
# Jeeny Smart Task Manager: Architecture & Design Documentation

Author: Muhammad Affan

Date: 6th July 2025

# Table of Contents

# 1. Tech Stack Analysis

## Current Implementation Stack

- *Frontend Framework:* Next.js 14+ with App Router
- *Runtime:* Node.js
- *Database:* Supabase (PostgreSQL)
- *Authentication:* Supabase Auth
- *Styling:* Tailwind CSS
- *Real-time Features:* Supabase Realtime
- *Deployment:* Vercel
- *Language:* Typescript

# 2. Architecture Rationale

### a) Next.js 14+ with App Router

- *Full-Stack Integration:* Combines frontend and backend in unified codebase
- *Server-Side Rendering:* Enhanced SEO and faster initial page loads
- *Built-in API Routes:* Eliminates need for separate backend server
- *File-based Routing:* Intuitive and maintainable route organization
- *Typescript Integration:* Native type safety support
- *Performance Optimizations:* Automatic code splitting, image optimization

### b) Supabase (PostgreSQL + BaaS)

- *Managed PostgreSQL:* Production-ready database without DevOps overhead
- *Integrated Authentication:* OAuth, JWT, and row-level security built-in
- *Real-time Subscriptions:* WebSocket-based live updates for collaboration
- *Auto-generated APIs:* RESTful and GraphQL endpoints from database schema
- *Row-Level Security:* Database-enforced user data isolation
- *Cost-Effective:* Generous free tier for development and scaling

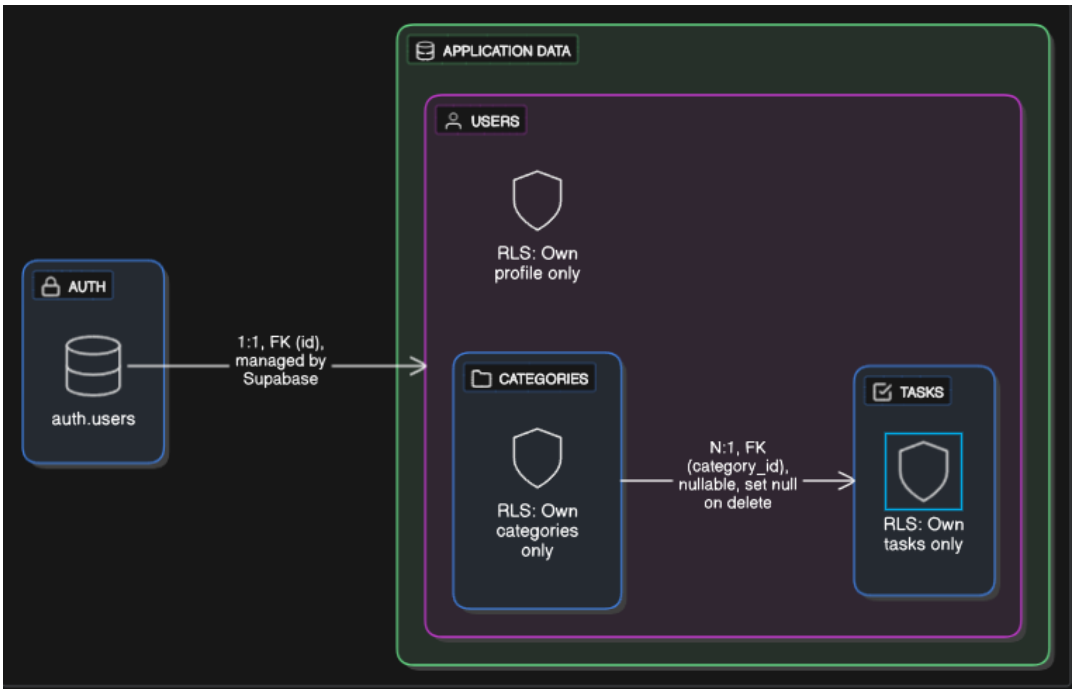
### c) Typescript

- *Compile-time Type Safety:* Catches errors before runtime
- *Enhanced Developer Experience:* Superior IDE support and autocomplete
- *Self-documenting Code:* Interfaces serve as living documentation
- *Team Collaboration:* Enforces consistent data structures across team

