Asset Manager - Complete Project Documentation

📋 Project Overview

Project Name: Asset Manager
Tech Stack: Next.js 14+, PostgreSQL (Supabase), TypeScript, Tailwind CSS
Deployment: Vercel (Frontend) + Supabase (Database)
Version Control: GitHub with CI/CD pipeline

🎯 Core Requirements

1. Authentication & Authorization

User Roles

- **Admin Role**
  - Full system access
  - User management capabilities
  - Configuration management
  - View all assets across departments
- **User Role**
  - Create and manage own assets
  - View only self-created assets
  - Read-only access to categories and departments

Authentication Features

- Secure login/logout
- Role-based access control (RBAC)
- Session management
- Password hashing and security

---

2. Admin Functionalities

User Management

- Create new users with roles (Admin/User)
- Assign users to departments
- View list of all users
- Edit user details
- Deactivate/activate users

Asset Category Management

- Create asset categories (e.g., Electronics, Furniture, Vehicles)
- Edit existing categories
- View all categories
- Delete categories (with validation)

Department Management

- Create departments (e.g., IT, Finance, HR, Operations)
- Edit department details
- View all departments
- Delete departments (with validation)

Asset Management

- View all assets system-wide
- Delete any asset
- Filter assets by category, department, user
- Export asset reports

Admin Dashboard

- Total users count
- Total assets count
- Assets by category (chart/graph)
- Assets by department (chart/graph)
- Recent activity log

- Quick action buttons

---

3. User Functionalities

Asset Management

- Create new assets with required fields
- View only self-created assets
- Edit own assets
- Filter/search own assets

Asset Data Fields (Required)

- Asset Name (string, required)
- Category (dropdown, required)
- Date Purchased (date picker, required)
- Cost (decimal, required)
- Department (dropdown, required)
- Description (optional text area)
- Serial Number (optional)
- Status (Active/Inactive)

User Dashboard

- Total assets created by user
- Total value of assets
- Assets by category (pie chart)
- Recent assets list
- Quick create asset button

🏗️ Project Structure

```
asset-manager/
├── .github/
```

```
│   └── workflows/
│       └── ci-cd.yml              # GitHub Actions workflow
├── public/
│   ├── images/
│   └── favicon.ico
├── src/
│   ├── app/
│   │   ├── (auth)/
│   │   │   ├── login/
│   │   │   │   └── page.tsx
│   │   │   └── layout.tsx
│   │   ├── (dashboard)/
│   │   │   ├── admin/
│   │   │   │   ├── dashboard/
│   │   │   │   │   └── page.tsx
│   │   │   │   ├── users/
│   │   │   │   │   ├── page.tsx
│   │   │   │   │   ├── create/
│   │   │   │   │   │   └── page.tsx
│   │   │   │   │   └── [id]/
│   │   │   │   │       └── page.tsx
│   │   │   │   ├── categories/
│   │   │   │   │   ├── page.tsx
│   │   │   │   │   └── create/
│   │   │   │   │       └── page.tsx
│   │   │   │   ├── departments/
│   │   │   │   │   ├── page.tsx
│   │   │   │   │   └── create/
│   │   │   │   │       └── page.tsx
│   │   │   │   └── assets/
│   │   │   │       └── page.tsx
│   │   │   ├── user/
│   │   │   │   ├── dashboard/
│   │   │   │   │   └── page.tsx
│   │   │   │   └── assets/
│   │   │   │       ├── page.tsx
│   │   │   │       ├── create/
│   │   │   │       │   └── page.tsx
│   │   │   │       └── [id]/
│   │   │   │           ├── page.tsx
│   │   │   │           └── edit/
│   │   │   │               └── page.tsx
│   │   │   └── layout.tsx
│   │   ├── api/
```

```
│   │   │   ├── auth/
│   │   │   │   ├── login/
│   │   │   │   │   └── route.ts
│   │   │   │   └── logout/
│   │   │   │       └── route.ts
│   │   │   ├── users/
│   │   │   │   ├── route.ts
│   │   │   │   └── [id]/
│   │   │   │       └── route.ts
│   │   │   ├── assets/
│   │   │   │   ├── route.ts
│   │   │   │   └── [id]/
│   │   │   │       └── route.ts
│   │   │   ├── categories/
│   │   │   │   ├── route.ts
│   │   │   │   └── [id]/
│   │   │   │       └── route.ts
│   │   │   └── departments/
│   │   │       ├── route.ts
│   │   │       └── [id]/
│   │   │           └── route.ts
│   │   ├── layout.tsx
│   │   ├── page.tsx
│   │   └── globals.css
│   ├── components/
│   │   ├── ui/
│   │   │   ├── button.tsx
│   │   │   ├── input.tsx
│   │   │   ├── select.tsx
│   │   │   ├── table.tsx
│   │   │   ├── card.tsx
│   │   │   ├── dialog.tsx
│   │   │   └── toast.tsx
│   │   ├── layout/
│   │   │   ├── Navbar.tsx
│   │   │   ├── Sidebar.tsx
│   │   │   └── Footer.tsx
│   │   ├── auth/
│   │   │   ├── LoginForm.tsx
│   │   │   └── ProtectedRoute.tsx
│   │   ├── dashboard/
│   │   │   ├── StatsCard.tsx
│   │   │   ├── ChartWidget.tsx
│   │   │   └── RecentActivity.tsx
```

```
│   │   ├── assets/
│   │   │   ├── AssetForm.tsx
│   │   │   ├── AssetTable.tsx
│   │   │   ├── AssetCard.tsx
│   │   │   └── AssetFilters.tsx
│   │   ├── users/
│   │   │   ├── UserForm.tsx
│   │   │   └── UserTable.tsx
│   │   ├── categories/
│   │   │   ├── CategoryForm.tsx
│   │   │   └── CategoryList.tsx
│   │   └── departments/
│   │       ├── DepartmentForm.tsx
│   │       └── DepartmentList.tsx
│   ├── lib/
│   │   ├── supabase/
│   │   │   ├── client.ts
│   │   │   ├── server.ts
│   │   │   └── middleware.ts
│   │   ├── hooks/
│   │   │   ├── useAuth.ts
│   │   │   ├── useAssets.ts
│   │   │   ├── useUsers.ts
│   │   │   └── useDashboard.ts
│   │   ├── utils/
│   │   │   ├── validation.ts
│   │   │   ├── formatting.ts
│   │   │   └── constants.ts
│   │   └── types/
│   │       ├── database.types.ts
│   │       ├── user.types.ts
│   │       └── asset.types.ts
│   └── middleware.ts
├── supabase/
│   ├── migrations/
│   │   ├── 001_initial_schema.sql
│   │   ├── 002_add_rls_policies.sql
│   │   └── 003_seed_data.sql
│   └── config.toml
├── .env.local.example
├── .env.local
├── .gitignore
├── next.config.js
├── package.json
```

```
├── tsconfig.json
├── tailwind.config.js
├── postcss.config.js
└── README.md
```

