



Flutter Frontend Implementation TODO List

AI Agent Step-by-Step Development Guide

Phase 1: Project Foundation & Setup (Sprint 1-2)



Task 1.1: App Launch Experience & Branding

Priority: Critical | Estimated Time: 4-5 hours

☐ **Splash Screen with Animated Logo**

- ☐ Create `lib/features/splash/presentation/screens/splash_screen.dart`
- ☐ Design and implement animated company logo
- ☐ Add Lottie animation for logo entrance effect:
- ☐ Logo scale animation (0.5x to 1.2x to 1.0x)
- ☐ Logo opacity fade-in animation
- ☐ Background gradient animation
- ☐ Implement typewriter effect for app name "FoodApp Cameroon"
- ☐ Add tagline with fade-in animation "Délicieux à votre porte"
- ☐ Create loading indicator with custom styling
- ☐ Add app initialization logic (3-4 seconds duration)
- ☐ Handle splash screen navigation to next screen

☐ **Terms and Conditions Screen**

- ☐ Create `lib/features/legal/presentation/screens/terms_conditions_screen.dart`
- ☐ Design interactive terms and conditions layout:
- ☐ Animated header with description icon
- ☐ Scrollable terms content container
- ☐ Scroll progress indicator
- ☐ "Read to bottom" detection logic
- ☐ Animated checkbox for terms acceptance
- ☐ Enable/disable continue button based on acceptance
- ☐ Create `lib/features/legal/widgets/terms_content.dart` with:
- ☐ Complete terms and conditions content in French
- ☐ Formatted sections with headers and bullet points
- ☐ Company information and legal disclaimers
- ☐ Contact information for legal inquiries
- ☐ Implement smooth navigation to onboarding after acceptance

☐ **Welcome & Onboarding Screens**

☐ Create `lib/features/onboarding/presentation/screens/welcome_screen.dart`

☐ Design 3-page onboarding flow:

☐ **Page 1:** "Bienvenue chez FoodApp" - Company introduction

☐ **Page 2:** "Notre Mission" - Connect Cameroonians with restaurants

☐ **Page 3:** "Livraison Rapide" - Fast delivery promise

☐ Implement page view with smooth transitions

☐ Add animated page indicators

☐ Create skip button for quick navigation

☐ Add next/previous navigation controls

☐ Include Lottie animations for each onboarding page

☐ Implement "Get Started" button on final page

☐ **Company Valorization Pages**

☐ Create `lib/features/company/presentation/screens/`:

☐ `about_us_screen.dart` - Company story and mission

☐ `our_team_screen.dart` - Staff showcase and team members

☐ `contact_us_screen.dart` - Contact information and form

☐ `careers_screen.dart` - Job opportunities and company culture

☐ **About Us Screen Implementation:**

☐ Hero section with company image and parallax effect

☐ Company mission statement section

☐ Company values with animated icons

☐ Statistics section (orders delivered, restaurants partnered, etc.)

☐ Timeline of company milestones

☐ Testimonials from customers and partners

☐ **Our Team Screen Implementation:**

☐ Team introduction section with animated header

☐ Team members grid with photo and bio:

☐ Founder/CEO profile with story

☐ Key team members (CTO, Operations Manager, etc.)

☐ Department heads (Marketing, Support, etc.)

☐ Company culture highlights

☐ "Join Our Team" call-to-action section

☐ Animated team statistics

☐ **Contact Us Screen Implementation:**

☐ Contact information cards:

☐ Phone number with click-to-call

☐ Email address with click-to-email

☐ WhatsApp Business number

- ☐ Office address with map integration
- ☐ Contact form with validation:
 - ☐ Name, email, message fields
 - ☐ Form validation and error handling
 - ☐ Success animation after submission
- ☐ Office hours and location details
- ☐ Social media links
- ☐ **Navigation Flow Setup**
 - ☐ Configure app launch sequence:
 1. Splash Screen (3-4 seconds)
 2. Terms & Conditions (first-time users)
 3. Welcome/Onboarding (first-time users)
 4. Main App Interface
 - ☐ Implement first-time user detection with shared preferences
 - ☐ Create smooth transitions between screens
 - ☐ Add proper back navigation handling
 - ☐ Implement "Skip" functionality for returning users

Acceptance Criteria:

- ☒ Animated splash screen with company branding
 - ☒ Interactive terms and conditions with acceptance flow
 - ☒ Complete 3-page onboarding experience
 - ☒ Professional About Us and Team showcase pages
 - ☒ Functional contact form and information
 - ☒ Smooth navigation flow from launch to main app
 - ☒ First-time user detection working properly
-

Task 1.2: Initial Project Setup & Configuration

Priority: Critical | Estimated Time: 2-3 hours

- ☐ **Initialize Flutter Project**
 - ☐ Create new Flutter project with `flutter create food_delivery_app`
 - ☐ Configure project for Flutter 3.24.0 and Dart 3.5.0
 - ☐ Setup proper folder structure following clean architecture
 - ☐ Configure `analysis_options.yaml` with strict linting rules

