**Ratchet Money Marketing Funnel Supabase Setup**

Perfect! Since you already have a Supabase setup for your main application, here are the streamlined steps to integrate the Ratchet Money marketing funnel:

**🎯 Steps for Existing Supabase Setup**

**1. Extend Your Existing Supabase Database**

**bash**

**Apply to index.html**

**Run**

*# Add the marketing funnel tables to your existing database*

psql -h your-existing-supabase-host -U postgres -d postgres -f supabase-schema-clean.sql

**What this adds to your existing database:**

* leads table for marketing funnel data
* email\_logs table for email tracking
* email\_templates table for email automation
* Custom types and functions for the marketing system

**2. Configure Environment Variables**

Add these to your existing .env file:

env

Apply to index.html

# Marketing Funnel - Supabase (use your existing project)

REACT\_APP\_SUPABASE\_URL=https://your-existing-project.supabase.co

REACT\_APP\_SUPABASE\_ANON\_KEY=your-existing-anon-key

REACT\_APP\_SUPABASE\_SERVICE\_ROLE\_KEY=your-existing-service-role-key

# Marketing Funnel - Email Service

REACT\_APP\_EMAIL\_PROVIDER=mock

REACT\_APP\_APP\_URL=http://localhost:3000

# Marketing Funnel - Analytics (optional)

REACT\_APP\_GOOGLE\_ANALYTICS\_ID=your-existing-ga-id

**3. Update Your Existing Supabase Client**

Your existing src/lib/supabase.ts should already work, but verify it includes:

typescript

Apply to index.html

*// Make sure your existing supabase client has these types*

export type UserSegment = 'stress-free' | 'relationship-spender' | 'emotional-manager' | 'crisis-mode'

export type ProductTier = 'Budget ($10)' | 'Mid-tier ($20)' | 'Professional ($50)'

export type EmailType = 'confirmation' | 'assessment\_results' | 'follow\_up'

export type EmailStatus = 'sent' | 'delivered' | 'failed'

**4. Test Integration with Existing App**

**bash**

**Apply to index.html**

**Run**

*# Start your existing development server*

npm start

*# Test the marketing funnel components:*

*# - Navigate to /assessment or wherever you want the funnel*

*# - Test email collection*

*# - Complete assessment*

*# - Verify data is saved to your existing Supabase*

**🔄 Integration Options**

**Option A: Separate Route/Page**

Add the marketing funnel as a new route in your existing app:

typescript

Apply to index.html

*// In your existing App.tsx or router*

import { AssessmentWorkflow } from './components/AssessmentWorkflow'

*// Add route*

<Route path="/marketing-funnel" element={<AssessmentWorkflow />} />

**Option B: Embedded Component**

Embed the funnel into existing pages:

typescript

Apply to index.html

*// In your existing landing page or component*

import { EmailCollection } from './components/EmailCollection'

import { AssessmentForm } from './components/AssessmentForm'

*// Use as needed*

<EmailCollection onEmailSubmitted={handleEmail} />

**Option C: Standalone Marketing App**

Keep it separate but use the same Supabase:

bash

Apply to index.html

Run

*# Create a separate marketing app directory*

mkdir ratchet-money-marketing

cd ratchet-money-marketing

*# Copy the marketing components and use your existing Supabase credentials*

**🛠️ Database Integration Steps**

**Step 1: Verify Database Access**

**bash**

**Apply to index.html**

**Run**

*# Test connection to your existing Supabase*

npx supabase status

*# or check your existing app's database connection*

**Step 2: Add Marketing Tables**

**sql**

**Apply to index.html**

*-- Run this in your existing Supabase SQL editor*

*-- This adds the marketing funnel tables without affecting your existing data*

*-- Enable extensions if not already enabled*

CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

*-- Add marketing funnel tables*

CREATE TABLE IF NOT EXISTS leads (

  id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

  email VARCHAR(255) UNIQUE NOT NULL,

  name VARCHAR(255),

  phone VARCHAR(50),

  segment VARCHAR(100) NOT NULL DEFAULT 'stress-free',

  score INTEGER NOT NULL DEFAULT 0,

  product\_tier VARCHAR(100) NOT NULL DEFAULT 'Budget ($10)',

  created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  confirmed BOOLEAN DEFAULT FALSE,

