🎴 Chkoba Mobile - Fast Track Development Plan

🎯 Goal: Playable in 2 Weeks

Week 1: Core Game (Playable Locally)

Week 2: Multiplayer + Polish

🗓️ Sprint Breakdown

Sprint 1 (Days 1-3): Foundation + Local Game Engine

Goal: Play solo against yourself locally

Day 1: Project Setup & Core Models

- [ ] Flutter project with Material 3
- [ ] Install dependencies: `riverpod`, `flutter_animate`
- [ ] Create folder structure (see architecture below)
- [ ] Define core models:
  - `Card` (suit, value, display)
  - `Player` (id, cards, captured, team)
  - `GameState` (players, table, deck, turn, scores)
  - `Move` (player, card, captures)

Day 2: Game Engine Core

- [ ] `DeckService`: shuffle, deal
- [ ] `CaptureEngine`: validate captures, find possible moves
- [ ] `TurnManager`: next turn, round logic
- [ ] `ScoringEngine`: count cards, coins, 7-livra, chkobas
- [ ] Unit tests for all logic

Day 3: Basic UI (Local Game)

- [ ] Simple card widgets (no fancy design yet)
- [ ] Table view showing 4 cards
- [ ] Player hand (3 cards)
- [ ] Tap card → show valid captures → tap table card(s) → confirm
- [ ] **Milestone: You can play 2-player locally!** 🎉

Sprint 2 (Days 4-5): Polish Local Game

Goal: Beautiful UI, animations, scoring

Day 4: UI Enhancement

- [ ] Beautiful card designs
- [ ] Red theme implementation
- [ ] Dealing animation (cards fly from deck)
- [ ] Capture animation (cards collect to pile)
- [ ] Score display panel
- [ ] Turn indicator

Day 5: Complete Game Flow

- [ ] Round system (deal 3 cards when hands empty)
- [ ] Hand-end scoring screen
- [ ] Match-end celebration (reach 11/21)
- [ ] Restart game button
- [ ] **Milestone: Complete local game with scoring!** 🎊

Sprint 3 (Days 6-8): Supabase + Multiplayer

Goal: Play with your love online

Day 6: Supabase Setup

- [ ] Create Supabase project (free tier)
- [ ] Database schema:

```sql
-- rooms
id, host_id, config, status, created_at

-- room_players
room_id, player_id, position, team, joined_at

-- game_states
room_id, state_json, version, updated_at

-- moves
room_id, player_id, move_json, timestamp
```

- [ ] Enable Realtime for tables

- [ ] Set up Row Level Security (RLS) policies

Day 7: Room System

- [ ] Anonymous auth (quick guest join)
- [ ] Create room screen (choose 2/4 players, 11/21 points)
- [ ] Generate shareable room code
- [ ] Join room screen
- [ ] Waiting lobby (shows joined players)
- [ ] Auto-start when all joined

Day 8: Realtime Sync

- [ ] Subscribe to game state changes
- [ ] Broadcast moves to all players
- [ ] Turn validation (only current player can play)
- [ ] Optimistic updates + rollback on conflict
- [ ] **Milestone: Multiplayer works!** 🚀

Sprint 4 (Days 9-10): Polish & Arabic

Goal: Beautiful, bilingual, ready to share

Day 9: Localization

- [ ] `flutter_localizations` setup
- [ ] Arabic (ar.json) + English (en.json)
- [ ] RTL layout support
- [ ] Language switcher
- [ ] Arabic card names (شكوبة theme)

Day 10: Final Polish

- [ ] White theme option
- [ ] Sound effects (optional)
- [ ] Connection status indicator
- [ ] Error handling & reconnection
- [ ] Loading states everywhere
- [ ] **Milestone: Production ready!** 💎

Sprint 5 (Days 11-14): Testing & Deployment

Goal: In her hands ❤️

Days 11-12: Testing

- [ ] Test 2-player matches end-to-end
- [ ] Test 4-player teams
- [ ] Test disconnection/reconnection
- [ ] Fix critical bugs
- [ ] Performance optimization

