📱 PROJECT REQUIREMENTS DOCUMENTATION
TikTok-Style Restaurant Discovery Mobile Application

Document Type: Complete Functional & Technical Specifications
Target Audience: Development Company / Software Agency

📋 TABLE OF CONTENTS

---

1. EXECUTIVE SUMMARY

1.1 Project Description

A mobile application that allows users to discover restaurants and cafes through a **TikTok-style vertical video feed**. Users scroll through short video content showcasing different places, can like, save, filter, and access detailed information about each location.

1.2 Key Features

- **Vertical Video Feed** with autoplay (TikTok/Reels-style)
- **Location-based Discovery** with map integration
- **Advanced Filtering System** (cuisine, price, distance, rating…)
- **Social Features** (like, save, share)
- **User Authentication** (Apple Sign In, Google Sign In)
- **Offline Support** with local caching

1.3 Target Platforms

- **iOS** (iPhone, iPad)
- **Android** (Phone, Tablet)

1.4 Technology Stack

- **Framework:** React Native with Expo
- **Backend:** Supabase (PostgreSQL + Real-time)
- **State Management:** Zustand
- **Navigation:** React Navigation
- **Video Player:** expo-video
- **Maps:** react-native-maps

---

2. PROJECT OVERVIEW

2.1 Business Objectives

1. **User Engagement:** Create an addictive, scrollable video experience
2. **Discovery:** Help users find restaurants/cafes based on preferences
3. **Social Sharing:** Enable users to share favorite places

2.2 Success Metrics

- User retention rate (Day 1, Day 7, Day 30)
- Average session duration
- Number of places viewed per session
- Filter usage rate
- Share/like engagement rate

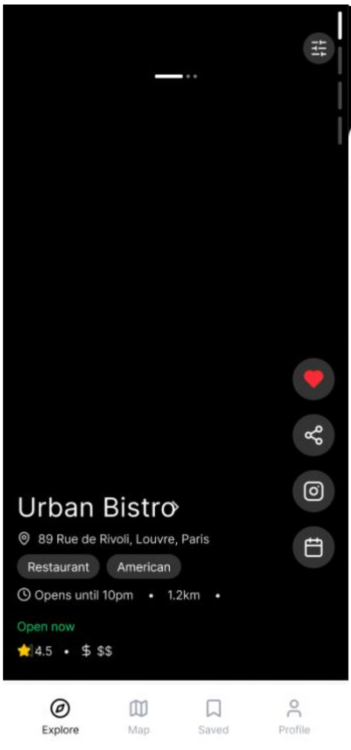
2.3 Target Users

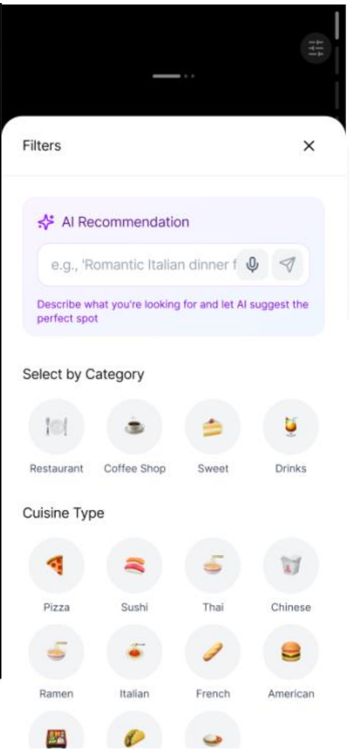
- **Primary:** Food enthusiasts aged 18-45
- **Secondary:** Tourists looking for local recommendations
- **Tertiary:** Social media users who enjoy video content

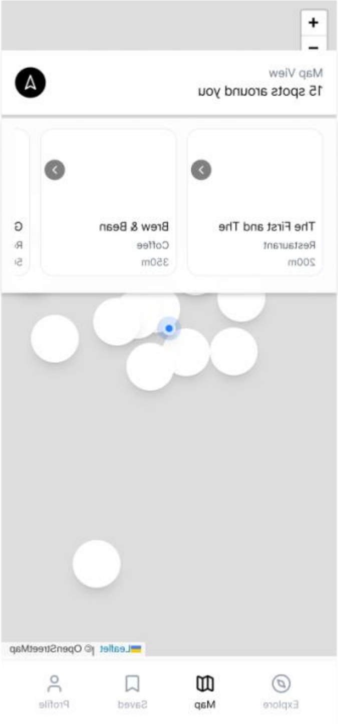
---

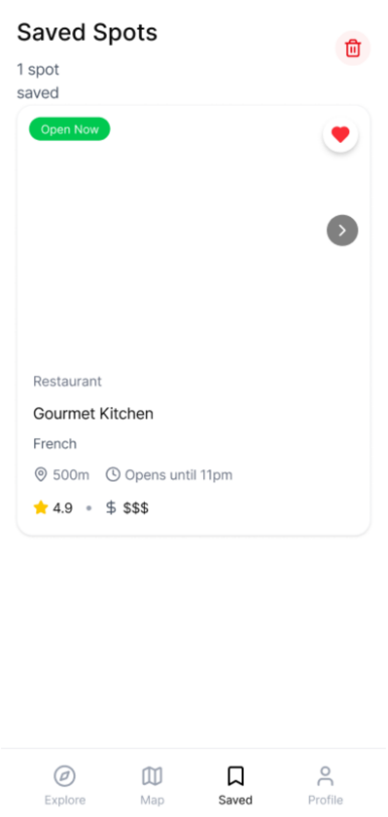
3. FUNCTIONAL REQUIREMENTS

3.1 Authentication & User Management


Urban Bistro
89 Rue de Rivoli, Louvre, Paris
Restaurant
American
Opens until 10pm • 1.2km •
Open now
4.5 • $ $$
Explore
Map
Saved
Profile