  assessment\_completed BOOLEAN DEFAULT FALSE,

  assessment\_answers JSONB DEFAULT '{}',

  email\_sequence\_sent INTEGER DEFAULT 0,

  last\_email\_sent TIMESTAMP WITH TIME ZONE,

  lead\_source VARCHAR(100),

  utm\_source VARCHAR(100),

  utm\_medium VARCHAR(100),

  utm\_campaign VARCHAR(100),

  utm\_term VARCHAR(100),

  utm\_content VARCHAR(100),

  contact\_method VARCHAR(50) DEFAULT 'email',

  beta\_interest BOOLEAN DEFAULT FALSE,

  status VARCHAR(50) DEFAULT 'active'

);

*-- Add other tables as needed...*

**Step 3: Set Up Row Level Security (RLS)**

**sql**

**Apply to index.html**

*-- Enable RLS on marketing tables*

ALTER TABLE leads ENABLE ROW LEVEL SECURITY;

*-- Create policies for marketing funnel*

CREATE POLICY "Marketing funnel can insert leads" ON leads

  FOR INSERT WITH CHECK (true);

CREATE POLICY "Marketing funnel can read leads" ON leads

  FOR SELECT USING (true);

CREATE POLICY "Marketing funnel can update leads" ON leads

  FOR UPDATE USING (true);

**�� Testing Strategy**

**Phase 1: Database Integration Test**

**bash**

**Apply to index.html**

**Run**

*# Test that marketing funnel can write to your existing Supabase*

npm start

*# Complete the assessment flow*

*# Check your Supabase dashboard to see if data is saved*

**Phase 2: Component Integration Test**

**bash**

**Apply to index.html**

**Run**

*# Test components in your existing app context*

*# Verify no conflicts with existing components*

*# Test routing and navigation*

**Phase 3: End-to-End Test**

**bash**

**Apply to index.html**

**Run**

*# Test complete user journey*

*# Verify email automation (mock mode)*

*# Check analytics tracking*

**📊 Monitoring Integration**

**Check Your Existing Supabase Dashboard**

* **Table Editor**: Verify leads table is created and receiving data
* **Logs**: Monitor for any errors or conflicts
* **Storage**: Check if you need additional storage for marketing data

**Verify No Conflicts**

* **Table names**: Ensure no naming conflicts with existing tables
* **Functions**: Check for any function name conflicts
* **Policies**: Verify RLS policies don't interfere with existing app

**🚀 Next Steps**

**Immediate (Today)**

1. **Add marketing tables** to your existing Supabase
2. **Update environment variables** with your existing credentials
3. **Test database connection** with marketing funnel
4. **Verify data is being saved** correctly

**This Week**

1. **Integrate components** into your existing app structure
2. **Test complete workflow** end-to-end
3. **Refine content** and user experience
4. **Set up email templates** in your database

**Next Week**

1. **Optimize conversion rates** based on testing
2. **Add analytics tracking** to your existing setup
3. **Prepare for production** deployment
4. **Document integration** for your team

**🔧 Troubleshooting Common Issues**

**If Tables Already Exist**

**sql**

**Apply to index.html**

*-- Check if tables exist first*

SELECT table\_name FROM information\_schema.tables

WHERE table\_schema = 'public' AND table\_name IN ('leads', 'email\_logs');

*-- If they exist, you can either:*

*-- 1. Drop and recreate (if safe)*

*-- 2. Alter existing tables to add missing columns*

*-- 3. Use different table names*

**If RLS Conflicts**

**sql**

**Apply to index.html**

*-- Check existing policies*

SELECT \* FROM pg\_policies WHERE tablename = 'leads';

*-- Adjust policies as needed for your use case*

Would you like me to help you with any specific step, such as adding the tables to your existing Supabase or integrating the components into your current app structure?