🚀 SideQuest - Quick Start Guide

Get your development environment set up and start building in 30 minutes!

Prerequisites

Before you begin, make sure you have:

- ✅ **Node.js 18+** installed (Download)
- ✅ **VS Code** or **Cursor IDE** (VS Code | Cursor)
- ✅ **Expo Go** app on your phone (iOS | Android)
- ✅ **Supabase account** (free) - Sign up
- ✅ **Anthropic API key** - Get key
- ✅ **Git** installed

Step 1: Create Your Project (5 minutes)

```
# Create new Expo project with TypeScript
npx create-expo-app@latest sidequest-app --template

# When prompted, choose: blank (TypeScript)

cd sidequest-app
```

Step 2: Install Dependencies (3 minutes)

```
# Core dependencies
npx expo install react-native-reanimated react-native-gesture-handler
npm install moti

# Backend & API
npm install @supabase/supabase-js @anthropic-ai/sdk

# State management & forms
```

```
npm install zustand react-hook-form zod

# Styling (Tailwind for React Native)
npm install nativewind
npm install --save-dev tailwindcss@3.3.2

# Navigation
npm install @react-navigation/native @react-navigation/bottom-tabs @react-navigat
npx expo install react-native-screens react-native-safe-area-context

# Icons
npx expo install @expo/vector-icons
```

Step 3: Setup NativeWind (Tailwind) (2 minutes)

```
# Initialize Tailwind
npx tailwindcss init
```

Update `tailwind.config.js`:

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: [
    "./App.{js,jsx,ts,tsx}",
    "./src/**/*.{js,jsx,ts,tsx}"
  ],
  theme: {
    extend: {
      colors: {
        'quest-green': '#00FF87',
        'quest-purple': '#6B48FF',
        'gold': '#FFD700',
        'dark-bg': '#0A0E27',
        'card-bg': '#1A1F3A',
        'text-secondary': '#A0AEC0',
      },
    },
  },
  plugins: [],
}
```

Create `babel.config.js`:

```
module.exports = function(api) {
```

```
  api.cache(true);
  return {
    presets: ['babel-preset-expo'],
    plugins: [
      'nativewind/babel',
      'react-native-reanimated/plugin', // Must be last!
    ],
  };
};
```

Step 4: Setup Supabase (10 minutes)

4.1 Create Supabase Project

1. Go to https://supabase.com/
2. Click "Start your project"
3. Create new organization (if needed)
4. Create new project:
   - Name: `sidequest`
   - Database Password: (save this securely!)
   - Region: Choose closest to you
5. Wait for project to provision (~2 minutes)

4.2 Get Your API Keys

1. Go to Project Settings > API
2. Copy:
   - `Project URL` (looks like: https://xxxxx.supabase.co)
   - `anon public` key (long string)

4.3 Run Database Setup

In Supabase Dashboard:

1. Go to SQL Editor
2. Click "New Query"

3. Copy and paste this schema:

```
-- Enable UUID extension
CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

-- Users table
CREATE TABLE public.users (
  id UUID PRIMARY KEY REFERENCES auth.users(id) ON DELETE CASCADE,
  email TEXT UNIQUE NOT NULL,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  subscription_tier TEXT DEFAULT 'free' CHECK (subscription_tier IN ('free', 'pre
  subscription_expires_at TIMESTAMP WITH TIME ZONE,
  onboarding_completed BOOLEAN DEFAULT FALSE,
  monthly_quest_count INTEGER DEFAULT 0,
  last_quest_reset_date DATE DEFAULT CURRENT_DATE
);

-- User profiles
CREATE TABLE public.user_profiles (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID REFERENCES public.users(id) ON DELETE CASCADE,
  skills TEXT[] DEFAULT '{}',
  available_hours_per_week INTEGER,
  resources TEXT[] DEFAULT '{}',
  goals TEXT[] DEFAULT '{}',
  interests TEXT[] DEFAULT '{}',
  location_type TEXT CHECK (location_type IN ('urban', 'suburban', 'rural')),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  UNIQUE(user_id)
);

-- Quests table
CREATE TABLE public.quests (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID REFERENCES public.users(id) ON DELETE CASCADE,
  status TEXT DEFAULT 'suggested' CHECK (status IN ('suggested', 'active', 'compl
  ai_generated BOOLEAN DEFAULT FALSE,
  custom_data JSONB,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  started_at TIMESTAMP WITH TIME ZONE,
  completed_at TIMESTAMP WITH TIME ZONE
);

-- Earnings table
CREATE TABLE public.earnings (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
```

```
  user_id UUID REFERENCES public.users(id) ON DELETE CASCADE,
  quest_id UUID REFERENCES public.quests(id) ON DELETE CASCADE,
  amount DECIMAL(10, 2) NOT NULL,
  date DATE NOT NULL DEFAULT CURRENT_DATE,
  notes TEXT,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

-- Indexes for performance
CREATE INDEX idx_quests_user_id ON public.quests(user_id);
CREATE INDEX idx_quests_status ON public.quests(status);
CREATE INDEX idx_earnings_user_id ON public.earnings(user_id);