🗄️ Database Schema

Tables

1. users

```sql
CREATE TABLE users (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  email VARCHAR(255) UNIQUE NOT NULL,
  password_hash VARCHAR(255) NOT NULL,
  full_name VARCHAR(255) NOT NULL,
  role VARCHAR(50) NOT NULL CHECK (role IN ('admin', 'user')),
  department_id UUID REFERENCES departments(id),
  is_active BOOLEAN DEFAULT true,
  created_at TIMESTAMP DEFAULT NOW(),
  updated_at TIMESTAMP DEFAULT NOW()
);
```

2. departments

```sql
CREATE TABLE departments (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  name VARCHAR(255) UNIQUE NOT NULL,
  description TEXT,
  created_at TIMESTAMP DEFAULT NOW(),
  updated_at TIMESTAMP DEFAULT NOW()
);
```

3. categories

```sql
```

```sql
CREATE TABLE categories (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  name VARCHAR(255) UNIQUE NOT NULL,
  description TEXT,
  created_at TIMESTAMP DEFAULT NOW(),
  updated_at TIMESTAMP DEFAULT NOW()
);
```

4. assets

```sql
CREATE TABLE assets (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  asset_name VARCHAR(255) NOT NULL,
  category_id UUID REFERENCES categories(id) NOT NULL,
  department_id UUID REFERENCES departments(id) NOT NULL,
  date_purchased DATE NOT NULL,
  cost DECIMAL(12, 2) NOT NULL,
  description TEXT,
  serial_number VARCHAR(255),
  status VARCHAR(50) DEFAULT 'active' CHECK (status IN ('active', 'inactive')),
  created_by UUID REFERENCES users(id) NOT NULL,
  created_at TIMESTAMP DEFAULT NOW(),
  updated_at TIMESTAMP DEFAULT NOW()
);
```

5. activity_logs

```sql
CREATE TABLE activity_logs (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  user_id UUID REFERENCES users(id),
  action VARCHAR(255) NOT NULL,
  entity_type VARCHAR(100),
  entity_id UUID,
  details JSONB,
  created_at TIMESTAMP DEFAULT NOW()
);
```

Indexes

```sql
CREATE INDEX idx_assets_created_by ON assets(created_by);
CREATE INDEX idx_assets_category ON assets(category_id);
CREATE INDEX idx_assets_department ON assets(department_id);
CREATE INDEX idx_users_email ON users(email);
CREATE INDEX idx_users_department ON users(department_id);
```