☐ Setup `.gitignore` for Flutter projects

☐ **Dependencies Installation & Configuration**

☐ Add all core dependencies to `pubspec.yaml`:

yaml

State Management

flutter_riverpod: ^2.4.0

riverpod_annotation: ^2.3.0

riverpod_generator: ^2.3.0

Navigation

go_router: ^12.0.0

auto_route: ^7.0.0

Networking

dio: ^5.3.0

retrofit: ^4.0.0

pretty_dio_logger: ^1.3.0

Local Storage

hive: ^2.2.3

hive_flutter: ^1.1.0

drift: ^2.12.0

shared_preferences: ^2.2.0

UI & Animations

lottie: ^2.6.0

rive: ^0.11.0

shimmer: ^3.0.0

flutter_staggered_animations: ^1.1.1

cached_network_image: ^3.3.0

Real-time & Notifications

socket_io_client: ^2.0.3

firebase_messaging: ^14.6.0

flutter_local_notifications: ^15.1.0

Maps & Location

google_maps_flutter: ^2.5.0

location: ^5.0.0

geocoding: ^2.1.0

Utilities

intl: ^0.18.1

url_launcher: ^6.1.14

image_picker: ^1.0.4

permission_handler: ^11.0.0

font_awesome_flutter: ^10.5.0

share_plus: ^7.1.0

Development

build_runner: ^2.4.0

json_annotation: ^4.8.1

json_serializable: ^6.7.0

flutter_launcher_icons: ^0.13.1

- ☐ Run `flutter pub get` and resolve any version conflicts
- ☐ Setup code generation with `build_runner`
- ☐ **Assets Preparation**
- ☐ Create `assets/` folder structure:

```
assets/  
├── images/  
│   ├── logo/  
│   │   ├── app_logo.png  
│   │   └── app_logo_transparent.png  
│   ├── company/  
│   │   ├── company_hero.jpg  
│   │   ├── team_photo.jpg  
│   │   └── office_photo.jpg  
│   ├── onboarding/  
│   │   ├── welcome_food.png  
│   │   ├── mission.png  
│   │   └── delivery.png  
│   └── team/  
│       ├── ceo_photo.jpg  
│       ├── cto_photo.jpg  
│       └── operations_manager.jpg  
├── animations/  
│   ├── app_logo_animation.json  
│   ├── welcome_food.json  
│   ├── mission.json  
│   ├── delivery.json  
│   └── loading_spinner.json  
└── icons/  
    ├── app_icon.png  
    └── notification_icon.png
```

- ☐ Configure assets in `pubspec.yaml`
- ☐ Optimize all images for mobile (WebP format recommended)
- ☐ Create app icons for different platforms and sizes

☐ **Development Environment Setup**

- ☐ Configure VS Code/Android Studio with Flutter extensions
- ☐ Setup debugging configurations
- ☐ Configure hot reload and hot restart
- ☐ Setup emulators for Android and iOS testing
- ☐ Configure Git hooks for pre-commit linting

Acceptance Criteria:

- ☒ Flutter app runs successfully on both Android and iOS
 - ☒ All dependencies compile without errors
 - ☒ Code generation works properly
 - ☒ Development environment is fully configured
-

Task 1.3: Design System Implementation

Priority: Critical | Estimated Time: 4-5 hours

☐ **Core Design Tokens & Theme**

☐ Create `lib/core/theme/design_tokens.dart` with:

- ☐ Color palette constants
- ☐ Typography scale
- ☐ Spacing system (8dp grid)
- ☐ Border radius constants
- ☐ Elevation levels
- ☐ Animation durations

☐ Create `lib/core/theme/app_theme.dart` with:

- ☐ Light theme configuration
- ☐ Dark theme configuration
- ☐ Custom color scheme
- ☐ Typography theme
- ☐ Component themes (AppBar, Button, Card, etc.)

☐ **Typography System**

- ☐ Implement custom TextStyles following Material Design 3
- ☐ Create responsive text sizing utilities
- ☐ Setup font loading (Google Fonts integration)
- ☐ Create text theme extensions
- ☐ Implement accessibility font scaling

- ☐ **Custom Components Library**
- ☐ Create `lib/shared/widgets/` directory structure:
- ☐ `buttons/` - Custom button components
- ☐ `cards/` - Various card layouts
- ☐ `forms/` - Input fields and form components
- ☐ `loading/` - Loading states and skeletons
- ☐ `navigation/` - Navigation components
- ☐ `dialogs/` - Modal and dialog components
- ☐ Implement base components:
- ☐ `AppBar` with loading and disabled states
- ☐ `AppCard` with elevation and shadows
- ☐ `AppTextField` with validation styling
- ☐ `LoadingIndicator` with custom animations
- ☐ `EmptyState` with illustrations
- ☐ `ErrorState` with retry functionality
- ☐ **Responsive Layout System**
- ☐ Create `lib/core/utils/responsive.dart`:
- ☐ Screen breakpoint constants
- ☐ Responsive helper methods
- ☐ Adaptive widget utilities
- ☐ Screen type detection (mobile/tablet/desktop)

Acceptance Criteria:

- ☒ Complete design system with consistent theming
- ☒ All base components render correctly
- ☒ Responsive layout works on different screen sizes
- ☒ Light and dark themes implemented
- ☒ Typography scales properly with system settings

Task 1.4: Project Structure & Architecture Setup

Priority: Critical | Estimated Time: 3-4 hours

- ☐ **Feature-based Architecture Setup**
- ☐ Create feature directories:

lib/features/

- |— splash/
 - | |— presentation/
 - | |— providers/
- |— legal/
 - | |— presentation/
 - | |— widgets/
- |— onboarding/
 - | |— presentation/
 - | |— providers/
- |— company/
 - | |— presentation/
 - | |— data/
 - | |— providers/
- |— auth/
 - | |— data/
 - | |— domain/
 - | |— presentation/
 - | |— providers/
- |— home/
- |— restaurants/
- |— orders/
- |— profile/
- |— cart/
- |— admin/
- |— driver/