Filters
AI Recommendation
e.g., 'Romantic Italian dinner f
Describe what you're looking for and let AI suggest the perfect spot
Select by Category
Restaurant
Coffee Shop
Sweet
Drinks
Cuisine Type
Pizza
Sushi
Thai
Chinese
Ramen
Italian
French
American


+
−
Map View
15 spots around you
The First and The
Restaurant
200m
Brew & Bean
Coffee
350m
Leaflet | © OpenStreetMap
Profile
Saved
Map
Explore



Saved Spots

1 spot
saved

Open Now

Restaurant

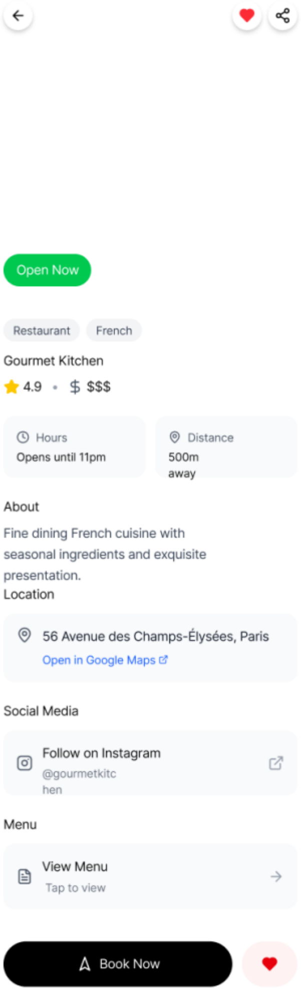
Gourmet Kitchen

French

500m Opens until 11pm

★ 4.9 • $ $$$

Explore Map Saved Profile



Open Now

Restaurant French

Gourmet Kitchen

★ 4.9 • $ $$$

Hours
Opens until 11pm

Distance
500m
away

About

Fine dining French cuisine with seasonal ingredients and exquisite presentation.

