Dana AI Documentation

System Overview

Dana AI is a comprehensive social media management and automation platform that integrates with multiple messaging platforms (Facebook, Instagram, WhatsApp) to provide AI-powered customer interactions.

Architecture

Database Schema

The system uses Supabase as its database with the following key tables:

1. **profiles**
   - Stores user profile information
   - Links to auth.users through id
   - Tracks account setup and welcome email status
   - Fields: id, email, company, account_setup_complete, welcome_email_sent

2. **conversations**
   - Stores customer conversations across platforms
   - Links to users through user_id
   - Fields: id, user_id, platform, client_name, client_company, status

3. **messages**
   - Stores individual messages within conversations
   - Links to conversations through conversation_id
   - Fields: id, conversation_id, content, sender_type, created_at

4. **responses**
   - Tracks AI-generated responses
   - Links to users through user_id
   - Fields: id, user_id, content, platform

5. **tasks**
   - Manages tasks generated from conversations
   - Links to users through user_id
   - Fields: id, user_id, description, status, priority, platform, client_name

6. **interactions**
   - Tracks all user interactions across platforms
   - Links to users through user_id
   - Fields: id, user_id, platform, client_name, interaction_type

7. **knowledge_files**
   - Stores uploaded knowledge base documents
   - Links to users through user_id
   - Fields: id, user_id, file_name, file_size, file_type, content

8. **subscription_tiers**
   - Defines available subscription plans
   - Fields: id, name, description, price, features, platforms

9. **user_subscriptions**
   - Manages user subscriptions
   - Links users to subscription tiers
   - Fields: id, user_id, subscription_tier_id, status, start_date, end_date

10. **admin_users**
    - Manages admin access and roles
    - Links to users through user_id
    - Fields: id, user_id, email, username, role

11. **integrations_config**
    - Stores platform integration settings
    - Links to users through user_id
    - Fields: id, user_id, integration_type, config, status

Frontend Components

The React application is organized into the following key components:

1. **App.tsx**
   - Main application component
   - Handles authentication state
   - Manages routing between main sections
   - Controls subscription and setup flows

2. **AuthForm.tsx**
   - Handles user sign-in and sign-up
   - Manages password reset flow
   - Stores credentials in localStorage if "Remember me" is checked

3. **Sidebar.tsx**
   - Main navigation component
   - Provides access to all major sections
   - Shows user context and logout option

4. **Dashboard Components**
   - MetricCard.tsx: Displays individual metrics
   - TopIssuesChart.tsx: Shows common customer issues
   - InteractionChart.tsx: Visualizes platform interactions
   - ConversationsList.tsx: Lists recent conversations

5. **Feature Components**
   - KnowledgeBase.tsx: Manages knowledge base documents
   - Integrations.tsx: Handles platform integrations
   - Support.tsx: Provides customer support interface
   - RateUs.tsx: Handles user feedback

6. **Admin Components**
   - AdminApp.tsx: Admin dashboard entry point
   - AdminDashboard.tsx: Main admin interface
   - AdminLogin.tsx: Secure admin authentication

Backend Integration

1. **Supabase Integration**
   - Authentication handled through @supabase/supabase-js
   - Real-time subscriptions for live updates
   - Row Level Security (RLS) for data protection

2. **Socket.IO Server**
   - Handles real-time metrics updates
   - Manages WebSocket connections
   - Broadcasts database changes to clients

Security

1. **Row Level Security (RLS)**
   - All tables have RLS enabled
   - Users can only access their own data
   - Admins have elevated access through policies

2. **Authentication**
   - Email/password authentication
   - Secure password reset flow
   - Session management through Supabase

3. **Admin Access**
   - Separate admin authentication flow

   - Role-based access control
   - Secure admin operations

Data Flow

1. **User Authentication**
   ```
   User Login/Signup → Supabase Auth → Create Profile → Setup Flow
   ```

2. **Conversation Flow**
   ```
   Platform Message → Make.com Workflow → AI Processing → Response Generation → Platform Reply
   ```

3. **Task Management**
   ```
   Message Analysis → Task Creation → Priority Assignment → Dashboard Update
   ```

4. **Analytics Flow**
   ```
   User Interactions → Real-time Processing → Socket.IO → Dashboard Updates
   ```

Integration Setup

1. **Platform Requirements**
   - Facebook Developer account
   - Instagram Professional account
   - WhatsApp Business API access

2. **Make.com Workflow**
   - Handles message routing
   - Processes platform-specific formats
   - Manages AI integration

3. **Supabase Setup**
   - Database configuration
   - Authentication setup
   - RLS policy configuration

Subscription Management

1. **Tier Structure**
   - Starter Package: Single platform
   - Professional Package: Dual platform
   - Enterprise Package: All platforms

2. **Features by Tier**
   - AI responses
   - Analytics access
   - Platform support
   - Priority support levels

Development Guidelines

1. **Code Organization**
   - Components in src/components
   - Hooks in src/hooks
   - Types in src/types
   - Utilities in src/lib

2. **State Management**
   - React hooks for local state
   - Supabase real-time for global state
   - Socket.IO for live updates

3. **Styling**
   - Tailwind CSS for styling
   - Lucide React for icons
   - Responsive design patterns

4. **Error Handling**
   - Consistent error boundaries
   - User-friendly error messages
   - Proper error logging

Deployment

1. **Environment Setup**
   - Supabase configuration
   - Socket.IO server setup
   - Make.com workflow deployment

2. **Build Process**
   - Frontend build with Vite

   - Server deployment
   - Database migrations

Maintenance

1. **Regular Tasks**
   - Database backups
   - Log monitoring
   - Performance optimization

2. **Updates**
   - Security patches
   - Feature updates
   - Dependency management

Troubleshooting

1. **Common Issues**
   - Authentication problems
   - Integration disconnections
   - Real-time update delays

2. **Resolution Steps**
   - Check connection status
   - Verify credentials
   - Review error logs

Support Resources

1. **Documentation**
   - API documentation
   - Integration guides
   - Troubleshooting guides

2. **Contact**
   - Technical support
   - Admin support
   - Sales inquiries