Local Development Setup Guide

This guide provides step-by-step instructions to set up the Season Analysis Application on your local machine for development.

Prerequisites

Before you begin, ensure you have the following installed on your system:

- **Node.js** (version 18 or higher)
- **npm** (comes with Node.js)
- **MySQL** (version 8.0 or higher)
- **Git**
- **Google Gemini API Key** (get from Google AI Studio)
- **Code Editor** (VS Code recommended)

You MIGHT NEED TO ADD THEM TO YOUR ENVIRONMENT VARIABLES (ON WINDOWS).

Step 1: Clone the Repository

Open your terminal and clone the repository:

```
git clone https://github.com/orsiczako/season-analysis.git
cd season-analysis
```

Step 2: Database Setup

2.1 Start MySQL Server

Make sure your MySQL server is running on your local machine.

2.2 Create Database

The database must be created manually before running the application. Open MySQL command line or a database management tool (like MySQL Workbench, DBeaver, or phpMyAdmin) and execute:

```
CREATE DATABASE season_analysis CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;
```

Important: The database tables (account, color_seasons, favorite_colors) will be created automatically by Sequelize when you first start the backend server. However, the color_seasons table will be empty and needs to be populated manually with the four seasons.

You can use your existing MySQL root user for local development. The connection details will be configured in the backend `.env` file in the next step.

2.3 Populate Color Seasons Table

After the backend server creates the tables, you need to manually insert the four color seasons. Execute this SQL:

```
USE season_analysis;

INSERT INTO color_seasons (name, description, createdAt, updatedAt) VALUES
('spring', 'Warm, bright, and clear colors. Best for people with golden
undertones.', NOW(), NOW()),
('summer', 'Cool, soft, and muted colors. Best for people with blue
undertones.', NOW(), NOW()),
('autumn', 'Warm, deep, and muted colors. Best for people with golden
undertones.', NOW(), NOW()),
('winter', 'Cool, bright, and clear colors. Best for people with blue
undertones.', NOW(), NOW());
```

Note: You can do this step after starting the backend server for the first time, or before if you prefer.

Step 3: Backend Setup

3.1 Navigate to Backend Directory

```
cd Backend
```

3.2 Install Dependencies

```
npm install
```

3.3 Create Environment File

Create a new file named `.env` in the Backend directory:

```
# On Windows PowerShell
New-Item .env

# On macOS/Linux
touch .env
```

3.4 Configure Environment Variables

Open the `.env` file and add the following configuration:

```
# Server Configuration
PORT=3000
NODE_ENV=development

# Database Configuration
DB_HOST=localhost
DB_PORT=3306
DB_NAME=season_analysis
DB_USER=season_user
DB_PASSWORD=your_password_here

# JWT Authentication
JWT_SECRET=your_super_secret_jwt_key_change_this_in_production

# Google Gemini API
GEMINI_API_KEY=your_gemini_api_key_here

# Email Configuration (for local password recovery)
SMTP_HOST=smtp.gmail.com
SMTP_PORT=587
SMTP_USER=your_email@gmail.com
SMTP_PASSWORD=your_app_password_here
EMAIL_FROM=your_email@gmail.com

# Frontend URL (for CORS)
FRONTEND_URL=http://localhost:5173
```

Important Notes:

- Replace `your_password_here` with your MySQL password
- Replace `your_super_secret_jwt_key_change_this_in_production` with a random secure string, you should generate one
- Replace `your_gemini_api_key_here` with your Google Gemini API key
- For Gmail SMTP, you need to use an App Password (not your regular password)

3.5 Initialize Database Tables

The application will automatically create tables and seed initial data on first run. Start the backend server:

```
npm start
```

You should see output indicating:

- Database connection established

- Tables created (account, color_seasons, favorite_colors)
- Color seasons seeded (Spring, Summer, Autumn, Winter)
- Server listening on port 3000

3.6 Verify Backend is Running

Open your browser and navigate to:

```
http://localhost:3000
```

You should see a basic response or the frontend if static files are being served.

Step 4: Frontend Setup

4.1 Open New Terminal Window

Keep the backend server running and open a new terminal window.

4.2 Navigate to Frontend Directory

```
cd Frontend
```

If you're in the Backend directory:

```
cd ../Frontend
```

4.3 Install Dependencies

```
npm install
```

4.4 Create Environment File (Optional)

Create a `.env` file in the Frontend directory if you need to customize the API URL:

```
VITE_API_URL=http://localhost:3000
```

By default, the frontend is configured to use `http://localhost:3000` for development.

4.5 Start Development Server

```
npm run dev
```

You should see output indicating:

```
VITE v5.x.x  ready in xxx ms

➜  Local:   http://localhost:5173/
➜  Network: use --host to expose
```

Additional Configuration

Setting Up Gmail App Password

1. Go to Google Account settings
2. Navigate to Security > 2-Step Verification
3. Scroll down to App passwords
4. Generate a new app password for "Mail"
5. Copy the 16-character password
6. Use this password in SMTP_PASSWORD in `.env`

Getting Google Gemini API Key

1. Visit Google AI Studio: https://makersuite.google.com/app/apikey
2. Sign in with your Google account
3. Click "Create API Key"
4. Copy the API key
5. Paste it in GEMINI_API_KEY in `.env`