Location

56 Avenue des Champs-Élysées, Paris
Open in Google Maps

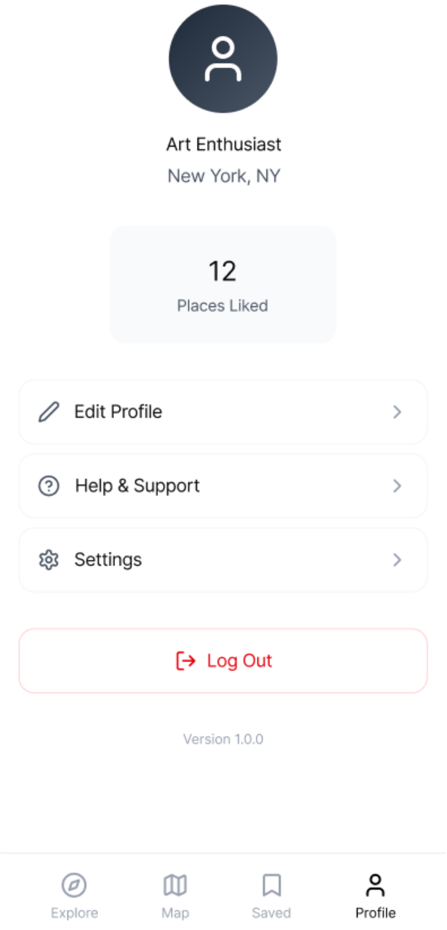
Social Media

Follow on Instagram
@gourmetkitc
hen

Menu

View Menu
Tap to view

Book Now



Art Enthusiast

New York, NY

12
Places Liked

Edit Profile

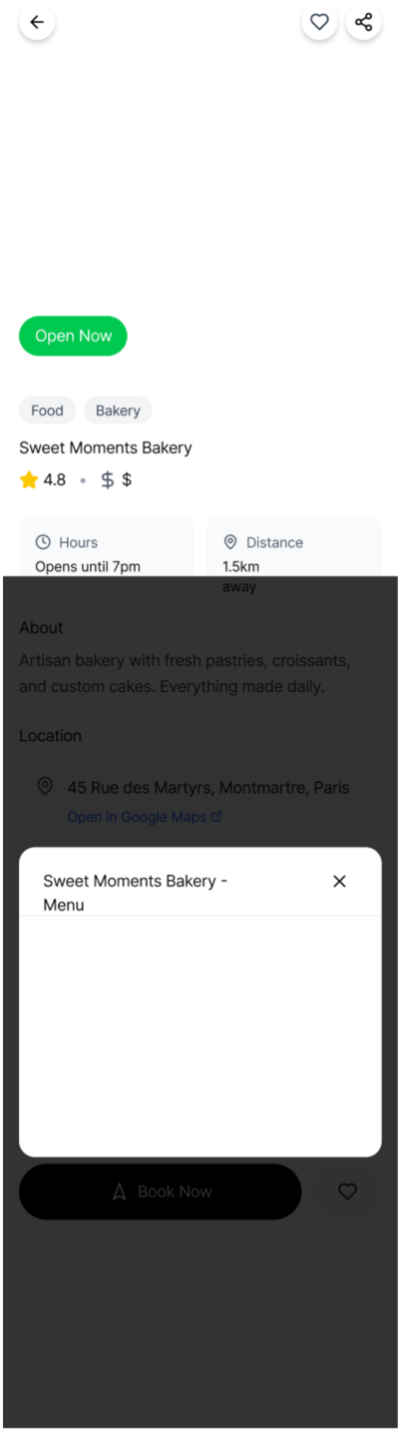
Help & Support

Settings

Log Out

Version 1.0.0

Explore Map Saved Profile



Artisan bakery with fresh pastries, croissants, and custom cakes. Everything made daily.

45 Rue des Martyrs, Montmartre, Paris

3.1.1 Login Screen

Requirements:
- Apple Sign In (required for iOS)
- Google Sign In (optional)
- Email/Password (optional, future phase)
- Guest mode (optional, limited features)

User Flow:
1. App launches → Check if user is logged in
2. If not logged in → Show Login Screen
3. User selects authentication method
4. After successful auth → Navigate to Feed Screen

Acceptance Criteria:
- ✅ Seamless authentication flow
- ✅ User session persists across app restarts
- ✅ Logout functionality available
- ✅ Error handling for failed authentication

3.1.2 User Profile

Requirements:
- Display user information (name, email, avatar)
- Edit profile (future phase)
- View favorite places count
- Logout option

3.2 Feed Screen (Main Feature)

3.2.1 Video Feed

Core Functionality:
- **Vertical Scroll:** Infinite scroll through video content
- **Autoplay:** Video automatically plays when visible (>50% viewport)
- **Single Active Video:** Only one video plays at a time
- **Auto-pause:** Videos pause when scrolled out of view
- **Loop:** Videos loop continuously
- **Mute/Unmute:** Active video has sound, others are muted

Technical Requirements:
- Smooth scrolling (60 FPS)
- Preload next 2-3 videos for seamless playback
- Memory management (unload videos far from viewport)
- Handle video loading errors gracefully

User Interactions:
- **Swipe Up:** Next video
- **Swipe Down:** Previous video
- **Tap Video:** Pause/Play toggle
- **Swipe Left:** Open image gallery
- **Long Press:** Show quick actions menu (future)

3.2.2 Place Information Overlay

Displayed Information:
- Place name (large, bold)
- Address (with distance if location enabled)
- Description (2 lines max, expandable)
- Phone number (tappable)
- Rating (stars)
- Price range (€, €€, €€€, €€€€)

Positioning:
- Bottom of screen
- Above tab bar
- Semi-transparent background for readability

3.2.3 Action Buttons

Right Side Actions (Vertical Stack):
1. **Dislike** (thumbs down) - Hide this place
2. **Like** (heart) - Add to favorites
3. **Share** (share icon) - Share place
4. **Instagram** (Instagram logo) - Open Instagram profile

Behavior:
- Icons with semi-transparent background
- Animated feedback on tap
- Like button changes color when favorited

3.2.4 Filter Button

Location: Top right corner
Functionality:
- Opens filter modal
- Shows active filter count badge
- Quick reset option

3.3 Filter System

3.3.1 Filter Modal

Available Filters:

1. **Place Type**
   - Restaurant
   - Café
   - Multi-select

2. **Cuisine Type**
   - Indian

   - Italian
   - Japanese
   - Korean
   - Lebanese
   - Iranian
   - Moroccan
   - Thai
   - French
   - Multi-select

3. **Distance**
   - Slider: 0-10 km
   - Based on user location
   - Default: 5 km

4. **Price Range**
   - € (Budget)
   - €€ (Moderate)
   - €€€ (Expensive)
   - €€€€ (Very Expensive)
   - Multi-select

5. **Rating**
   - Slider: 0-5 stars
   - Minimum rating filter
   - Default: 0 (all ratings)

6. **Availability** (Future Phase)
   - Party size (number of people)
   - Date picker
   - Time range (start/end)

7. **Transport Mode** (Future Phase)
   - Walk
   - Car
   - Transit

UI Components:
- Checkboxes for multi-select options
- Sliders for numeric values
- Date/Time pickers for availability
- "Apply" button (applies filters and closes modal)
- "Reset" button (clears all filters)
- "Cancel" button (closes without applying)

Behavior:
- Filters persist in local storage

- Applied filters show count badge on filter button
- Filter modal can be opened from Feed Screen
- First-time users see filter screen on app launch