Day 13: Build & Deploy

- [ ] Build Android APK
- [ ] Test on real devices
- [ ] Upload to Google Drive or direct share
- [ ] (Optional) TestFlight for iOS

Day 14: Play Together! 🎴 ❤️

- [ ] Celebrate your first match
- [ ] Gather feedback
- [ ] Plan future improvements

🏛️ Clean Architecture

```
lib/
├── main.dart
│
├── core/
│   ├── models/
│   │   ├── card.dart            # Card(suit, rank, value)
│   │   ├── player.dart          # Player state
│   │   ├── game_state.dart      # Complete game state
│   │   ├── game_config.dart     # 2/4 players, 11/21 points
│   │   └── move.dart            # Move representation
│   │
│   ├── engine/
│   │   ├── deck_service.dart    # Shuffle, deal cards
│   │   ├── capture_engine.dart  # Validate & compute captures
│   │   ├── turn_manager.dart    # Turn logic, round progression
│   │   └── scoring_engine.dart  # Score calculation
│   │
│   └── constants/
│       ├── card_values.dart     # Tunisian values (Q=8, J=9, K=10)
│       └── game_rules.dart      # Constants (cards per deal, etc.)
```

```
│
├── features/
│   ├── home/
│   │   ├── presentation/
│   │   │   └── home_screen.dart
│   │   └── widgets/
│   │       ├── create_room_button.dart
│   │       └── join_room_button.dart
│   │
│   ├── lobby/
│   │   ├── data/
│   │   │   └── room_repository.dart      # Supabase room operations
│   │   ├── presentation/
│   │   │   ├── create_room_screen.dart
│   │   │   ├── join_room_screen.dart
│   │   │   └── waiting_lobby_screen.dart
│   │   └── providers/
│   │       └── room_provider.dart        # Riverpod state
│   │
│   ├── game/
│   │   ├── data/
│   │   │   └── game_repository.dart      # Supabase game sync
│   │   ├── presentation/
│   │   │   └── game_screen.dart          # Main game UI
│   │   ├── widgets/
│   │   │   ├── card_widget.dart
│   │   │   ├── table_widget.dart
│   │   │   ├── player_hand_widget.dart
│   │   │   ├── score_panel.dart
│   │   │   └── turn_indicator.dart
│   │   └── providers/
│   │       └── game_provider.dart        # Game state management
│   │
│   └── settings/
│       ├── presentation/
│       │   └── settings_screen.dart
│       └── providers/
│           ├── theme_provider.dart       # Red/White theme
│           └── locale_provider.dart      # Arabic/English
│
├── shared/
│   ├── widgets/
│   │   ├── chkoba_button.dart
│   │   ├── loading_overlay.dart
│   │   └── error_dialog.dart
│   └── utils/
│       ├── extensions.dart
```

```
│       └── validators.dart
│
└── l10n/
    ├── app_ar.arb                  # Arabic translations
    └── app_en.arb                  # English translations
```

🧪 Testing Strategy

Unit Tests (Priority)

dart

```
test/core/engine/
├── deck_service_test.dart
├── capture_engine_test.dart
├── turn_manager_test.dart
└── scoring_engine_test.dart
```

Key Test Cases:

- Single card capture
- Multiple card sum capture
- Chkoba detection (table cleared)
- Round transitions
- Scoring accuracy
- Team scoring (4-player mode)

Integration Tests (Secondary)

