**1. Tech Stack**

* **Framework: ASP.NET Core 8 MVC**
* **ORM: Entity Framework Core**
* **Database: SQLite**
* **Authentication:**
  + **Custom login (email/password + OTP)**
  + **Google OAuth (with manual “Confirm” step)**
* **Token Handling: System.IdentityModel.Tokens.Jwt (for OTP & password reset links)**
* **Views: Razor + custom CSS**
* **Areas: Admin, User**
* **Email: SMTP or any external service (e.g. SendGrid)**

**2. User Types & Access Levels**

| **User Type** | **Dashboard** | **Area** | **Notes** |
| --- | --- | --- | --- |
| **Anonymous** | **❌** | **Public** | **Can only read public poems** |
| **Authenticated** | **✅** | **/User** | **Must verify via OTP or “Confirm” after OAuth** |
| **Admin** | **✅ (full)** | **/Admin** | **Manage users, poems, view stats** |

**3. Project Structure**

**pgsql**

**CopyEdit**

**VerseCraft/**

**│**

**├── Areas/**

**│ ├── Admin/ # Admin controllers & views**

**│ └── User/ # User-specific controllers & views**

**│**

**├── Controllers/**

**│ ├── AuthController.cs**

**│ ├── HomeController.cs**

**│ └── PoemController.cs**

**│**

**├── Models/**

**│ ├── User.cs**

**│ ├── Poem.cs**

**│ ├── Comment.cs**

**│ └── OTPToken.cs**

**│**

**├── ViewModels/**

**│ ├── RegisterViewModel.cs**

**│ ├── LoginViewModel.cs**

**│ └── VerifyOtpViewModel.cs**

**│**

**├── Services/**

**│ ├── OTPService.cs**

**│ └── EmailService.cs**

**│**

**├── Data/**

**│ └── VerseCraftDbContext.cs**

**│**

**├── Views/**

**│ ├── Shared/**

**│ ├── Auth/**

**│ │ ├── Register.cshtml**

**│ │ ├── Login.cshtml**

**│ │ └── VerifyOtp.cshtml**

**│ ├── Poem/**

**│ └── Home/**

**│**

**└── appsettings.json**

**4. Database Design & Relationships**

**4.1 Tables**

**Users**

| **Column** | **Type** | **Constraints** |
| --- | --- | --- |
| **Id** | **INTEGER** | **PK, Auto-increment** |
| **Name** | **TEXT** | **Nullable, Max 100** |
| **Email** | **TEXT** | **Required, Unique, Max 150** |
| **PasswordHash** | **TEXT** | **Nullable (for Google users), Max 255** |
| **IsVerified** | **BOOLEAN** | **Default false** |
| **IsAdmin** | **BOOLEAN** | **Default false** |
| **CreatedAt** | **DATETIME** | **Default = CURRENT\_TIMESTAMP** |

**Poems**

| **Column** | **Type** | **Constraints** |
| --- | --- | --- |
| **Id** | **INTEGER** | **PK, Auto-increment** |
| **Title** | **TEXT** | **Required, Max 200** |
| **Content** | **TEXT** | **Required** |
| **CreatedAt** | **DATETIME** | **Default = CURRENT\_TIMESTAMP** |
| **UserId** | **INTEGER** | **FK → Users(Id), OnDelete Cascade** |
| **IsPublic** | **BOOLEAN** | **Default true** |

**Comments**

| **Column** | **Type** | **Constraints** |
| --- | --- | --- |
| **Id** | **INTEGER** | **PK, Auto-increment** |
| **Text** | **TEXT** | **Required, Max 500** |
| **CreatedAt** | **DATETIME** | **Default = CURRENT\_TIMESTAMP** |
| **UserId** | **INTEGER** | **FK → Users(Id), OnDelete Cascade** |
| **PoemId** | **INTEGER** | **FK → Poems(Id), OnDelete Cascade** |

**OTPToken**

| **Column** | **Type** | **Constraints** |
| --- | --- | --- |
| **Id** | **INTEGER** | **PK, Auto-increment** |
| **Token** | **TEXT** | **Required, Max 10** |
| **Type** | **INTEGER** | **Required (0=AccountVerification,1=PasswordReset)** |
| **Expiry** | **DATETIME** | **Required** |
| **UserId** | **INTEGER** | **FK → Users(Id), OnDelete Cascade** |

**4.2 Relationships**

* **User ↔ Poem: 1 User has many Poems**
* **User ↔ Comment: 1 User has many Comments**
* **Poem ↔ Comment: 1 Poem has many Comments**
* **User ↔ OTPToken: 1 User has many OTPTokens**

**5. Authentication Flows**

**5.1 Manual Registration & OTP Verification**

1. **User submits Name/Email/Password.**
2. **System hashes password (BCrypt), creates User (IsVerified=false).**
3. **Generate OTP (OTPService), store in OTPToken, send via email.**
4. **User enters OTP on VerifyOtp page.**
5. **On success: set User.IsVerified=true, delete OTP, redirect to login.**

**5.2 Google OAuth + Confirm Button**

1. **User clicks “Login with Google” → OAuth flow → get email/name.**
2. **If new: create User (IsVerified=false).**
3. **Redirect to Confirm page with “Confirm” button.**
4. **On click: set User.IsVerified=true, redirect to dashboard.**

**5.3 Forgot Password**

1. **User requests password reset (enter email).**
2. **Generate JWT reset token, email link to user.**
3. **Link opens ResetPassword form.**
4. **On submit: verify JWT, hash new password, save to User.PasswordHash.**

**6. JWT Claims Usage**

* **Purpose: Email verification / Password reset**
* **Claims:**
  + **email**
  + **exp (expiry)**
  + **purpose (VerifyEmail / ResetPassword)**
  + **jti (unique token ID)**

**7. Validation Rules**

* **Email: Required, unique, valid format, max 150**
* **Password: Min 6 chars, strong regex (upper, lower, digit, special)**
* **Name: Optional, max 100**
* **Poem.Title: Required, max 200**
* **Poem.Content: Required**
* **Comment.Text: Required, max 500**

**8. Admin Dashboard**

* **Manage Users: Promote to Admin, revoke access, delete accounts**
* **Moderate Poems: Approve, edit, delete**
* **View Stats: Total users, poems, comments, daily activity**

**9. Access Constraints**

| **Action** | **Anonymous** | **Authenticated** | **Admin** |
| --- | --- | --- | --- |
| **View public poems** | **✅** | **✅** | **✅** |
| **Submit poem** | **❌** | **✅ (IsVerified == true)** | **✅** |
| **Comment on poem** | **❌** | **✅ (IsVerified == true)** | **✅** |
| **View dashboard** | **❌** | **✅ (IsVerified == true)** | **✅** |
| **Manage users/poems** | **❌** | **❌** | **✅** |

**Next Steps**

* **Scaffold DbContext & run migrations**
* **Build AuthController (manual & Google flows)**
* **Create Views for Register, Verify OTP, Login, Confirm, ResetPassword**
* **Implement EmailService for OTP & reset links**
* **Protect routes with Role/Verification checks in controllers**

**Let me know which part you’d like to tackle first—step by step!**