3.4 Map Screen

3.4.1 Map Display

Requirements:
- Interactive map (Google Maps / Apple Maps)
- Display all places as pins/markers
- User location marker (if permission granted)
- Clustering for nearby places (if many markers)

Map Controls:
- Zoom in/out
- Pan to explore
- Center on user location button
- Filter toggle (show/hide filtered places)

3.4.2 Place Markers

Marker Design:
- Custom pin with place type icon
- Selected marker highlighted
- Show place name on tap

Interactions:
- Tap marker → Show place info card
- Info card shows:
  - Place name
  - Address
  - Rating
  - Price range
  - "View Details" button
  - "Get Directions" button

3.5 Favorites Screen

3.5.1 Favorites List

Display:
- Grid or list view of favorited places
- Place card shows:
  - Thumbnail image
  - Place name
  - Address

  - Rating
  - Price range

Interactions:
- Tap place → Navigate to Feed and scroll to that place
- Long press → Remove from favorites
- Swipe to delete (optional)

Empty State:
- Message: "No favorites yet"
- CTA: "Start exploring"

3.6 Place Detail Screen

3.6.1 Detail View

Sections:

1. **Image Gallery**
   - Horizontal scrollable gallery
   - Full-screen view on tap
   - Video thumbnail (if available)

2. **Place Information**
   - Name (large)
   - Description (full text)
   - Address (with map link)
   - Phone (tappable)
   - Website (tappable)
   - Opening hours (if available)
   - Instagram link

3. **Actions**
   - Like/Unlike button
   - Share button
   - Get Directions button
   - Call button

4. **Additional Info**
   - Rating breakdown
   - Price range
   - Cuisine type
   - Place type

3.7 Account Screen

3.7.1 Account Management

Display:
- User avatar
- User name
- Email
- Number of favorites
- App version

Actions:
- Edit profile (future)
- Settings (future)
- Logout button
- Delete account (future)

---

4. TECHNICAL REQUIREMENTS

4.1 Technology Stack

4.1.1 Frontend Framework

React Native with Expo
- **Version:** Expo SDK 54
- **React:** 18.2.0
- **React Native:** 0.81.5
- **TypeScript:** 5.9.2

Rationale:
- Cross-platform development (iOS + Android)
- Fast development cycle
- Over-the-air updates
- Native performance

4.1.2 Key Dependencies

```json
{
  "expo": "54.0.20",
  "react": "18.2.0",
  "react-dom": "18.2.0",
  "expo-video": "~3.0.15",
  "@react-navigation/native": "^7.1.17",
  "@react-navigation/bottom-tabs": "^7.4.7",
  "@react-navigation/stack": "^7.4.8",
  "zustand": "^5.0.8",
  "@supabase/supabase-js": "^2.45.4",
```

  "expo-location": "^19.0.7",
  "react-native-maps": "1.20.1",
  "expo-apple-authentication": "~8.0.7",
  "expo-auth-session": "~7.0.8",
  "@react-native-async-storage/async-storage": "^2.2.0"
}
```

#### 4.1.3 State Management

**Zustand Stores:**
- `authStore` - User authentication state
- `favoritesStore` - User favorites (with persistence)
- `filtersStore` - Active filters (with persistence)
- `hiddenStore` - Hidden places

**Persistence:**
- AsyncStorage for local persistence
- Supabase sync for favorites (when authenticated)

### 4.2 Architecture

#### 4.2.1 Folder Structure

```
src/
 ├── api/   # API clients & services
 │   ├── client.ts   # Supabase client
 │   ├── placesService.ts # Places API service
 │   └── env.ts   # Environment variables
 ├── assets/   # Static assets
 │   ├── images/   # Image assets
 │   └── videos/   # Video assets (local)
 ├── components/   # Reusable components
 │   ├── VideoPlayerFullScreen.tsx
 │   ├── ImageGallery.tsx
 │   ├── FilterModal.tsx
 │   └── PlacePin.tsx
 ├── hooks/   # Custom React hooks
 │   ├── useVideoPlayback.ts # Video playback logic
 │   ├── usePlaces.ts   # Places data fetching
 │   └── useFilters.ts   # Filter logic
 ├── navigation/   # Navigation setup
 │   └── AppNavigator.tsx
 ├── screens/   # Screen components
 │   ├── FeedScreen.tsx

```
│   ├── MapScreen.tsx
│   ├── FavoritesScreen.tsx
│   ├── PlaceDetailScreen.tsx
│   ├── LoginScreen.tsx
│   └── AccountScreen.tsx
├── store/              # Zustand stores
│   ├── authStore.ts
│   ├── favoritesStore.ts
│   ├── filtersStore.ts
│   └── hiddenStore.ts
├── types/              # TypeScript types
│   └── index.ts
└── utils/              # Utility functions
    ├── cache.ts
    ├── videoPreloader.ts
    └── logger.ts
```

4.2.2 Video Playback Architecture

Critical Component: `useVideoPlayback` Hook

Requirements:
- ✅ Player created ONLY with valid source
- ✅ NO `player` in useEffect dependencies (prevents infinite loops)
- ✅ Unique key for VideoView (forces re-creation on source change)
- ✅ Proper cleanup on unmount
- ✅ Error handling with retry mechanism

Implementation:
See `GUIDE_USEVIDEOPLAYER_EXPO_GO.md` for complete implementation details.

