UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA
UNAN – LEÓN

AÑO LECTIVO: 2025
SEMESTRE: II

Componente: Programacion Orientada a la Web II:

Grupo: 1
Nombre del docente: Juan Carlos Leyton Briones
Proyecto de encuestas.
Integrantes:

Ismael Alberto Martinez Lacayo *22-06305-0*

León, Nicaragua, 2025.

UmaTeams

UmaTeams es una aplicación web desarrollada en ASP.NET Core MVC que permite a los usuarios explorar, buscar y gestionar información sobre los personajes del universo de **Umamusume**.

Descripción del Proyecto

Este proyecto fue creado tomando como base la Tarea 2 del componente. Su objetivo principal es proporcionar una plataforma donde los usuarios puedan consultar datos sobre personajes de Umamusume, incluyendo sus nombres en inglés y japonés, categoría y colores distintivos.

Funcionalidades Clave

- **Búsqueda y listado de personajes**: Los usuarios pueden buscar personajes de Umamusume y ver detalles completos de cada uno.
- **Gestión de equipos:** Permite la creación y administración de equipos personalizados con personajes seleccionados.
- **Detalles enriquecidos**: Cada personaje muestra información relevante y visual (imágenes, colores, categorías).
- **Autenticación de usuarios**: Sistema de login integrado usando ASP.NET Identity.
- **Responsive Design**: Utiliza Bootstrap 5 para asegurar una experiencia óptima en cualquier dispositivo.

Arquitectura y Estructura

- **Backend**: ASP.NET Core MVC + Entity Framework Core + MySQL.
- **Frontend**: HTML5, Bootstrap 5, Razor Views (Como algo provisional antes de React).
- **Gestión de datos:** Base de datos relacional, con migraciones y modelo de usuario personalizado.
- **Seed dinámico:** Opción para poblar la base de datos desde una API externa (ver método SeedFromApiAsync en AppDbContext).

Principales componentes

- **Data/AppDbContext.cs:** Contexto de la base de datos, define las tablas principales (CharacterInfo, UmaTeams, TeamMembers) y la gestión de usuarios.
- Models/: Modelos para vistas y entidad de usuario.
- **Views/Uma/Details.cshtml:** Página de detalles del personaje, incluye descripción, funcionalidades, y herramientas usadas.
- **Views/Uma/Find.cshtml:** Página de búsqueda avanzada de personajes.
- **Program.cs:** Configuración principal de la aplicación, conexión a la base de datos y sistema de autenticación.
- **Migrations/:** Migraciones de EF Core para la estructura de la base de datos.

Herramientas Utilizadas

- ASP.NET Core MVC
- Entity Framework Core
- Bootstrap 5
- MySQL
- jQuery y plugins de validación (ver licencias en /wwwroot/lib/)
- Sistema de autenticación ASP.NET Identity

Instalación y Configuración

1. **Clonar el repositorio:**

git clone https://github.com/eltotha/UmaTeams.git

2. **Configura la cadena de conexión a MySQL en el archivo appsettings.json:**

```
"ConnectionStrings": {
  "DefaultConnection":
"Server=localhost;Database=UmaTeamsDb;User=root;Password=yourpassword;
"
}
```

3. **Ejecuta las migraciones para crear la base de datos:**

dotnet ef database update

4. **Inicia la aplicación:**

dotnet run

5. **Accede a la dirección indicada por el proyecto en el navegador.**

API Usadas

Para este proyecto se uso la api publica de umapyoi.net para la inserccion de datos en la tabla characterinfo mediante las seeds:

https://umapyoi.net/docs/index.html

Base de Datos UmaTeams

establece la base de datos principal y las tablas necesarias para la aplicación UmaTeams, cuyo propósito es gestionar información sobre personajes de Umamusume y equipos de usuarios, incluyendo autenticación y roles mediante ASP.NET Core Identity.

Tablas Creadas

1. **CharacterInfo:**

Contiene información de los personajes de Umamusume.

- Campos principales: id, nombre en inglés y japonés, categoría, colores, imagen, etc.
- Índices: Para búsqueda rápida por nombre en inglés y japonés.

2. Tablas de Identidad (ASP.NET Core Identity):

Permiten la autenticación y autorización de usuarios.

AspNetRoles: Roles de usuario (Admin, User, Moderator).

AspNetRoleClaims: Claims asociados a roles.

AspNetUsers: Usuarios registrados, con campos para autenticación y perfil.

AspNetUserClaims: Claims de usuario.

AspNetUserLogins: Proveedores externos de login.

AspNetUserRoles: Relación usuarios-roles.

AspNetUserTokens: Tokens de autenticación y recuperación.

Incluyen índices para mejorar el rendimiento en búsquedas y validaciones.

3. UmaTeams:

Registra los equipos creados por usuarios.

- Campos principales: Id, TeamName

4. TeamMembers:

Relaciona los personajes (CharacterInfo) con los equipos (UmaTeams).

- Campos principales: Id, TeamId (FK), UmaId (FK), nombre e imagen del personaje en el equipo.
- Relaciones: Llaves foráneas con cascada para mantener la integridad referencial.

Indices

Se crean diversos índices para acelerar búsquedas y consultas frecuentes en tablas clave.