- Full 2-player game flow
- Multiplayer room lifecycle
- Realtime sync accuracy

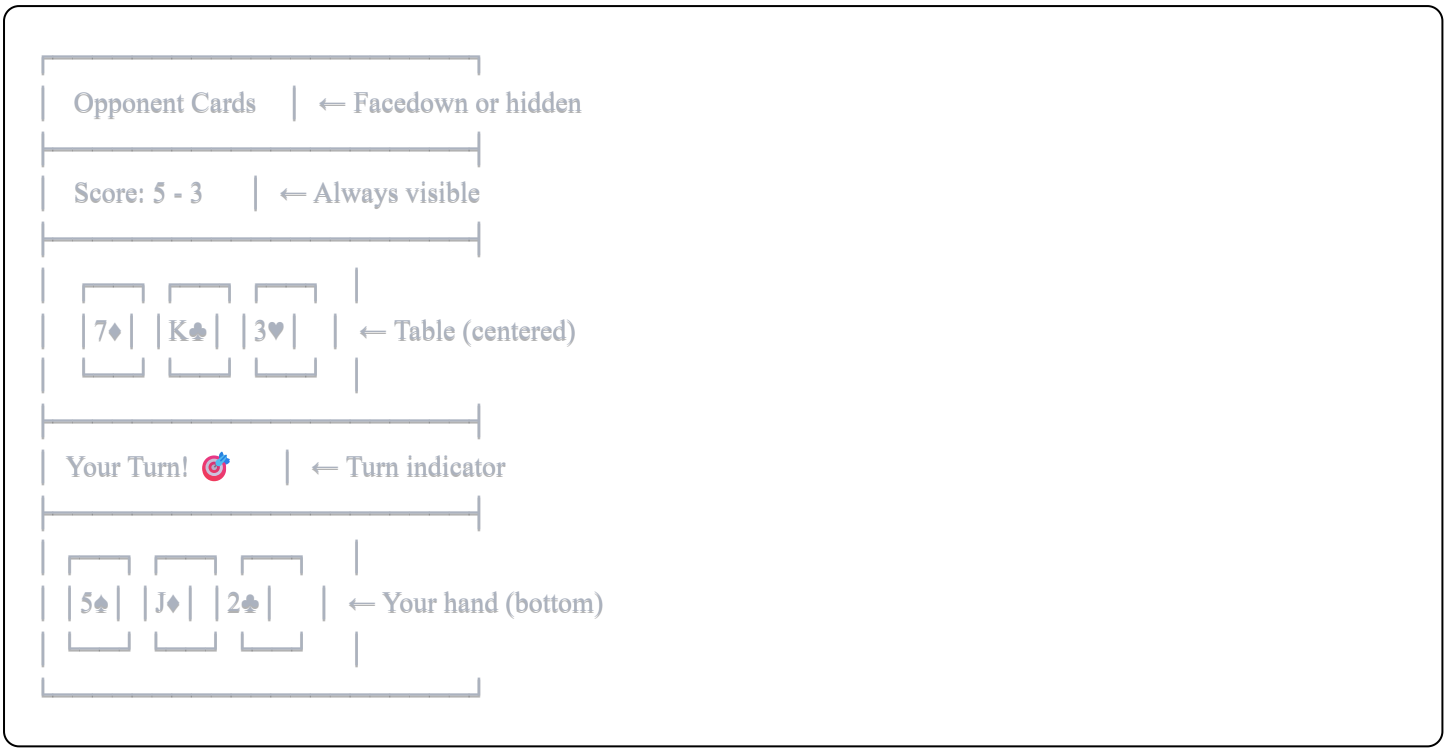
🎨 UI/UX Principles

1. Simplicity First

- Large, tappable cards
- Clear visual feedback on tap

- Obvious whose turn it is

2. Visual Hierarchy



3. Gesture Flow

1. Tap your card → highlights it
2. Tap table card(s) → shows sum preview
3. Confirm button appears → tap to execute
4. Animation plays → cards collect

4. Animations

- Cards deal: 0.3s stagger
- Capture: 0.4s slide + fade
- Chkoba: 0.6s celebration ✨
- Turn change: 0.2s fade