Row Level Security (RLS) Policies

```sql
```

```sql
-- Enable RLS
ALTER TABLE assets ENABLE ROW LEVEL SECURITY;
ALTER TABLE users ENABLE ROW LEVEL SECURITY;

-- Policy: Users can only view their own assets
CREATE POLICY "Users can view own assets" ON assets
  FOR SELECT
  USING (auth.uid() = created_by);

-- Policy: Admins can view all assets
CREATE POLICY "Admins can view all assets" ON assets
  FOR SELECT
  USING (
    EXISTS (
      SELECT 1 FROM users
      WHERE id = auth.uid() AND role = 'admin'
    )
  );

-- Policy: Users can insert assets
CREATE POLICY "Users can create assets" ON assets
  FOR INSERT
  WITH CHECK (auth.uid() = created_by);

-- Policy: Admins can delete assets
CREATE POLICY "Admins can delete assets" ON assets
  FOR DELETE
  USING (
    EXISTS (
      SELECT 1 FROM users
      WHERE id = auth.uid() AND role = 'admin'
    )
  );
```

🚀 Setup Instructions

Prerequisites

- Node.js 18+ installed
- Git installed
- GitHub account

- Vercel account (free tier)
- Supabase account (free tier)

Initial Setup

1. Create GitHub Repository

```bash
```

```
# Navigate to your project location
cd /path/to/your/projects

# Create project directory
mkdir asset-manager
cd asset-manager

# Initialize git
git init

# Create initial README
echo "# Asset Manager" > README.md

# Create .gitignore
cat > .gitignore << EOL
# Dependencies
node_modules/
.pnp
.pnp.js

# Testing
coverage/

# Next.js
.next/
out/
build/
dist/

# Environment variables
.env
.env.local
.env*.local

# Logs
npm-debug.log*
yarn-debug.log*
yarn-error.log*

# OS
.DS_Store
*.pem
```

```
# Debug
.vscode/
.idea/

# Vercel
.vercel

# TypeScript
*.tsbuildinfo
next-env.d.ts
EOL

# Initial commit
git add .
git commit -m "Initial commit"

# Create GitHub repo (via GitHub CLI or manually on GitHub.com)
# If using GitHub CLI:
gh repo create asset-manager --public --source=. --remote=origin

# Or manually add remote:
git remote add origin https://github.com/YOUR_USERNAME/asset-manager.git
git branch -M main
git push -u origin main
```

2. Clone on Different Machine

```
bash

# Clone the repository
git clone https://github.com/YOUR_USERNAME/asset-manager.git
cd asset-manager

# Install dependencies
npm install

# Create .env.local file
cp .env.local.example .env.local

# Edit .env.local with your credentials
nano .env.local
```

3. Initialize Next.js Project

```bash
# Create Next.js app with TypeScript
npx create-next-app@latest . --typescript --tailwind --app --src-dir

# Install additional dependencies
npm install @supabase/supabase-js @supabase/auth-helpers-nextjs
npm install recharts lucide-react
npm install -D @types/node
```

4. Setup Supabase

1. Go to https://supabase.com
2. Create new project
3. Wait for database to provision
4. Go to Project Settings > API
5. Copy the following:
   - Project URL
   - Anon/Public Key
   - Service Role Key (for admin operations)
6. Create `.env.local`:

```env
NEXT_PUBLIC_SUPABASE_URL=your_supabase_project_url
NEXT_PUBLIC_SUPABASE_ANON_KEY=your_supabase_anon_key
SUPABASE_SERVICE_ROLE_KEY=your_service_role_key
```

7. Run migrations in Supabase SQL Editor:
   - Copy schema from Database Schema section
   - Execute in SQL Editor

5. Setup Vercel

1. Go to https://vercel.com
2. Click "Import Project"
3. Import your GitHub repository

4. Configure project:
    - Framework Preset: Next.js
    - Root Directory: ./
    - Build Command: `npm run build`
    - Output Directory: .next
5. Add Environment Variables:
    - Add all variables from `.env.local`
6. Deploy

6. Configure GitHub Integration

Vercel automatically sets up GitHub integration. Every push to `main` will trigger deployment.

