RemindMeX

Afeature-richmulti-timer mobileapplication built with React Native and Expo. Create, manage, andtrackmultiple timers with persistent state, background notifications, and an intuitive mobile-first interface.

Features

Core Functionality

• Multiple Timers: Create unlimited timers with custom labels and durations
• Full Timer Control: Start, pause, resume, reset, and delete timers
• Real-time Countdown: Live updates with accurate time tracking
• Background Support: Timers continue running when app is in background
• Local Notifications: Push notifications when timers complete, even when app is closed
• Persistent State: All timers automatically saved and restored between app sessions

Enhanced Features

• Recurring Timers: Set timers to repeat daily, weekly, or monthly
• Drag-to-Reorder: Organize timers by dragging and dropping
• Swipe-to-Delete: Quick gesture-based timer deletion
• Quick Presets: One-tap timer creation (5, 10, 15, 30 min, 1, 2 hours)
• Haptic Feedback: Tactile responses for all interactions
• Dark Mode: Full theme support with system preference detection
• Empty State: Helpful UI when no timers exist with permission prompts

Getting Started

Prerequisites

• Node.js (v18 or higher)
• npm or yarn
• Expo CLI
• For iOS: macOS with Xcode
• For Android: Android Studio with SDK

Installation

• Clone the repository
• Install dependencies
• Start the development server
• Run on a device/emulator

```
1. Clone the repository
   git clone https://github.com/yourusername/remindmex.git
   cd remindmex

2. Install dependencies
   npm install

3. Start development server
   npm start
```

Architecture & Design Decisions

State Management

Theapp usesa custom hooks-based architecture rather than external state management libraries:
• useTimers Hook: Central timer management with local state
• React Context: Theme management with ThemeContext
• AsyncStorage: Persistent data layer for timer state

Why this approach?

• Reduces bundle size and dependencies
• Provides fine-grained control over timer logic
• Eliminates complexity of Redux/MobX

Timer Logic & Accuracy

KeyImplementation Details:
1. Timers recalculate remaining time based on startedAt instead of just decrementing.
2. Benefits: prevents drift, stays accurate in background, handles system time changes.
3. App State Handling: Listens to AppState changes to recalc timers when returning foreground.

Technical stack

Category	**Technology** React Native + Expo	**Purpose**
Framework	TypeScript	Cross-platform development
Language	Expo Router	Type safety and DX
Routing	NativeWind	File-based routing
Styling	AsyncStorage	Tailwind CSS for React Native
Storage	expo-notifications	Local persistence
Notifications	react-native-gesture-handler	Local push notifications
Gestures	react-native-reanimated	Swipe and drag interactions
Animations	expo-haptics	Smooth animations
Haptics		Tactile feedback

```
4. Run on device/emulat
```