🗄️ Supabase Schema

sql

```sql
-- Enable Realtime
ALTER publication supabase_realtime ADD TABLE rooms, game_states, moves;

-- Rooms
CREATE TABLE rooms (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  code TEXT UNIQUE NOT NULL,
  host_id UUID NOT NULL,
  player_count INT NOT NULL CHECK (player_count IN (2, 4)),
  target_score INT NOT NULL CHECK (target_score IN (11, 21)),
  status TEXT NOT NULL DEFAULT 'waiting' CHECK (status IN ('waiting', 'playing', 'finished')),
  created_at TIMESTAMPTZ DEFAULT NOW()
);

-- Room Players
CREATE TABLE room_players (
  room_id UUID REFERENCES rooms(id) ON DELETE CASCADE,
  player_id UUID NOT NULL,
  player_name TEXT,
  position INT NOT NULL CHECK (position BETWEEN 0 AND 3),
  team INT CHECK (team IN (0, 1)),
  joined_at TIMESTAMPTZ DEFAULT NOW(),
  PRIMARY KEY (room_id, player_id)
);

-- Game States (stores JSON of current game)
CREATE TABLE game_states (
  room_id UUID PRIMARY KEY REFERENCES rooms(id) ON DELETE CASCADE,
  state JSONB NOT NULL,
  version INT NOT NULL DEFAULT 1,
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

-- Moves Log (for replay/validation)
CREATE TABLE moves (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  room_id UUID REFERENCES rooms(id) ON DELETE CASCADE,
  player_id UUID NOT NULL,
  move JSONB NOT NULL,
  created_at TIMESTAMPTZ DEFAULT NOW()
);

-- RLS Policies
ALTER TABLE rooms ENABLE ROW LEVEL SECURITY;
ALTER TABLE room_players ENABLE ROW LEVEL SECURITY;
ALTER TABLE game_states ENABLE ROW LEVEL SECURITY;
```

```
ALTER TABLE moves ENABLE ROW LEVEL SECURITY;

-- Anyone can read rooms they're part of
CREATE POLICY "Players can read their rooms"
  ON rooms FOR SELECT
  USING (id IN (SELECT room_id FROM room_players WHERE player_id = auth.uid()));

-- Similar policies for other tables...
```

🚀 Quick Start Commands

bash

```
# 1. Create Flutter project
flutter create chkoba_mobile
cd chkoba_mobile

# 2. Add dependencies (pubspec.yaml)
dependencies:
  flutter:
    sdk: flutter
  riverpod: ^2.5.0
  flutter_riverpod: ^2.5.0
  supabase_flutter: ^2.5.0
  flutter_animate: ^4.5.0
  uuid: ^4.4.0

dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^4.0.0

# 3. Run
flutter pub get
flutter run

# 4. Build APK (when ready)
flutter build apk --release
```

💡 Pro Tips for Speed

1. MVP First

Don't build:

- ❌ Complex animations initially
- ❌ Profile systems
- ❌ Chat
- ❌ Leaderboards

Do build:

- ✅ Core game loop
- ✅ Multiplayer sync
- ✅ One beautiful theme

2. Use Riverpod Generators

dart

```dart
@riverpod
class GameController extends _$GameController {
  @override
  GameState build() => GameState.initial();

  void playCard(Card card, List<Card> captures) {
    // Update state
  }
}
```

3. Local First, Network Later

- Build the entire game working locally first
- Add Supabase as a "save/load" layer later
- This keeps you focused and unblocked

4. Optimize the Critical Path

The fastest path to playing together:

1. Create room → Share code → Join → Play
2. Skip accounts, profiles, matchmaking
3. Anonymous auth is enough!

🎯 Success Metrics

Week 1 Done ✓

- [ ] You can play locally on one phone
- [ ] Scoring works correctly
- [ ] Game looks decent

Week 2 Done ✓

- [ ] Two phones can play together
- [ ] Room creation works
- [ ] Realtime sync is stable

Ready to Share ❤️

- [ ] APK builds successfully
- [ ] Tested on 2 real devices
- [ ] Arabic UI looks good
- [ ] No critical bugs

🔧 Troubleshooting Quick Fixes

Problem	Quick Fix
Cards not capturing	Check `CaptureEngine` sum logic
Realtime delays	Add optimistic updates
State conflicts	Use version numbers in DB
RTL breaking	Wrap with `Directionality` widget
Performance lag	Reduce animation complexity

❤️ Final Note

This plan prioritizes **getting it in your hands fast** while maintaining quality.

Week 1 = Solo game works
Week 2 = You're playing together

Every hour you spend on "nice-to-haves" is time not playing with your love. Stay focused on the core loop, ship it, then iterate based on real play sessions together.

Now go build something beautiful! 🎴 ✨

"A small table. Two hearts. One deck."