-- Enable Row Level Security
ALTER TABLE public.users ENABLE ROW LEVEL SECURITY;
ALTER TABLE public.user_profiles ENABLE ROW LEVEL SECURITY;
ALTER TABLE public.quests ENABLE ROW LEVEL SECURITY;
ALTER TABLE public.earnings ENABLE ROW LEVEL SECURITY;

-- RLS Policies - Users can only see their own data
CREATE POLICY "Users can view own data" ON public.users
  FOR SELECT USING (auth.uid() = id);

CREATE POLICY "Users can update own data" ON public.users
  FOR UPDATE USING (auth.uid() = id);

CREATE POLICY "Users can view own profile" ON public.user_profiles
  FOR SELECT USING (auth.uid() = user_id);

CREATE POLICY "Users can insert own profile" ON public.user_profiles
  FOR INSERT WITH CHECK (auth.uid() = user_id);

CREATE POLICY "Users can update own profile" ON public.user_profiles
  FOR UPDATE USING (auth.uid() = user_id);

CREATE POLICY "Users can view own quests" ON public.quests
  FOR SELECT USING (auth.uid() = user_id);

CREATE POLICY "Users can insert own quests" ON public.quests
  FOR INSERT WITH CHECK (auth.uid() = user_id);

CREATE POLICY "Users can update own quests" ON public.quests
  FOR UPDATE USING (auth.uid() = user_id);

CREATE POLICY "Users can view own earnings" ON public.earnings
  FOR SELECT USING (auth.uid() = user_id);
```

```
CREATE POLICY "Users can insert own earnings" ON public.earnings
  FOR INSERT WITH CHECK (auth.uid() = user_id);
```

4. Click “Run” (bottom right)
5. Verify success (should see “Success. No rows returned”)

Step 5: Configure Environment Variables (2 minutes)

Create `.env` file in your project root:

```
# Supabase
EXPO_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
EXPO_PUBLIC_SUPABASE_ANON_KEY=your-anon-key-here

# Anthropic AI
EXPO_PUBLIC_ANTHROPIC_API_KEY=sk-ant-your-key-here
```

⚠️ **Important:** Add `.env` to your `.gitignore`:

```
echo ".env" >> .gitignore
```

Step 6: Create Project Structure (3 minutes)

```
# Create folder structure
mkdir -p src/{components,screens,services,hooks,store,types,constants,utils,navig
mkdir -p src/components/{common,quests,onboarding,animations}
mkdir -p src/screens/{auth,onboarding,quests,profile}
```

Step 7: Create Supabase Client (2 minutes)

Create `src/services/supabase.ts`:

```
import 'react-native-url-polyfill/auto';
import AsyncStorage from '@react-native-async-storage/async-storage';
import { createClient } from '@supabase/supabase-js';

const supabaseUrl = process.env.EXPO_PUBLIC_SUPABASE_URL!;
const supabaseAnonKey = process.env.EXPO_PUBLIC_SUPABASE_ANON_KEY!;

export const supabase = createClient(supabaseUrl, supabaseAnonKey, {
```

```
    auth: {
      storage: AsyncStorage,
      autoRefreshToken: true,
      persistSession: true,
      detectSessionInUrl: false,
    },
  });
```

Install required dependencies:

```
npm install react-native-url-polyfill @react-native-async-storage/async-storage
npx expo install @react-native-async-storage/async-storage
```

Step 8: Start Development! (immediate)

```
# Start Expo development server
npx expo start
```

Then:

1. Scan the QR code with Expo Go on your phone
2. Or press `a` for Android emulator
3. Or press `i` for iOS simulator

Next Steps

Now that your environment is set up, start building:

Week 1: Authentication

- Create login/signup screens
- Implement Supabase auth
- Add navigation

Week 2: Onboarding

- Build multi-step onboarding flow
- Create skill/resource selectors

- Save profile to database

Week 3: Quest System

- Integrate Claude API
- Build quest generation
- Create quest discovery UI

Week 4: Quest Tracking

- Build active quests screen
- Add progress tracking
- Polish and test

Helpful Commands

```
# Clear cache if things break
npx expo start -c

# Install iOS dependencies (Mac only)
npx pod-install

# Check for outdated packages
npx expo-doctor

# Build for production later
eas build --platform android
eas build --platform ios
```

Troubleshooting

“Module not found” errors:

```
npm install
npx expo start -c
```

Supabase connection issues:

- Check .env file exists and has correct values
- Verify API keys are correct in Supabase dashboard
- Make sure you're not using the `service_role` key (use `anon` key)

TypeScript errors:

```
npm install --save-dev @types/react @types/react-native
```

Animation not working:

- Make sure `react-native-reanimated/plugin` is LAST in babel.config.js plugins
- Clear cache: `npx expo start -c`

Resources

- **Expo Docs:** https://docs.expo.dev/
- **Supabase Docs:** https://supabase.com/docs
- **React Navigation:** https://reactnavigation.org/
- **NativeWind:** https://www.nativewind.dev/
- **Anthropic API:** https://docs.anthropic.com/

What's Next?

Once you have everything running:

1. Check out the `/code-templates` folder for starter components
2. Read the full master plan PDF for detailed architecture
3. Join the Expo Discord for help: https://chat.expo.dev/

You're ready to start building! 🚀