☐ Create core directories:

lib/core/

- |— constants/
- |— error/
- |— network/
- |— utils/
- |— theme/

☐ **Base Classes & Interfaces**

- ☐ Create `lib/core/error/failures.dart` - Error handling classes
- ☐ Create `lib/core/error/exceptions.dart` - Custom exceptions
- ☐ Create `lib/core/network/network_info.dart` - Connectivity checking
- ☐ Create `lib/core/utils/validators.dart` - Form validation utilities

☐ Create base repository interfaces

☐ **Model Classes & Serialization**

- ☐ Create model classes for all entities:
- ☐ `User` model with JSON serialization
- ☐ `Restaurant` model with JSON serialization
- ☐ `MenuItem` model with JSON serialization
- ☐ `Order` model with JSON serialization
- ☐ `Address` model with JSON serialization
- ☐ Setup JSON serialization with `json_serializable`
- ☐ Generate model code with `build_runner`

Acceptance Criteria:

- ☒ Clean architecture structure implemented
 - ☒ All model classes with proper serialization
 - ☒ Base classes and interfaces created
 - ☒ Error handling system in place
 - ☒ Code generation working for models
-

Phase 2: Authentication & User Management (Sprint 3-4)

Task 2.1: Authentication System Implementation

Priority: Critical | Estimated Time: 6-8 hours

☐ Authentication UI Components

- ☐ Create `lib/features/auth/presentation/screens/`:
- ☐ `splash_screen.dart` - Animated splash with logo
- ☐ `onboarding_screen.dart` - App introduction slides
- ☐ `login_screen.dart` - Login form with validation
- ☐ `register_screen.dart` - Registration form
- ☐ `forgot_password_screen.dart` - Password reset
- ☐ `otp_verification_screen.dart` - Phone/email verification
- ☐ `welcome_screen.dart` - Post-registration welcome

☐ Authentication Logic & State Management

- ☐ Create `lib/features/auth/providers/`:
- ☐ `auth_provider.dart` - Authentication state management
- ☐ `user_provider.dart` - User profile state
- ☐ `login_form_provider.dart` - Login form validation
- ☐ `register_form_provider.dart` - Registration form validation

- ☐ Implement authentication methods:
- ☐ Email/password login
- ☐ Phone number registration
- ☐ OTP verification
- ☐ Social login (Google, Facebook)
- ☐ Biometric authentication
- ☐ Token management (JWT)
- ☐ Automatic token refresh
- ☐ **Form Validation & User Experience**
- ☐ Implement real-time form validation
- ☐ Add loading states for all auth actions
- ☐ Create smooth transitions between auth screens
- ☐ Add error handling with user-friendly messages
- ☐ Implement password strength indicator
- ☐ Add "Remember Me" functionality
- ☐ **Security Features**
- ☐ Secure token storage using FlutterSecureStorage
- ☐ Implement session timeout
- ☐ Add device fingerprinting
- ☐ Biometric authentication integration
- ☐ Implement login attempt limiting

Acceptance Criteria:

- ☒ Complete authentication flow working
 - ☒ Form validation with real-time feedback
 - ☒ Secure token management
 - ☒ Social login integration
 - ☒ Smooth animations and transitions
 - ☒ Error handling and loading states
-

Task 2.2: User Profile Management

Priority: High | Estimated Time: 4-5 hours

- ☐ **Profile UI Implementation**
- ☐ Create `lib/features/profile/presentation/screens/`:
- ☐ `profile_screen.dart` - Main profile view

- ☐ `edit_profile_screen.dart` - Profile editing form
- ☐ `address_management_screen.dart` - Address CRUD
- ☐ `payment_methods_screen.dart` - Payment method management
- ☐ `notification_settings_screen.dart` - Notification preferences
- ☐ `account_settings_screen.dart` - Account configuration
- ☐ **Profile Features Implementation**
- ☐ Profile picture upload and crop functionality
- ☐ Multiple address management with map integration
- ☐ Payment method storage and selection
- ☐ Notification preferences (push, email, SMS, WhatsApp)
- ☐ Privacy settings and data export
- ☐ Account deletion functionality
- ☐ Language and region settings
- ☐ **Address Management System**
- ☐ Google Maps integration for address selection
- ☐ Geocoding and reverse geocoding
- ☐ Address validation and formatting
- ☐ Favorite addresses (Home, Work, etc.)
- ☐ Delivery zone validation
- ☐ Address search and autocomplete

Acceptance Criteria:

- ☒ Complete profile management system
 - ☒ Image upload and cropping working
 - ☒ Address management with map integration
 - ☒ Payment method storage
 - ☒ Notification preferences working
 - ☒ Account settings and privacy controls
-

Phase 3: Core App Features (Sprint 5-7)

Task 3.1: Home Screen & Dashboard

Priority: Critical | Estimated Time: 5-6 hours

- ☐ **Home Screen Layout & Components**
- ☐ Create `lib/features/home/presentation/screens/home_screen.dart`:

- ☐ Custom scroll view with collapsing app bar
- ☐ Location selector with current location display
- ☐ Search bar with voice search integration
- ☐ Promotional banners carousel
- ☐ Quick action buttons grid
- ☐ Restaurant categories section
- ☐ Recommended restaurants list
- ☐ Recent orders section
- ☐ **Home Screen Interactive Elements**
- ☐ Implement pull-to-refresh functionality
- ☐ Add floating cart button with item count badge
- ☐ Create notification bell with unread indicator
- ☐ Implement search with real-time suggestions
- ☐ Add voice search with animation
- ☐ Location change with map modal
- ☐ **Personalized Content**
- ☐ User greeting with name and avatar
- ☐ Personalized restaurant recommendations
- ☐ Recent order quick reorder options
- ☐ Favorite restaurants quick access
- ☐ Promotional content based on user preferences
- ☐ Weather-based food suggestions
- ☐ **Performance Optimizations**
- ☐ Implement lazy loading for restaurant list
- ☐ Add image preloading for visible items
- ☐ Cache restaurant data for offline viewing
- ☐ Optimize scroll performance with viewport
- ☐ Implement efficient state management

Acceptance Criteria:

- ☒ Responsive home screen with smooth scrolling
 - ☒ All interactive elements working properly
 - ☒ Personalized content displaying correctly
 - ☒ Search and voice search functional
 - ☒ Pull-to-refresh and lazy loading implemented
 - ☒ Performance optimized for large lists
-

Task 3.2: Restaurant Discovery & Menu System

Priority: Critical | Estimated Time: 8-10 hours

☐ Restaurant List & Search

- ☐ Create `lib/features/restaurants/presentation/screens/`:
- ☐ `restaurant_list_screen.dart` - Restaurant discovery
- ☐ `restaurant_detail_screen.dart` - Restaurant profile
- ☐ `menu_screen.dart` - Restaurant menu
- ☐ `search_screen.dart` - Advanced search
- ☐ `filter_screen.dart` - Filter options

☐ Restaurant Card & List Components

- ☐ Design restaurant card with:
- ☐ Hero image with placeholder and error states
- ☐ Restaurant name, rating, and delivery time
- ☐ Cuisine tags and price range
- ☐ Distance and delivery fee
- ☐ Favorite button with animation
- ☐ Open/closed status indicator
- ☐ Implement different list views (grid, list, map)
- ☐ Add skeleton loading for better UX

☐ Restaurant Detail Screen

- ☐ Parallax header with restaurant photos
- ☐ Restaurant information section
- ☐ Reviews and ratings display
- ☐ Operating hours and contact info
- ☐ Menu categories with sticky headers
- ☐ Share restaurant functionality
- ☐ Report/feedback options

☐ Menu System Implementation

- ☐ Menu category navigation (sticky tabs)
- ☐ Menu item cards with:
- ☐ Food images with zoom functionality
- ☐ Name, description, and price
- ☐ Dietary indicators (vegan, halal, etc.)
- ☐ Customization options modal
- ☐ Add to cart with quantity selector
- ☐ Nutritional information

- ☐ Search within menu functionality
- ☐ Menu item availability status
- ☐ Popular/recommended item badges
- ☐ **Advanced Search & Filtering**
- ☐ Text search with autocomplete
- ☐ Voice search integration
- ☐ Filter by:
- ☐ Cuisine type
- ☐ Price range
- ☐ Rating
- ☐ Delivery time
- ☐ Distance
- ☐ Dietary preferences
- ☐ Open now
- ☐ Sort options (popularity, rating, distance, price)
- ☐ Search history and saved searches
- ☐ Location-based search

Acceptance Criteria:

- ☒ Restaurant discovery with search and filters
 - ☒ Detailed restaurant profiles with reviews
 - ☒ Complete menu system with customization
 - ☒ Voice search functionality
 - ☒ Advanced filtering and sorting
 - ☒ Smooth navigation and performance
-

Task 3.3: Shopping Cart & Order Management

Priority: Critical | Estimated Time: 6-8 hours

- ☐ **Shopping Cart Implementation**
- ☐ Create `lib/features/cart/presentation/screens/`:
- ☐ `cart_screen.dart` - Shopping cart view
- ☐ `checkout_screen.dart` - Order checkout process
- ☐ `order_confirmation_screen.dart` - Order success
- ☐ `payment_screen.dart` - Payment processing
- ☐ **Cart Functionality**

- ☐ Add/remove items with animations
- ☐ Quantity adjustment with stepper controls
- ☐ Item customization preservation
- ☐ Price calculations (subtotal, taxes, fees)
- ☐ Delivery time estimation
- ☐ Minimum order value checking
- ☐ Cart persistence across app sessions
- ☐ Multiple restaurant handling
- ☐ **Checkout Process**
- ☐ Multi-step checkout with progress indicator
- ☐ Delivery address selection
- ☐ Delivery time slot selection
- ☐ Special instructions input
- ☐ Coupon/promo code application
- ☐ Order summary with itemized pricing
- ☐ Payment method selection
- ☐ Order placement confirmation
- ☐ **Order Tracking System**
- ☐ Create `lib/features/orders/presentation/screens/`:
- ☐ `order_tracking_screen.dart` - Live order tracking
- ☐ `order_history_screen.dart` - Past orders
- ☐ `order_details_screen.dart` - Order information
- ☐ `receipt_screen.dart` - Digital receipt
- ☐ **Real-time Order Updates**
- ☐ WebSocket connection for live updates
- ☐ Order status progression display
- ☐ Estimated delivery time updates
- ☐ Driver location tracking on map
- ☐ Push notifications for status changes
- ☐ Order modification/cancellation options

Acceptance Criteria:

- ☒ Complete shopping cart functionality
- ☒ Smooth checkout process
- ☒ Real-time order tracking
- ☒ Order history and reordering
- ☒ Payment integration working

- ☒ Push notifications for order updates
-

Phase 4: Advanced Features & Real-time Communication (Sprint 8-10)