4.3 Performance Requirements

4.3.1 Video Performance

- **Frame Rate:** 60 FPS during scroll
- **Video Loading:** Preload next 2-3 videos
- **Memory Management:** Unload videos >3 positions away
- **Autoplay Delay:** < 200ms after video becomes visible

4.3.2 App Performance

- **Cold Start:** < 2 seconds
- **Screen Transitions:** < 300ms
- **Filter Application:** < 500ms
- **Map Rendering:** < 1 second

4.3.3 Network Optimization

- **Image Caching:** Cache images locally
- **Video Caching:** Cache video thumbnails
- **API Caching:** Cache places data (TTL: 10 minutes)
- **Offline Support:** Show cached data when offline

4.4 Security Requirements

4.4.1 Authentication

- Secure token storage
- Token refresh mechanism
- Session timeout handling

4.4.2 Data Protection

- Encrypt sensitive data in AsyncStorage
- Secure API communication (HTTPS)
- User data privacy compliance (GDPR)

4.5 Platform-Specific Requirements

4.5.1 iOS

- **Minimum Version:** iOS 13.0
- **Permissions:**
  - Location (When In Use)
  - Camera (for future features)
  - Photo Library (for sharing)
- **App Store Guidelines:**
  - Privacy policy required
  - Age rating: 4+ (or appropriate)
  - Content guidelines compliance

4.5.2 Android

- **Minimum Version:** Android 8.0 (API 26)
- **Permissions:**
  - ACCESS_FINE_LOCATION
  - ACCESS_COARSE_LOCATION
  - READ_EXTERNAL_STORAGE (for Android < 13)
  - READ_MEDIA_IMAGES (for Android 13+)
- **Play Store Guidelines:**
  - Privacy policy required
  - Content rating

  - Target SDK: Latest

---

5. USER FLOW & EXPERIENCE

5.1 First-Time User Flow

```
1. App Launch
   ↓
2. Splash Screen (1.5 seconds)
   ↓
3. Login Screen
   - Apple Sign In button
   - Google Sign In button
   ↓
4. Authentication
   ↓
5. Filter Screen (First Time)
   - Welcome message
   - Filter options
   - "Start Exploring" button
   ↓
6. Feed Screen
   - First video autoplays
   - Tutorial overlay (optional)
```

5.2 Returning User Flow

```
1. App Launch
   ↓
2. Splash Screen
   ↓
3. Check Authentication
   - If logged in → Feed Screen
   - If not logged in → Login Screen
   ↓
4. Feed Screen
   - Resume from last position (optional)
   - Videos autoplay
```

5.3 Feed Interaction Flow

```
User on Feed Screen
  ↓
Scroll Up/Down
  ↓
Video becomes visible (>50%)
  ↓
Autoplay starts
  ↓
User sees place info
  ↓
User can:
  - Like (adds to favorites)
  - Share (opens share sheet)
  - Dislike (hides place)
  - Tap info (opens details)
  - Swipe left (opens gallery)
  - Tap filter (opens filters)
```

5.4 Filter Flow

```
User taps Filter button
  ↓
Filter Modal opens
  ↓
User selects filters
  ↓
User taps "Apply"
  ↓
Modal closes
  ↓
Feed updates with filtered results
  ↓
Filter badge shows count
```

5.5 Navigation Flow

```
Bottom Tab Bar:
  - Feed (default)
  - Map
  - Favorites
  - Account
```

From Feed:
   - Tap place info → Place Detail
   - Swipe left → Image Gallery

From Map:
   - Tap marker → Place Info Card
   - Tap "Details" → Place Detail

From Favorites:
   - Tap place → Navigate to Feed (scroll to place)
   - Tap place → Place Detail
```

---

## 6. UI/UX DESIGN SPECIFICATIONS

### 6.1 Design System

#### 6.1.1 Colors

**Primary:**
- Background: `#000000` (Black)
- Text: `#FFFFFF` (White)
- Accent: `#FF4757` (Red for likes)

**Secondary:**
- Overlay: `rgba(0, 0, 0, 0.6)`
- Tab Bar: `#FFFFFF`
- Border: `#E9ECEF`

#### 6.1.2 Typography

**Fonts:**
- Primary: System font (SF Pro on iOS, Roboto on Android)
- Sizes:
  - Title: 24px, Bold
  - Subtitle: 18px, Semi-bold
  - Body: 16px, Regular
  - Caption: 14px, Regular

#### 6.1.3 Spacing

- Small: 8px
- Medium: 16px
- Large: 24px
- XLarge: 32px
```

6.1.4 Components

Buttons:
- Height: 50px
- Border radius: 25px
- Padding: 16px horizontal

Icons:
- Size: 30px (large), 22px (small)
- Color: White with semi-transparent background

