Basil - Cultural Cards App

Project Structure

```
basil/
├── public/
│   └── index.html
├── src/
│   ├── components/
│   │   ├── Auth/
│   │   │   ├── Login.jsx
│   │   │   └── SignUp.jsx
│   │   ├── Cards/
│   │   │   ├── CardGrid.jsx
│   │   │   ├── CardItem.jsx
│   │   │   └── CardDetail.jsx
│   │   ├── Forms/
│   │   │   ├── AddCardForm.jsx
│   │   │   └── ImageUpload.jsx
│   │   ├── Layout/
│   │   │   ├── Navbar.jsx
│   │   │   └── Footer.jsx
│   │   └── Music/
│   │       └── AudioPlayer.jsx
│   ├── lib/
│   │   ├── supabase.js
│   │   └── constants.js
│   ├── hooks/
│   │   ├── useAuth.js
│   │   └── useCards.js
│   ├── App.jsx
│   ├── index.css
│   └── main.jsx
├── .env.example
├── package.json
└── README.md
```

Database Schema (Supabase)

Tables

users (handled by Supabase Auth)
- id (uuid, primary key)
- email
- created_at

cards
- id (uuid, primary key)
- user_id (uuid, foreign key → users.id)
- title (text)

- description (text)
- image_url (text)
- music_url (text, nullable)
- music_file_url (text, nullable)
- category (text, nullable)
- card_id (text, unique - for QR codes like "001", "002")
- created_at (timestamp)
- updated_at (timestamp)

Storage Buckets

- **card-images**: For uploaded illustrations
- **card-music**: For uploaded MP3 files

Row Level Security (RLS) Policies

cards table:
- SELECT: Anyone can view (public)
- INSERT: Authenticated users only
- UPDATE: Only card owner
- DELETE: Only card owner

Setup Instructions

1. Create Supabase Project
```bash
# Go to supabase.com
# Create new project
# Note your project URL and anon key
```

2. Set up Database
Run this SQL in Supabase SQL Editor:

```sql
-- Create cards table
CREATE TABLE cards (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  user_id UUID REFERENCES auth.users(id) ON DELETE CASCADE,
  title TEXT NOT NULL,
  description TEXT NOT NULL,
  image_url TEXT NOT NULL,
  music_url TEXT,
  music_file_url TEXT,
  category TEXT,
  card_id TEXT UNIQUE NOT NULL,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

-- Enable RLS
ALTER TABLE cards ENABLE ROW LEVEL SECURITY;

-- Public read access
CREATE POLICY "Anyone can view cards"
```

```
  ON cards FOR SELECT
  TO public
  USING (true);

-- Authenticated users can insert
CREATE POLICY "Authenticated users can insert cards"
  ON cards FOR INSERT
  TO authenticated
  WITH CHECK (auth.uid() = user_id);

-- Users can update their own cards
CREATE POLICY "Users can update own cards"
  ON cards FOR UPDATE
  TO authenticated
  USING (auth.uid() = user_id)
  WITH CHECK (auth.uid() = user_id);

-- Users can delete their own cards
CREATE POLICY "Users can delete own cards"
  ON cards FOR DELETE
  TO authenticated
  USING (auth.uid() = user_id);

-- Create index for faster queries
CREATE INDEX cards_card_id_idx ON cards(card_id);
CREATE INDEX cards_user_id_idx ON cards(user_id);
CREATE INDEX cards_category_idx ON cards(category);
```

3. Set up Storage
In Supabase Dashboard → Storage:
1. Create bucket: `card-images` (public)
2. Create bucket: `card-music` (public)

4. Initialize React Project
```bash
npm create vite@latest basil -- --template react
cd basil
npm install @supabase/supabase-js
npm install react-router-dom
npm install lucide-react
npm install
```

5. Environment Variables
Create `.env` file:
```
VITE_SUPABASE_URL=your_supabase_url
VITE_SUPABASE_ANON_KEY=your_supabase_anon_key
```

6. Deploy to Netlify
```bash
npm run build
# Drag the 'dist' folder to Netlify
```

````
# Or connect your GitHub repo
# Add environment variables in Netlify dashboard
```

## Features Implementation Order

1. ✅ Supabase setup & auth
2. ✅ Basic card display (read from database)
3. ✅ User authentication (login/signup)
4. ✅ Add card form (with image upload)
5. ✅ Music support (URL embed + file upload)
6. ✅ Delete own cards
7. ✅ Card detail view (?id=XXX routing)
8. ✅ QR code generation
9. ⏳ Categories (optional)
10. ⏳ Search/filter (optional)

## QR Code Generation

Use `qrcode` library:
```bash
npm install qrcode
```

Generate QR codes for each card:
```javascript
import QRCode from 'qrcode';

const generateQR = async (cardId) => {
  const url = `https://reyhanlikartlar.netlify.app/?id=${cardId}`;
  const qrCodeDataURL = await QRCode.toDataURL(url);
  return qrCodeDataURL;
};
```

## Next Steps

I'll provide the code artifacts in the following order:
1. Supabase configuration
2. Core hooks (auth, cards)
3. Main components
4. App structure
````
````