées
- L'authentification fonctionne correctement
- Le menu et les détails des plats sont accessibles
- Le processus de commande est complet et fonctionnel
- Le suivi en temps réel fonctionne
- Les paiements sont sécurisés
- La géolocalisation et la zone de livraison fonctionnent

### 10.2. Techniques

- Code propre et documenté
- Respect de l'architecture définie
- Utilisation correcte de GetX
- Tests unitaires (> 60% de couverture)
- Application sans crash majeur
- Performance acceptable

### 10.3. Qualité

- Interface utilisateur professionnelle et appétissante
- Design cohérent avec l'identité du restaurant
- Photos de qualité des plats
- Performance acceptable
- Sécurité des données
- Compatibilité multi-plateforme
- Expérience utilisateur fluide

## 11. RISQUES ET CONTRAINTES

### 11.1. Risques Identifiés

- **Risque d'apprentissage** : Apprentissage de Flutter et GetX pour tous les membres
- **Risque technique** : Complexité de certaines fonctionnalités (paiement, géolocalisation, temps réel)
- **Contrainte de temps** : Projet réalisé sur 40h de cours, progression variable
- **Disponibilité des services** : Dépendance aux services externes (Firebase, Stripe, etc.)
- **Qualité des données** : Nécessité de contenu de qualité (photos, descriptions)

### 11.2. Mitigation

- Apprentissage collaboratif et partage de connaissances au sein de l'équipe
- Documentation et tutoriels utilisés en commun

- Focus sur les fonctionnalités essentielles en priorité
- Adaptation du scope selon
- Tests et corrections au fur et à mesure

## 12. ANNEXES

### 12.1. Glossaire

- **API** : Application Programming Interface
- **JWT** : JSON Web Token
- **SDK** : Software Development Kit
- **GPS** : Global Positioning System
- **WebSocket** : Protocole de communication temps réel

### 12.2. Références

- **Documentation Flutter** : <https://flutter.dev/docs>
- **Documentation GetX** : <https://pub.dev/packages/get>
- **Guide GetX** : <https://github.com/jonataslaw/getx>
- **Firebase** : <https://firebase.google.com/docs>
- **Google Maps API** : <https://developers.google.com/maps>
- **Stripe** : <https://stripe.com/docs>

**Document créé le** : 24 novembre 2025

**Dernière mise à jour** : 24 novembre 2025

**Prochaine révision** : À définir selon avancement du projet