To verify:

1. Go to Vercel Dashboard > Your Project > Settings > Git
2. Ensure "Automatic Deployments" is enabled
3. Production Branch should be `main`

🔄 CI/CD Pipeline Configuration

GitHub Actions Workflow

Create `.github/workflows/ci-cd.yml`:

```yaml
```

```yaml
name: CI/CD Pipeline

on:
  push:
    branches: [main, develop]
  pull_request:
    branches: [main, develop]

jobs:
  test:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'

      - name: Install dependencies
        run: npm ci

      - name: Run linter
        run: npm run lint

      - name: Type check
        run: npm run type-check

      - name: Run tests
        run: npm run test
        env:
          NEXT_PUBLIC_SUPABASE_URL: ${{ secrets.NEXT_PUBLIC_SUPABASE_URL }}
          NEXT_PUBLIC_SUPABASE_ANON_KEY: ${{ secrets.NEXT_PUBLIC_SUPABASE_ANON_KEY }}

  build:
    needs: test
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
```

```yaml
      uses: actions/checkout@v3

    - name: Setup Node.js
      uses: actions/setup-node@v3
      with:
        node-version: '18'
        cache: 'npm'

    - name: Install dependencies
      run: npm ci

    - name: Build application
      run: npm run build
      env:
        NEXT_PUBLIC_SUPABASE_URL: ${{ secrets.NEXT_PUBLIC_SUPABASE_URL }}
        NEXT_PUBLIC_SUPABASE_ANON_KEY: ${{ secrets.NEXT_PUBLIC_SUPABASE_ANON_KEY }}

    - name: Upload build artifacts
      uses: actions/upload-artifact@v3
      with:
        name: build
        path: .next

  deploy:
    needs: build
    runs-on: ubuntu-latest
    if: github.ref == 'refs/heads/main'

    steps:
      - name: Deploy to Vercel
        run: echo "Deployment handled by Vercel GitHub integration"
```

Add Scripts to package.json

```json
```

```
{
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "type-check": "tsc --noEmit",
    "test": "echo \"Tests will be added\" && exit 0"
  }
}
```

Configure GitHub Secrets

1. Go to GitHub Repository > Settings > Secrets and variables > Actions
2. Add the following secrets:
   - `NEXT_PUBLIC_SUPABASE_URL`
   - `NEXT_PUBLIC_SUPABASE_ANON_KEY`
   - `SUPABASE_SERVICE_ROLE_KEY`

Environment Variables

Development (.env.local)

```
env

# Supabase
NEXT_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key

# App Configuration
NEXT_PUBLIC_APP_URL=http://localhost:3000
NODE_ENV=development
```

Production (Vercel)

Same variables as development, but with production URLs.

📝 Development Workflow

Working on Features

```bash
# Create feature branch
git checkout -b feature/asset-creation

# Make changes and commit regularly
git add .
git commit -m "feat: add asset creation form"

# Push to GitHub
git push origin feature/asset-creation

# Create Pull Request on GitHub
# After review, merge to main
# Vercel automatically deploys
```

Syncing Across Machines

```bash
# On Machine A - Push changes
git add .
git commit -m "feat: update dashboard"
git push origin main

# On Machine B - Pull changes
git pull origin main
npm install  # If dependencies changed
```

✨ Innovative Features to Add

1. **Asset QR Code Generation** - Generate QR codes for physical asset tracking
2. **Export Reports** - Export assets to PDF/CSV
3. **Asset History Timeline** - Track all changes made to an asset
4. **Dashboard Charts** - Visualize asset distribution with charts
5. **Search & Filters** - Advanced filtering by multiple criteria

6. **Bulk Upload** - Import assets via CSV
7. **Email Notifications** - Notify admins of new asset creation
8. **Asset Depreciation Calculator** - Calculate asset value over time
9. **Mobile Responsive Design** - Ensure perfect mobile experience
10. **Dark Mode** - Theme toggle for user preference

🧪 Testing Checklist

Authentication

- [ ] Login as Admin
- [ ] Login as User
- [ ] Logout functionality
- [ ] Session persistence
- [ ] Unauthorized access prevention

Admin Features

- [ ] Create user
- [ ] Create category
- [ ] Create department
- [ ] Delete asset
- [ ] View all assets
- [ ] View dashboard statistics

User Features

- [ ] Create asset
- [ ] View own assets only
- [ ] Edit own asset
- [ ] Cannot access admin routes
- [ ] View dashboard with own statistics

CI/CD

- [ ] Push to GitHub triggers build
- [ ] Tests run automatically
- [ ] Deployment succeeds on Vercel
- [ ] Environment variables work in production

📚 Resources

- Next.js Documentation
- Supabase Documentation
- Vercel Documentation
- Tailwind CSS
- TypeScript Handbook

🎯 Submission Checklist

- [ ] Application deployed and accessible
- [ ] GitHub repository is public/accessible
- [ ] README.md with setup instructions
- [ ] Eport team has write access to repository
- [ ] CI/CD pipeline working
- [ ] All core features implemented
- [ ] Responsive design
- [ ] Error handling implemented
- [ ] Security best practices followed

Good luck with your development! 🚀