Task 4.1: Payment System Integration

Priority: Critical | Estimated Time: 6-7 hours

☐ **Payment Method Management**

- ☐ Create payment method selection screen
- ☐ Implement Mobile Money integration:
 - ☐ MTN MoMo payment flow
 - ☐ Orange Money payment flow
 - ☐ EU Mobile payment support
- ☐ App wallet implementation
- ☐ Payment method storage and management
- ☐ Payment history and receipts

☐ **Payment Processing Flow**

- ☐ Payment amount calculation and display
- ☐ Payment method selection UI
- ☐ Payment confirmation screen
- ☐ Loading states during processing
- ☐ Success/failure feedback with animations
- ☐ Receipt generation and sharing
- ☐ Refund request functionality

☐ **Security & Validation**

- ☐ Input validation for payment forms
- ☐ Secure payment data handling
- ☐ Payment retry mechanism
- ☐ Transaction timeout handling
- ☐ Payment verification status
- ☐ Anti-fraud measures integration

Acceptance Criteria:

- ☒ Mobile Money payments working
- ☒ App wallet functionality
- ☒ Secure payment processing

- ☒ Payment history and receipts
 - ☒ Error handling and retry logic
 - ☒ Payment method management
-

Task 4.2: Push Notifications & Real-time Features

Priority: High | Estimated Time: 5-6 hours

☐ **Firestore Cloud Messaging Setup**

- ☐ Configure FCM for Android and iOS
- ☐ Implement notification permission handling
- ☐ Create notification service class
- ☐ Setup notification channels for Android
- ☐ Handle background and foreground notifications
- ☐ Implement notification tap handling

☐ **Real-time Notifications**

- ☐ Order status update notifications
- ☐ Driver assignment notifications
- ☐ Delivery proximity notifications
- ☐ Promotional notifications
- ☐ Payment confirmation notifications
- ☐ System maintenance notifications

☐ **WebSocket Integration**

- ☐ Setup Socket.IO client connection
- ☐ Implement connection management
- ☐ Handle reconnection logic
- ☐ Create event listeners for:
 - ☐ Order updates
 - ☐ Driver location updates
 - ☐ Chat messages
 - ☐ System announcements
- ☐ Implement connection status indicators

☐ **In-app Notification System**

- ☐ Notification center implementation
- ☐ Toast notifications for immediate feedback
- ☐ Badge counters for unread notifications
- ☐ Notification preferences management

- ☐ Notification history and archiving

Acceptance Criteria:

- ☒ Push notifications working on both platforms
 - ☒ Real-time order updates via WebSocket
 - ☒ In-app notification system
 - ☒ Notification preferences working
 - ☒ Connection management and error handling
-

Task 4.3: WhatsApp Integration & Communication

Priority: High | Estimated Time: 4-5 hours

☐ WhatsApp Business Integration

- ☐ Implement WhatsApp deep linking
- ☐ Create WhatsApp contact buttons
- ☐ Setup message templates for:
 - ☐ Order confirmations
 - ☐ Delivery updates
 - ☐ Customer support
 - ☐ Promotional messages
- ☐ Handle WhatsApp status (installed/not installed)

☐ Customer Support Chat

- ☐ In-app chat interface
- ☐ WhatsApp fallback option
- ☐ Chat history persistence
- ☐ File/image sharing capability
- ☐ Support agent availability status
- ☐ Quick reply templates

☐ Communication Preferences

- ☐ Communication channel selection
- ☐ Opt-in/opt-out for different message types
- ☐ Do not disturb settings
- ☐ Language preferences for messages
- ☐ Message frequency controls

Acceptance Criteria:

- ☒ WhatsApp integration working
 - ☒ In-app chat functionality
 - ☒ Message templates implemented
 - ☒ Communication preferences
 - ☒ Support system functional
-

Phase 5: Role-specific Interfaces (Sprint 11-13)



Task 5.1: Restaurant Owner Dashboard

Priority: High | Estimated Time: 8-10 hours

☐ **Restaurant Dashboard Main Screen**

- ☐ Create restaurant owner main dashboard with:
- ☐ Today's statistics cards (orders, revenue, rating)
- ☐ Live orders section with real-time updates
- ☐ Quick action buttons (menu, notifications, settings)
- ☐ Performance charts and graphs
- ☐ Recent customer reviews
- ☐ Inventory alerts

☐ **Order Management System**

- ☐ Live order notifications with sound alerts
- ☐ Order acceptance/rejection interface
- ☐ Preparation time estimation tools
- ☐ Order status update controls
- ☐ Customer communication interface
- ☐ Order details and special instructions view
- ☐ Batch order processing capabilities

☐ **Menu Management Interface**

- ☐ Menu category management
- ☐ Menu item CRUD operations
- ☐ Image upload and management
- ☐ Pricing and availability controls
- ☐ Bulk operations (enable/disable items)
- ☐ Menu analytics and performance
- ☐ Special offers and discounts setup

☐ **Restaurant Analytics Dashboard**

- ☐ Sales performance charts
- ☐ Popular items analysis
- ☐ Customer behavior insights
- ☐ Peak hours identification
- ☐ Revenue trending
- ☐ Comparison with previous periods
- ☐ Export reports functionality
- ☐ **Restaurant Profile Management**
- ☐ Restaurant information editing
- ☐ Business hours management
- ☐ Delivery radius configuration
- ☐ Contact information updates
- ☐ Photo gallery management
- ☐ Cuisine tags and categories
- ☐ Restaurant status (open/closed/busy)