6.2 Screen Mockups

6.2.1 Feed Screen

Layout:
```
┌─────────────────────────────┐
│                             │
│                             │
│      VIDEO PLAYER           │
│      (Full Screen)          │
│                             │
│                             │
│                             │
│  [Filter]                   │
│                             │
│  Place Name                 │
│  Address • Phone            │
│  Description...             │
│                             │
│          [Dislike]          │
│          [Like]             │
│          [Share]            │
│          [Instagram]        │
└─────────────────────────────┘
   [Feed] [Map] [❤] [👤]
```

6.2.2 Filter Modal

Layout:
```
┌─────────────────────────────┐
│  Filters         [X]        │
├─────────────────────────────┤
```

```
│                 │
│ Place Type          │
│ ☐ Restaurant         │
│ ☐ Café              │
│                 │
│ Cuisine              │
│ ☐ Indian ☐ Italian   │
│ ☐ Japanese ☐ Korean    │
│ ...               │
│                 │
│ Distance: [====●====]  │
│ 5 km                │
│                 │
│ Price Range           │
│ ☐ € ☐ €€ ☐ €€€     │
│                 │
│ Rating: [====●====]    │
│ 4.0 ⭐               │
│                 │
│ [Reset]   [Apply]     │
└─────────────────────┘
```

6.2.3 Map Screen

Layout:

```
┌─────────────────────┐
│ [📍]                │
│                 │
│    🗺 MAP           │
│                 │
│   📍 📍 📍          │
│                 │
│                 │
│  ┌───────────────┐    │
│  │ Place Name    │  │
│  │ Address       │  │
│  │ ⭐⭐⭐⭐ €€    │  │
│  │ [Details] [→]  │  │
│  └───────────────┘    │
└─────────────────────┘
   [Feed] [Map] [❤] [👤]
```

6.3 Animations

6.3.1 Transitions

- **Screen Transitions:** Slide (300ms)
- **Modal Open:** Fade + Scale (250ms)
- **Button Tap:** Scale (100ms)

6.3.2 Video Transitions

- **Scroll:** Smooth, no lag
- **Autoplay:** Fade in (200ms)
- **Pause:** Immediate

6.3.3 Feedback

- **Like Button:** Scale animation + color change
- **Share:** Haptic feedback
- **Filter Badge:** Pulse animation when active

---

7. DATABASE SCHEMA & SUPABASE SETUP

7.1 Database Schema

7.1.1 Places Table

```sql
CREATE TABLE places (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  name VARCHAR(255) NOT NULL,
  description TEXT,
  type VARCHAR(50) NOT NULL, -- 'restaurant' or 'cafe'
  cuisine VARCHAR(50) NOT NULL,
  latitude DECIMAL(10, 8) NOT NULL,
  longitude DECIMAL(11, 8) NOT NULL,
  video_url TEXT,
  images TEXT[], -- Array of image URLs
  rating DECIMAL(2, 1) DEFAULT 0.0, -- 0.0 to 5.0
  price_range VARCHAR(10) DEFAULT '$$', -- '$', '$$', '$$$', '$$$$'
  address TEXT NOT NULL,
  phone VARCHAR(20),
  website TEXT,
  instagram_url TEXT,
  opening_hours JSONB, -- Array of day/time objects
  created_at TIMESTAMP DEFAULT NOW(),
  updated_at TIMESTAMP DEFAULT NOW()
);
```

```
-- Indexes
CREATE INDEX idx_places_type ON places(type);
CREATE INDEX idx_places_cuisine ON places(cuisine);
CREATE INDEX idx_places_location ON places USING GIST(
  ll_to_earth(latitude, longitude)
);
```

7.1.2 Favorites Table

```sql
CREATE TABLE favorites (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID NOT NULL REFERENCES auth.users(id) ON DELETE CASCADE,
  place_id UUID NOT NULL REFERENCES places(id) ON DELETE CASCADE,
  created_at TIMESTAMP DEFAULT NOW(),
  UNIQUE(user_id, place_id)
);

-- Indexes
CREATE INDEX idx_favorites_user ON favorites(user_id);
CREATE INDEX idx_favorites_place ON favorites(place_id);
```

7.1.3 User Profiles Table

```sql
CREATE TABLE user_profiles (
  id UUID PRIMARY KEY REFERENCES auth.users(id) ON DELETE CASCADE,
  username VARCHAR(50),
  first_name VARCHAR(100),
  last_name VARCHAR(100),
  avatar_url TEXT,
  favorite_city VARCHAR(100),
  created_at TIMESTAMP DEFAULT NOW(),
  updated_at TIMESTAMP DEFAULT NOW()
);
```

7.2 Supabase Setup Instructions

7.2.1 Project Creation

1. Go to https://supabase.com
2. Create new project
3. Choose region (closest to users)

4. Set database password
5. Wait for project initialization

7.2.2 Database Setup

1. **Enable Extensions:**
   ```sql
   CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
   CREATE EXTENSION IF NOT EXISTS "earthdistance";
   ```