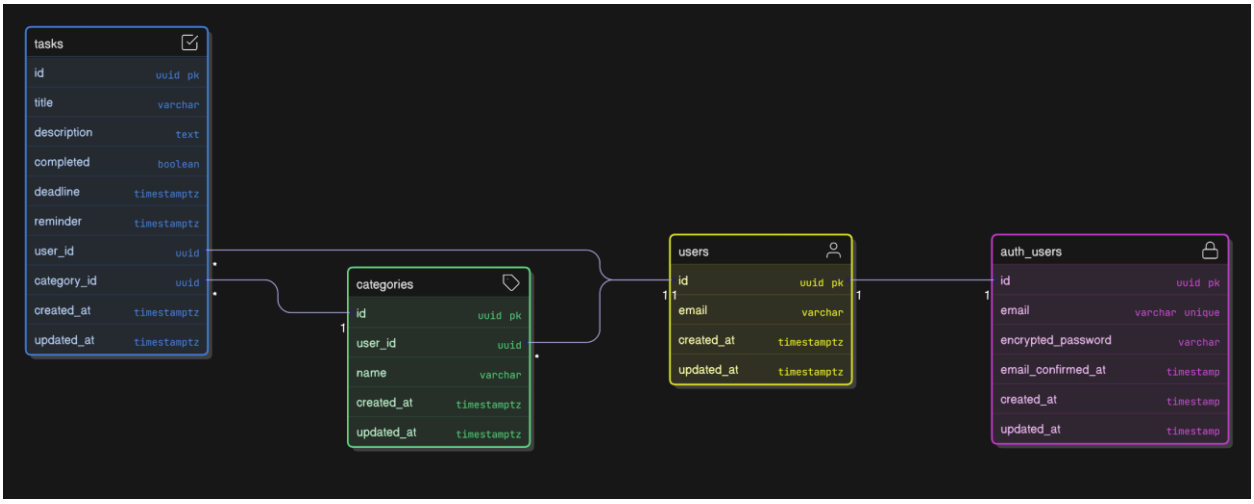
### d) Tailwind CSS

- *Utility-First Approach:* Rapid UI development and prototyping
- *Responsive Design:* Mobile-first methodology built-in
- *Theme Support:* Easy dark/light mode implementation
- *Optimized Bundle:* Only used utilities included in production
- *Design System:* Consistent spacing, typography, and color schemes

## 3. High-Level Architecture Diagram




## 4. Database Schema & Design



## 5. Detailed Schema

- auth.users (Managed by Supabase)

```
-- Built-in Supabase auth table
auth.users (
  id UUID PRIMARY KEY,
  email VARCHAR UNIQUE,
  encrypted_password VARCHAR,
  email_confirmed_at TIMESTAMP,
  created_at TIMESTAMP,
  updated_at TIMESTAMP,
  -- ... other auth fields
)
```

- users (Application user profile)

```
CREATE TABLE users (
  id UUID PRIMARY KEY REFERENCES auth.users(id) ON DELETE CASCADE,
  email VARCHAR NOT NULL,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

-- RLS Policies
ALTER TABLE users ENABLE ROW LEVEL SECURITY;
CREATE POLICY "Users can view own profile" ON users FOR SELECT USING
(auth.uid() = id);
CREATE POLICY "Users can update own profile" ON users FOR UPDATE USING
(auth.uid() = id);
```

- categories

```
CREATE TABLE categories (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  name VARCHAR(100) NOT NULL,
  user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  UNIQUE(name, user_id) -- Unique category name per user
);

-- RLS Policies
ALTER TABLE categories ENABLE ROW LEVEL SECURITY;
CREATE POLICY "Users can manage own categories" ON categories
  FOR ALL USING (auth.uid() = user_id);
```

- tasks

```sql
CREATE TABLE tasks (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  title VARCHAR(200) NOT NULL,
  description TEXT,
  completed BOOLEAN DEFAULT FALSE,
  deadline TIMESTAMP WITH TIME ZONE,
  reminder TIMESTAMP WITH TIME ZONE,
  user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
  category_id UUID REFERENCES categories(id) ON DELETE SET NULL,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

-- Indexes for performance
CREATE INDEX idx_tasks_user_id ON tasks(user_id);
CREATE INDEX idx_tasks_category_id ON tasks(category_id);
CREATE INDEX idx_tasks_reminder ON tasks(reminder) WHERE reminder IS NOT
NULL;
CREATE INDEX idx_tasks_deadline ON tasks(deadline) WHERE deadline IS NOT
NULL;

-- RLS Policies
ALTER TABLE tasks ENABLE ROW LEVEL SECURITY;
CREATE POLICY "Users can manage own tasks" ON tasks
  FOR ALL USING (auth.uid() = user_id);
```

## 6. Key Schema Features

### a) Security

- *Row-Level Security (RLS):* Database-enforced user isolation
- *UUID Primary Keys:* Non-guessable, secure identifiers
- *Foreign Key Constraints:* Data integrity

### b) Performance

- *Strategic Indexes:* Fast queries on user_id, dates, categories
- *Timestamp with Time Zone:* Proper timezone handling
- *Efficient Relationships:* Normalized structure

### c) Flexibility

- *Soft Category Deletion:* Tasks retain when category deleted
- *Nullable Fields:* Optional deadlines and reminders
- *Extensible:* Easy to add new fields (priority, tags, etc.)

## 7. Why Current Stack is Optimal for This Project

- *Rapid Development:* Full-stack in single codebase
- *Cost Effective:* Generous free tiers
- *Scalable:* Can handle significant growth
- *Maintainable:* Simple architecture, good DX
- *Production Ready:* Built-in optimizations and security
- The chosen stack perfectly balances development speed, maintainability, performance, and cost for a task management application that needs real-time features and can scale to thousands of users.

## Thank you all!