Acceptance Criteria:

- ☒ Complete restaurant owner dashboard
 - ☒ Real-time order management
 - ☒ Menu management functionality
 - ☒ Analytics and reporting
 - ☒ Restaurant profile management
 - ☒ Mobile-responsive design
-

Task 5.2: Driver/Delivery Agent Interface

Priority: High | Estimated Time: 6-8 hours

- ☐ **Driver Dashboard Implementation**
- ☐ Driver status toggle (online/offline)
- ☐ Available orders list with details
- ☐ Earnings tracker (daily/weekly/monthly)
- ☐ Performance metrics display
- ☐ Navigation integration
- ☐ Emergency contact features
- ☐ **Order Assignment & Management**
- ☐ Smart order matching notifications

- ☐ Order acceptance/rejection interface
- ☐ Multiple order batching capability
- ☐ Customer contact information
- ☐ Delivery instructions display
- ☐ Special delivery requirements
- ☐ Order cancellation handling
- ☐ **Navigation & Tracking**
- ☐ Google Maps integration
- ☐ Turn-by-turn navigation
- ☐ Real-time location sharing
- ☐ Multiple stop optimization
- ☐ Traffic-aware routing
- ☐ Arrival time estimation
- ☐ Location accuracy indicators
- ☐ **Delivery Completion System**
- ☐ Photo capture for proof of delivery
- ☐ Digital signature collection
- ☐ OTP verification from customers
- ☐ Delivery confirmation process
- ☐ Customer rating system
- ☐ Issue reporting functionality
- ☐ Cash collection tracking
- ☐ **Driver Performance & Earnings**
- ☐ Real-time earnings calculation
- ☐ Performance rating display
- ☐ Delivery statistics
- ☐ Bonus and incentive tracking
- ☐ Payout request functionality
- ☐ Performance improvement tips
- ☐ Achievement badges system

Acceptance Criteria:

- ☒ Complete driver interface functional
- ☒ Order management system working
- ☒ Navigation and tracking accurate
- ☒ Delivery proof system implemented
- ☒ Earnings and performance tracking

- ☒ Mobile-optimized for on-the-go use
-

Task 5.3: Admin Dashboard & Management

Priority: Medium | Estimated Time: 10-12 hours

☐ **Admin Dashboard Overview**

- ☐ Multi-metric overview cards
- ☐ Real-time system health monitoring
- ☐ Key performance indicators (KPIs)
- ☐ Recent activity feed
- ☐ Quick action shortcuts
- ☐ System alerts and notifications
- ☐ Financial overview summary

☐ **User Management System**

- ☐ User list with search and filtering
- ☐ User profile viewing and editing
- ☐ Account status management (active/suspended/banned)
- ☐ Role assignment and permissions
- ☐ Bulk user operations
- ☐ User activity monitoring
- ☐ Account verification tools

☐ **Restaurant Management**

- ☐ Restaurant onboarding approval process
- ☐ Restaurant profile verification
- ☐ Commission rate management
- ☐ Restaurant performance monitoring
- ☐ Menu audit and compliance
- ☐ Restaurant suspension/activation
- ☐ Bulk restaurant operations

☐ **Financial Management Dashboard**

- ☐ Revenue tracking and reporting
- ☐ Commission management
- ☐ Payout processing interface
- ☐ Transaction monitoring
- ☐ Refund management system
- ☐ Financial analytics and insights
- ☐ Tax reporting tools

☐ **Marketing Campaign Management**

- ☐ Campaign creation and scheduling
- ☐ Target audience selection
- ☐ Multi-channel campaign coordination
- ☐ A/B testing setup
- ☐ Campaign performance tracking
- ☐ ROI analysis and reporting
- ☐ Automated campaign triggers

☐ **WhatsApp Business Management**

- ☐ Template message management
- ☐ Bulk messaging interface
- ☐ WhatsApp analytics dashboard
- ☐ Contact list management
- ☐ Broadcast list creation
- ☐ Message scheduling system
- ☐ Opt-in/opt-out management

☐ **System Analytics & Reporting**

- ☐ Comprehensive analytics dashboard
- ☐ Custom report builder
- ☐ Data export capabilities
- ☐ Performance metrics tracking
- ☐ User behavior analysis
- ☐ Business intelligence insights
- ☐ Automated reporting system

Acceptance Criteria:

- ☒ Complete admin dashboard functional
 - ☒ User and restaurant management working
 - ☒ Financial management system
 - ☒ Marketing campaign tools
 - ☒ WhatsApp business management
 - ☒ Comprehensive analytics and reporting
-

Phase 6: Advanced Features & Optimization (Sprint 14-16)

Task 6.1: AI Features & Personalization

Priority: Medium | Estimated Time: 6-8 hours

☐ **AI-Powered Recommendations**

- ☐ Implement recommendation algorithm
- ☐ Personalized restaurant suggestions
- ☐ Menu item recommendations
- ☐ Time-based recommendations
- ☐ Weather-based suggestions
- ☐ Collaborative filtering implementation
- ☐ Content-based filtering

☐ **Smart Search Features**

- ☐ Natural language search processing
- ☐ Image-based food search
- ☐ Voice command processing
- ☐ Search result optimization
- ☐ Query suggestion improvements
- ☐ Search history analysis
- ☐ Personalized search results