2. **Create Tables:**
   - Run SQL scripts from section 7.1

3. **Set Row Level Security (RLS):**
   ```sql
   -- Places: Public read, admin write
   ALTER TABLE places ENABLE ROW LEVEL SECURITY;
   CREATE POLICY "Places are viewable by everyone" ON places
     FOR SELECT USING (true);

   -- Favorites: User-specific
   ALTER TABLE favorites ENABLE ROW LEVEL SECURITY;
   CREATE POLICY "Users can view own favorites" ON favorites
     FOR SELECT USING (auth.uid() = user_id);
   CREATE POLICY "Users can insert own favorites" ON favorites
     FOR INSERT WITH CHECK (auth.uid() = user_id);
   CREATE POLICY "Users can delete own favorites" ON favorites
     FOR DELETE USING (auth.uid() = user_id);
   ```

7.2.3 Storage Setup

1. **Create Bucket for Media:**
   - Bucket name: `place-media`
   - Public: Yes
   - File size limit: 50MB
   - Allowed MIME types: `image/*`, `video/*`

2. **Storage Policies:**
   ```sql
   -- Public read access
   CREATE POLICY "Public Access" ON storage.objects
     FOR SELECT USING (bucket_id = 'place-media');
   ```

7.2.4 Authentication Setup

1. **Enable Providers:**
   - Apple Sign In
   - Google Sign In

2. **Configure OAuth:**
   - Add redirect URLs
   - Configure client IDs and secrets

7.2.5 Environment Variables

```env
EXPO_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
EXPO_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
```

7.3 API Endpoints

7.3.1 Places API

Get All Places:
```
GET /rest/v1/places
```

Get Place by ID:
```
GET /rest/v1/places?id=eq.{place_id}
```

Search Places (with filters):
```
GET /rest/v1/places?type=eq.restaurant&cuisine=eq.italian
```

7.3.2 Favorites API

Get User Favorites:
```
GET /rest/v1/favorites?user_id=eq.{user_id}
```

Add Favorite:
```
POST /rest/v1/favorites
Body: { "user_id": "...", "place_id": "..." }
```

Remove Favorite:
```
DELETE /rest/v1/favorites?user_id=eq.{user_id}&place_id=eq.{place_id}
```

---

8. iOS & ANDROID PUBLICATION

8.1 iOS App Store Publication

8.1.1 Prerequisites

1. **Apple Developer Account**
   - Enroll in Apple Developer Program ($99/year)
   - Access to App Store Connect

2. **App Store Connect Setup**
   - Create new app
   - Bundle ID: `com.yourcompany.placesapp`
   - App name, description, keywords
   - Screenshots (required sizes):
     - iPhone 6.7" (1290 x 2796)
     - iPhone 6.5" (1242 x 2688)
     - iPhone 5.5" (1242 x 2208)
   - App icon (1024 x 1024)

3. **Privacy Policy**
   - Required URL
   - Must cover data collection and usage

8.1.2 Build Configuration

app.json:
```json
{
  "expo": {
    "ios": {
      "bundleIdentifier": "com.yourcompany.placesapp",
      "buildNumber": "1",
      "supportsTablet": true,
      "usesAppleSignIn": true,
      "infoPlist": {
        "NSLocationWhenInUseUsageDescription": "We use your location to show nearby places.",
```

        "NSLocationAlwaysAndWhenInUseUsageDescription": "We use your location to show nearby places.",
        "NSPhotoLibraryUsageDescription": "We need access to save images to your photo library."
      }
    }
  }
}

8.1.3 Build Process

1. **Install EAS CLI:**
   ```bash
   npm install -g eas-cli
   ```

2. **Login:**
   ```bash
   eas login
   ```

3. **Configure Project:**
   ```bash
   eas build:configure
   ```

4. **Build for iOS:**
   ```bash
   eas build --platform ios
   ```

5. **Submit to App Store:**
   ```bash
   eas submit --platform ios
   ```

8.1.4 App Store Review Guidelines

- **Content:** No inappropriate content
- **Functionality:** App must work as described
- **Privacy:** Privacy policy and data usage disclosure
- **Design:** Follow Apple Human Interface Guidelines
- **Performance:** No crashes or bugs

8.2 Google Play Store Publication

8.2.1 Prerequisites

1. **Google Play Developer Account**
   - One-time fee: $25
   - Access to Google Play Console

2. **Play Console Setup**
   - Create new app
   - Package name: `com.yourcompany.placesapp`
   - App name, description, screenshots
   - App icon (512 x 512)
   - Feature graphic (1024 x 500)

8.2.2 Build Configuration

app.json:
```json
{
  "expo": {
    "android": {
      "package": "com.yourcompany.placesapp",
      "versionCode": 1,
      "adaptiveIcon": {
        "foregroundImage": "./assets/adaptive-icon.png",
        "backgroundColor": "#ffffff"
      },
      "permissions": [
        "ACCESS_FINE_LOCATION",
        "ACCESS_COARSE_LOCATION",
        "READ_EXTERNAL_STORAGE",
        "WRITE_EXTERNAL_STORAGE"
      ]
    }
  }
}
```

8.2.3 Build Process

1. **Build for Android:**
   ```bash
   eas build --platform android
   ```

2. **Submit to Play Store:**
   ```bash
   eas submit --platform android
   ```

8.2.4 Play Store Requirements

- **Content Rating:** Complete questionnaire
- **Privacy Policy:** Required URL
- **Target API Level:** Latest Android version
- **64-bit Support:** Required for new apps
- **Data Safety:** Complete data safety form