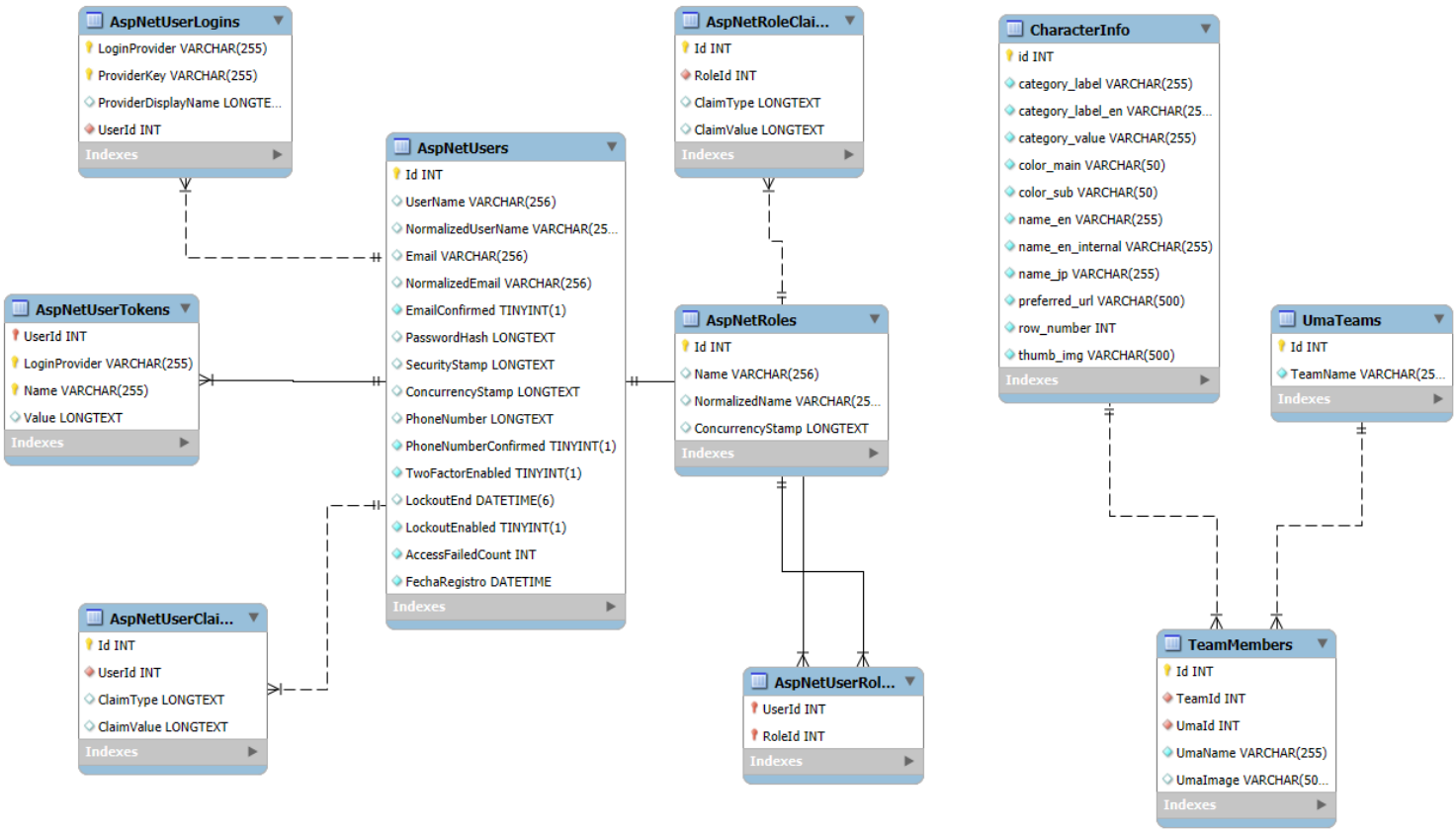
Datos Iniciales

- **Roles del sistema:** Admin, User, Moderator.
- **Usuario administrador:** Usuario "admin" con rol "Admin" y contraseña hash.
- **Equipos de ejemplo:** Tres equipos de muestra para pruebas y desarrollo.

Relaciones entre tablas

1. **AspNetRoles ←→ AspNetRoleClaims**
    - Relación 1:N. Cada rol puede tener múltiples claims.
    - FK: AspNetRoleClaims.RoleId → AspNetRoles.Id
2. **AspNetUsers ←→ AspNetUserClaims**
    - Relación 1:N. Cada usuario puede tener múltiples claims.
    - FK: AspNetUserClaims.UserId → AspNetUsers.Id
3. **AspNetUsers ←→ AspNetUserLogins**
    - Relación 1:N. Cada usuario puede tener múltiples logins externos.
    - FK: AspNetUserLogins.UserId → AspNetUsers.Id
4. **AspNetUsers ←→ AspNetUserRoles ←→ AspNetRoles**
    - Relación N:M entre usuarios y roles, resuelta por la tabla AspNetUserRoles.
    - FK: AspNetUserRoles.UserId → AspNetUsers.Id
    - FK: AspNetUserRoles.RoleId → AspNetRoles.Id
5. **AspNetUsers ←→ AspNetUserTokens**
    - Relación 1:N. Cada usuario puede tener múltiples tokens.
    - FK: AspNetUserTokens.UserId → AspNetUsers.Id
6. **UmaTeams ←→ TeamMembers**
    - Relación 1:N. Cada equipo puede tener múltiples miembros.
    - FK: TeamMembers.TeamId → UmaTeams.Id
7. **CharacterInfo ←→ TeamMembers**
    - Relación 1:N. Cada personaje puede estar en múltiples equipos.
    - FK: TeamMembers.UmaId → CharacterInfo.id

Modelo Entidad Relación



Repositorio de GitHub

https://github.com/eltotha/UmaTeams