☐ **Intelligent Notifications**

- ☐ Smart notification timing
- ☐ Personalized notification content
- ☐ Notification frequency optimization
- ☐ Behavioral trigger notifications
- ☐ Predictive order suggestions
- ☐ Customer retention notifications
- ☐ Re-engagement campaigns

Acceptance Criteria:

- ☒ AI recommendations working accurately
 - ☒ Smart search functionality implemented
 - ☒ Intelligent notification system
 - ☒ Personalization improving user experience
 - ☒ Performance optimized for AI features
-

Task 6.2: Analytics & Performance Monitoring

Priority: Medium | Estimated Time: 4-5 hours

☐ **User Analytics Integration**

- ☐ Firebase Analytics setup
- ☐ Custom event tracking
- ☐ User journey mapping
- ☐ Conversion funnel analysis
- ☐ Retention rate tracking
- ☐ Engagement metrics
- ☐ Performance monitoring

☐ **App Performance Monitoring**

- ☐ Crash reporting integration (Crashlytics)
- ☐ Performance metrics tracking
- ☐ Network request monitoring
- ☐ Memory usage optimization
- ☐ Battery usage optimization
- ☐ App startup time optimization
- ☐ Frame rate monitoring

☐ **Business Analytics Dashboard**

- ☐ Real-time business metrics
- ☐ Revenue tracking
- ☐ Order completion rates
- ☐ Customer satisfaction scores
- ☐ Driver performance metrics
- ☐ Restaurant performance tracking
- ☐ Marketing campaign effectiveness

Acceptance Criteria:

- ☒ Analytics tracking implemented
 - ☒ Performance monitoring working
 - ☒ Business metrics dashboard
 - ☒ Crash reporting functional
 - ☒ Optimization recommendations available
-

Task 6.3: Security & Privacy Features

Priority: High | Estimated Time: 5-6 hours

☐ **Security Implementation**

- ☐ Biometric authentication
- ☐ PIN/Pattern lock for app
- ☐ Session timeout management
- ☐ Data encryption at rest
- ☐ Secure API communications
- ☐ Certificate pinning
- ☐ Anti-tampering measures

☐ **Privacy Controls**

- ☐ Privacy settings interface
- ☐ Data sharing preferences
- ☐ Location privacy controls
- ☐ Marketing communication preferences
- ☐ Data deletion requests
- ☐ Privacy policy display
- ☐ GDPR compliance features

☐ **Security Monitoring**

- ☐ Suspicious activity detection
- ☐ Failed login attempt monitoring
- ☐ Device fingerprinting
- ☐ Fraud detection alerts
- ☐ Security audit logging
- ☐ Incident response procedures
- ☐ Security update mechanisms

Acceptance Criteria:

- ☒ Biometric authentication working
 - ☒ Privacy controls implemented
 - ☒ Data encryption in place
 - ☒ Security monitoring active
 - ☒ GDPR compliance features
 - ☒ Anti-fraud measures working
-

Phase 7: Testing, Optimization & Deployment (Sprint 17-18)

Task 7.1: Comprehensive Testing Implementation

Priority: Critical | Estimated Time: 8-10 hours

☐ **Unit Testing**

- ☐ Test all utility functions
- ☐ Test data models and serialization
- ☐ Test business logic functions
- ☐ Test validation methods
- ☐ Test state management providers
- ☐ Test API service methods
- ☐ Achieve 90%+ code coverage

☐ **Widget Testing**

- ☐ Test all custom widgets
- ☐ Test form validation
- ☐ Test navigation flows
- ☐ Test user interactions
- ☐ Test animation states
- ☐ Test loading and error states
- ☐ Test responsive layouts

☐ **Integration Testing**

- ☐ Test complete user journeys
- ☐ Test authentication flows
- ☐ Test order placement process
- ☐ Test payment processing
- ☐ Test real-time features
- ☐ Test offline functionality
- ☐ Test cross-platform compatibility

☐ **Performance Testing**

- ☐ Memory leak testing
- ☐ Battery usage testing
- ☐ Network efficiency testing
- ☐ Startup time optimization
- ☐ Scroll performance testing
- ☐ Image loading optimization
- ☐ Database query optimization

☐ **Accessibility Testing**

- ☐ Screen reader compatibility
- ☐ Color contrast validation
- ☐ Touch target size validation
- ☐ Keyboard navigation testing
- ☐ Font scaling testing
- ☐ Voice control testing
- ☐ WCAG compliance checking

Acceptance Criteria:

- ☒ 90%+ unit test coverage
 - ☒ All widget tests passing
 - ☒ Integration tests covering main flows
 - ☒ Performance benchmarks met
 - ☒ Accessibility standards compliance
 - ☒ Cross-platform testing completed
-

Task 7.2: Performance Optimization & Polish

Priority: High | Estimated Time: 6-7 hours

☐ App Performance Optimization

- ☐ Image optimization and caching
- ☐ Lazy loading implementation
- ☐ Memory management improvements
- ☐ Network request optimization
- ☐ Database query optimization
- ☐ Animation performance tuning
- ☐ Bundle size optimization

☐ User Experience Polish

- ☐ Micro-interaction refinements
- ☐ Loading state improvements
- ☐ Error message enhancements
- ☐ Navigation flow optimization
- ☐ Gesture handling improvements
- ☐ Haptic feedback integration
- ☐ Sound effect integration