8.3 Over-the-Air Updates (EAS Update)

8.3.1 Setup

1. **Enable Updates:**
   ```json
   {
     "expo": {
       "updates": {
         "enabled": true,
         "fallbackToCacheTimeout": 0,
         "url": "https://u.expo.dev/your-project-id"
       }
     }
   }
   ```

2. **Publish Update:**
   ```bash
   eas update --branch production --message "Bug fixes and improvements"
   ```

8.3.2 Update Strategy

- **Production Branch:** Stable releases
- **Preview Branch:** Beta testing
- **Development Branch:** Internal testing

---

9. TESTING REQUIREMENTS

9.1 Unit Testing

Coverage Target: 70%+

Test Areas:

- Custom hooks (`useVideoPlayback`, `usePlaces`, `useFilters`)
- Utility functions
- Store logic (Zustand stores)
- API services

Tools:
- Jest
- React Native Testing Library

9.2 Integration Testing

Test Scenarios:
- Authentication flow
- Video playback and scrolling
- Filter application
- Favorites management
- Map interactions

9.3 E2E Testing

Critical Paths:
1. User login → Browse feed → Like place → View favorites
2. Apply filters → View filtered results → Reset filters
3. View map → Select place → View details
4. Share place → External app opens

Tools:
- Detox (React Native E2E)
- Appium (Alternative)

9.4 Performance Testing

Metrics:
- App launch time
- Video playback FPS
- Scroll performance
- Memory usage
- Battery consumption

Tools:
- React Native Performance Monitor
- Flipper
- Xcode Instruments (iOS)
- Android Profiler

9.5 Device Testing

Required Devices:
- iPhone 12/13/14 (iOS 15+)
- iPhone SE (small screen)
- iPad (tablet)
- Samsung Galaxy S21+ (Android 11+)
- Pixel 6+ (Android 12+)

Test Scenarios:
- Different screen sizes
- Different OS versions
- Network conditions (3G, 4G, WiFi)
- Offline mode

9.6 User Acceptance Testing (UAT)

Test Plan:
1. **Alpha Testing:** Internal team (2 weeks)
2. **Beta Testing:** Selected users (4 weeks)
3. **Production Testing:** Soft launch (1 week)

Feedback Collection:
- In-app feedback form
- Crash reporting (Sentry)
- Analytics (user behavior)

---

10. DEVELOPMENT TIMELINE & MILESTONES

10.1 Phase 1: Foundation (Weeks 1-2)

Deliverables:
- Project setup
- Navigation structure
- Basic UI components
- Authentication flow

Milestones:
- ✅ App launches and navigates
- ✅ User can log in
- ✅ Basic screens render

10.2 Phase 2: Core Features (Weeks 3-5)

Deliverables:
- Video feed with autoplay
- Place information display

- Like/share functionality
- Basic filtering

Milestones:
- ✅ Videos play automatically
- ✅ Smooth scrolling
- ✅ Actions work correctly

10.3 Phase 3: Advanced Features (Weeks 6-7)

Deliverables:
- Map integration
- Favorites management
- Place detail screen
- Image gallery

Milestones:
- ✅ All screens functional
- ✅ Data persistence works
- ✅ Offline support

10.4 Phase 4: Polish & Testing (Weeks 8-9)

Deliverables:
- UI/UX refinements
- Performance optimization
- Bug fixes
- Testing completion

Milestones:
- ✅ All tests passing
- ✅ Performance targets met
- ✅ No critical bugs

10.5 Phase 5: Publication (Weeks 10-11)

Deliverables:
- App Store submission
- Play Store submission
- Marketing materials
- Documentation

Milestones:
- ✅ Apps submitted
- ✅ Approval received
- ✅ Apps live in stores

---

11. DELIVERABLES

11.1 Code Deliverables

- ✅ Complete source code
- ✅ Documentation (README, API docs)
- ✅ Environment configuration files
- ✅ Database migration scripts

11.2 Documentation Deliverables

- ✅ Technical documentation
- ✅ API documentation
- ✅ Deployment guide
- ✅ User manual (optional)

11.3 Assets Deliverables

- ✅ App icons (all sizes)
- ✅ Splash screens
- ✅ Screenshots for stores
- ✅ Marketing materials

11.4 Testing Deliverables

- ✅ Test plans
- ✅ Test reports
- ✅ Bug reports
- ✅ Performance reports

11.5 Deployment Deliverables

- ✅ Production builds
- ✅ App Store listings
- ✅ Play Store listings
- ✅ Analytics setup

---

12. APPENDIX

12.1 Key Files Reference

- **Video Playback Guide:** `GUIDE_USEVIDEOPLAYER_EXPO_GO.md`
- **Architecture Analysis:** `ARCHITECTURE_ANALYSIS.md`

- **Correction Guide:** `CORRECTION_DEFINITIVE.md`

12.2 Contact & Support

- **Technical Questions:** [Your contact]
- **Project Management:** [PM contact]
- **Design Questions:** [Designer contact]

12.3 Version History

- **v1.0.0** (2024): Initial requirements document

END OF DOCUMENT