Nirva

What is Nirva?

Nirva transforms AI assistants from passive advisors into active executors. Instead of just suggesting actions, Rube-powered AI can actually perform them across hundreds of applications including Slack, Notion, GitHub, Google Calendar, Jira, HubSpot, and many more.

Built on the Composio framework, Rube automates:

- **Authentication** - Seamlessly connects to your apps via OAuth, API keys, and other methods
- **Tool Discovery** - Intelligently identifies the right tools for any task
- **Execution** - Performs actions across multiple applications in parallel
- **Workflow Orchestration** - Chains complex multi-step operations across different platforms

Features

- **AI-Powered Chat Interface** - Natural language interaction powered by OpenAI GPT-4
- **500+ App Integrations** - Connect to virtually any popular business application
- **Secure Authentication** - User authentication via Supabase with per-user app connections
- **Tool Router** - Automatic tool discovery and execution via Composio's experimental Tool Router
- **Conversation Management** - Persistent chat history and conversation tracking
- **App Connection Management** - Connect and disconnect apps with a user-friendly interface
- **Real-time Streaming** - Live AI responses with tool execution visibility

Technical Architecture

Open Nirva is built with modern web technologies and leverages cutting-edge AI frameworks:

Core Technologies

- **Frontend**: Next.js 15 with React 19 and TypeScript
- **Styling**: Tailwind CSS with custom design system
- **Backend**: Next.js API Routes (serverless)
- **Database**: Supabase (PostgreSQL)
- **Authentication**: Supabase Auth
- **AI Framework**: Vercel AI SDK with OpenAI GPT-5
- **Tool Integration**: Composio with Tool Router and MCP (Model Context Protocol)

Architecture Components

1. **Tool Router**

Composio's experimental Tool Router is the brain of Open Rube. It:

- Discovers relevant tools from 500+ integrations based on user intent
- Manages authentication flows automatically
- Executes tools in parallel when possible
- Handles errors and retries intelligently

```
 // Tool Router creates MCP sessions per user
const mcpSession = await composio.experimental.toolRouter.crea
  toolkits: [] // Empty array allows all available tools
});
```

2. **Model Context Protocol (MCP)**

MCP facilitates seamless communication between the AI model and Composio tools:

- Standardized protocol for tool discovery and execution
- Streaming HTTP transport for real-time updates
- Session-based architecture for maintaining context

3. AI Agent Workflow

```
User Input → GPT-5 → Tool Router → Tool Discovery → Authentic
→ Tool Execution → Result Processing → AI Response → User
```

4. Database Schema

- **Users**: Managed by Supabase Auth
- **Conversations**: Chat session tracking
- **Messages**: Individual message history with role-based storage
- **Connected Accounts**: User's authenticated app connections

Getting Started

Prerequisites

Before you begin, ensure you have:

- Node.js 20+ installed
- npm, yarn, or pnpm package manager
- A Composio account and API key
- A Supabase project set up
- An OpenAI API key

Installation

1. **Clone the repository**

2. **Install dependencies**

```
npm install
# or
yarn install
# or
pnpm install
```

3. **Set up environment variables**

   Create a `.env.local` file in the root directory:

```
# Composio
COMPOSIO_API_KEY=your_composio_api_key_here

# OpenAI
OPENAI_API_KEY=your_openai_api_key_here

# Supabase
NEXT_PUBLIC_SUPABASE_URL=your_supabase_project_url
NEXT_PUBLIC_SUPABASE_ANON_KEY=your_supabase_anon_key

NEXT_PUBLIC_APP_URL=http://localhost:3000
```

4. **Set up Supabase database**

   Run the database setup script in your Supabase SQL editor to create the necessary tables for conversations and messages.

5. **Configure Supabase Auth**

   In your Supabase project dashboard:

   - Enable Google authentication and add Client Id and Client Secret
   - Configure redirect URLs to include your app's domain
   - Set up any additional OAuth providers if needed

Running the Application

1. **Development mode**

```
npm run dev
# or
yarn dev
# or
pnpm dev
```

2. **Open your browser**

   Navigate to http://localhost:3000

3. **Sign up/Sign in**

   Create an account or sign in with your credentials

4. **Connect apps**

   Go to the "Apps" tab and connect the applications you want your AI agent to access

5. **Start chatting**

   Ask your AI agent to perform tasks across your connected apps!

Production Build

```
npm run build
npm start
```

Project Structure

```
open-rube/
├── app/
│   ├── api/                    # API routes
│   │   ├── chat/               # Chat endpoint with Tool Route
│   │   ├── authConfig/         # Composio auth configuration
│   │   ├── authLinks/          # App connection management
│   │   ├── connectedAccounts/  # User's connected apps
│   │   └── conversations/      # Conversation history
│   ├── components/             # React components
```

```
│   │   ├── ChatPageWithAuth.tsx
│   │   ├── AppsPageWithAuth.tsx
│   │   ├── ToolCallDisplay.tsx
│   │   └── ...
│   ├── utils/                  # Utility functions
│   │   ├── composio.ts        # Composio client setup
│   │   ├── chat-history.ts    # Database operations
│   │   └── supabase/          # Supabase client configuration
│   └── auth/                   # Authentication pages
├── public/                     # Static assets
├── package.json
└── README.md
```

Configuration

Customizing Tool Router

You can customize which toolkits are available by modifying the Tool Router session creation in `app/api/chat/route.ts`:

```
const mcpSession = await composio.experimental.toolRouter.cre
  toolkits: ['github', 'slack', 'gmail'] // Specify specific
});
```

Changing the AI Model

Modify the model in `app/api/chat/route.ts`:

```
const result = await streamText({
  model: openai('gpt-5'), // Change to your preferred model
  // ...
});
```

Learn More

Composio Resources

- Composio Official Website
- Composio Documentation
- Composio GitHub Repository
- Tool Router Quick Start Guide

Related Technologies

- Next.js Documentation
- Vercel AI SDK
- Supabase Documentation
- Model Context Protocol

Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

License

This project is open source and available under the MIT License.

Acknowledgments

- Built with Composio - the skill layer for AI agents
- Inspired by Rube
- Powered by OpenAI's GPT-5