☐ Offline Experience Enhancement

- ☐ Offline data caching strategy
- ☐ Offline mode UI indicators
- ☐ Data synchronization logic
- ☐ Conflict resolution handling
- ☐ Progressive web app features
- ☐ Background sync implementation
- ☐ Offline error handling
- ☐ **Accessibility Enhancements**
- ☐ Voice-over support optimization
- ☐ High contrast mode support
- ☐ Large text support
- ☐ Reduced motion support
- ☐ Color blind friendly design
- ☐ Keyboard navigation improvement
- ☐ Semantic labeling enhancement

Acceptance Criteria:

- ☒ App startup time under 3 seconds
 - ☒ Smooth scrolling on all screens
 - ☒ Offline functionality working
 - ☒ Accessibility standards met
 - ☒ Battery usage optimized
 - ☒ Memory usage under control
-

Task 7.3: Deployment Preparation & App Store Submission

Priority: Critical | Estimated Time: 4-5 hours

- ☐ **Build Configuration**
- ☐ Release build configuration
- ☐ App signing setup (Android & iOS)
- ☐ ProGuard/R8 configuration
- ☐ App bundle optimization
- ☐ Security key management
- ☐ Environment configuration
- ☐ Build scripts automation
- ☐ **App Store Assets Creation**

- ☐ App icon design (multiple sizes)
- ☐ App screenshots (all device sizes)
- ☐ App store description writing
- ☐ Feature list compilation
- ☐ Privacy policy creation
- ☐ Terms of service creation
- ☐ App demo video creation
- ☐ **Platform-specific Preparations**
- ☐ **Android (Google Play Store):**
 - ☐ App bundle generation
 - ☐ Google Play Console setup
 - ☐ Store listing optimization
 - ☐ Content rating completion
 - ☐ Release management setup
 - ☐ Beta testing group creation
 - ☐ App signing key management
- ☐ **iOS (Apple App Store):**
 - ☐ Xcode project configuration
 - ☐ App Store Connect setup
 - ☐ TestFlight beta testing
 - ☐ App review guidelines compliance
 - ☐ App Store optimization
 - ☐ iOS-specific feature validation
 - ☐ Provisioning profile management
- ☐ **Quality Assurance Final Check**
 - ☐ Complete app functionality testing
 - ☐ Cross-device compatibility testing
 - ☐ Store policy compliance review
 - ☐ Legal requirements verification
 - ☐ Privacy policy compliance
 - ☐ Security audit completion
 - ☐ Performance benchmarking
- ☐ **Launch Preparation**
 - ☐ Marketing material preparation
 - ☐ Press release creation
 - ☐ Social media content preparation
 - ☐ Customer support preparation
 - ☐ App analytics setup

- ☐ Crash reporting configuration
- ☐ Update rollout strategy

Acceptance Criteria:

- ☒ Release builds generated successfully
 - ☒ App store assets completed
 - ☒ Both platforms ready for submission
 - ☒ Quality assurance passed
 - ☒ Legal compliance verified
 - ☒ Launch materials prepared
-



Success Metrics & KPIs

Technical Metrics

- ☐ **Performance Benchmarks:**
- ☐ App startup time < 3 seconds (including splash screen)
- ☐ Screen transition time < 300ms
- ☐ API response handling < 200ms
- ☐ Memory usage < 150MB under normal load
- ☐ Battery drain < 5% per hour of active use
- ☐ Crash rate < 1% of sessions
- ☐ **Quality Metrics:**
- ☐ Code coverage > 90%
- ☐ Zero critical security vulnerabilities
- ☐ Accessibility score > 95%
- ☐ App store rating target > 4.5 stars
- ☐ User retention rate > 70% (30 days)
- ☐ Onboarding completion rate > 85%

Business Metrics

- ☐ **User Engagement:**
- ☐ Daily active users growth > 10% monthly
- ☐ Session duration > 5 minutes average
- ☐ Order completion rate > 85%
- ☐ User onboarding completion > 80%

- ☐ Feature adoption rate > 60%
 - ☐ Terms and conditions acceptance rate > 95%
 - ☐ About Us page engagement rate > 25%
-

Post-Launch Maintenance Tasks

Immediate Post-Launch (First 2 weeks)

- ☐ Monitor app performance metrics
- ☐ Track crash reports and fix critical issues
- ☐ Monitor user feedback and reviews
- ☐ Track user onboarding funnel
- ☐ Monitor API performance and errors
- ☐ Collect user behavior analytics
- ☐ Address urgent user support tickets

Short-term Improvements (1-3 months)

- ☐ Implement user feedback suggestions
- ☐ Optimize performance based on real usage data
- ☐ Add missing features identified by users
- ☐ Improve onboarding based on drop-off analysis
- ☐ Enhance search and discovery features
- ☐ Optimize notification strategies
- ☐ Expand payment method options

Medium-term Enhancements (3-6 months)

- ☐ Advanced AI recommendation improvements
 - ☐ Social features implementation
 - ☐ Loyalty program enhancements
 - ☐ Multi-language support expansion
 - ☐ Advanced analytics dashboard
 - ☐ Integration with third-party services
 - ☐ Performance optimization based on scale
-

This comprehensive TODO list provides AI agents with clear, actionable tasks for implementing a world-class Flutter food delivery application. Each task includes specific deliverables, acceptance criteria, and estimated time requirements to ensure systematic development and quality outcomes.

Total Estimated Development Time: 135-165 hours Recommended Team Size: 2-3 Flutter developers working in parallel Development Duration: 18 sprints (20-22 weeks)

The tasks are designed to be executed incrementally, allowing for continuous testing, feedback integration, and iterative improvements throughout the development process. The app launch experience creates a strong first impression with professional branding and company valorization that builds trust with users from the very first interaction.