Instituto Politécnico Nacional
Escuela Superior de Cómputo

Guía Técnica Básica

Proyecto: Sistema Médico

Unidad de Aprendizaje: Ingeniería en Software
Profesor: Gabriel Hurtado Avilés

Integrantes del Equipo:
Contla Mota Luis Andrés
Escárcega Hernández Steven Arturo
Gómez Sanchez Héctor Rodrigo
García Mayorga Rodrigo

1. Descripción General del Sistema

1.1 Arquitectura

Framework: ASP.NET Core 9.0
Patrón de Diseño: MVC (Model-View-Controller)
Lenguaje: C#
Front-end: Razor Views, Bootstrap, jQuery
Containerización: Docker

1.2 Estructura del Proyecto

```
ProyectoIS/
├── Controllers/
│   └── HomeController.cs
├── Models/
├── Views/
├── wwwroot/
├── Properties/
│   └── launchSettings.json
└── Program.cs
```

2. Implementación del Login y Manejo de Roles

2.1 Configuración de Identity

```
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddDbContext<ApplicationDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

builder.Services.AddDefaultIdentity<IdentityUser>(options => {
    options.Password.RequireDigit = true;
    options.Password.RequiredLength = 8;
    options.Password.RequireNonAlphanumeric = true;
    options.SignIn.RequireConfirmedAccount = true;
})
.AddRoles<IdentityRole>()
.AddEntityFrameworkStores<ApplicationDbContext>();
```

2.2 Modelo de Usuario

```
public class ApplicationUser : IdentityUser
{
    public string Nombre { get; set; }
    public string Apellidos { get; set; }
    public DateTime FechaRegistro { get; set; }
    public string? Especialidad { get; set; }
}
```

3. Conexión a Base de Datos

3.1 Cadena de Conexión

```
{
  "ConnectionStrings": {
    "DefaultConnection":
"Server=(localdb)\\mssqllocaldb;Database=SistemaMedico;Trusted_Connection=True;MultipleActiveResultSets=true"
  }
}
```

3.2 Contexto de Base de Datos

```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
        : base(options)
    {
    }

    public DbSet<Paciente> Pacientes { get; set; }
    public DbSet<Cita> Citas { get; set; }
    public DbSet<HistorialMedico> HistorialesMedicos { get; set; }
}
```

4. Configuraciones de ASP.NET Core

4.1 Configuración de Servicios

```
builder.Services.AddControllersWithViews();
builder.Services.AddRazorPages();

// Configuración de Cookies
builder.Services.ConfigureApplicationCookie(options =>
{
    options.Cookie.HttpOnly = true;
    options.ExpireTimeSpan = TimeSpan.FromMinutes(30);
    options.LoginPath = "/Identity/Account/Login";
    options.AccessDeniedPath = "/Identity/Account/AccessDenied";
    options.SlidingExpiration = true;
});

// Configuración de CORS
builder.Services.AddCors(options =>
{
    options.AddPolicy("AllowSpecificOrigins",
        builder => builder.WithOrigins("https://localhost:7073")
                          .AllowAnyMethod()
                          .AllowAnyHeader());
});
```

4.2 Configuración de Middleware

```
var app = builder.Build();

if (app.Environment.IsDevelopment())
{
    app.UseDeveloperExceptionPage();
}
else
{
    app.UseExceptionHandler("/Home/Error");
    app.UseHsts();
}
```

```
app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();
app.UseAuthentication();
app.UseAuthorization();

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");
```

5. Seguridad

5.1 Políticas de Autorización

```
builder.Services.AddAuthorization(options =>
{
    options.AddPolicy("RequireAdminRole", policy =>
        policy.RequireRole("Admin"));
    options.AddPolicy("RequireMedicoRole", policy =>
        policy.RequireRole("Medico"));
});
```

5.2 Protección contra CSRF

```
builder.Services.AddAntiforgery(options =>
{
    options.Cookie.Name = "X-CSRF-TOKEN";
    options.HeaderName = "X-CSRF-TOKEN";
});
```

6. Docker

6.1 Configuración de Docker

El proyecto incluye soporte para Docker con las siguientes configuraciones:

- Puerto HTTP: 8080
- Puerto HTTPS: 8081
- SSL habilitado

Para ejecutar:

```
docker build -t sistema-medico .
docker run -p 8080:8080 -p 8081:8081 sistema-medico
```

7. Requisitos del Sistema

- .NET 9.0 SDK o superior
- SQL Server 2019 o superior
- Visual Studio 2022 o VS Code con C# extensions
- Docker Desktop (opcional)

8. Comandos Útiles

```
# Actualizar base de datos
dotnet ef database update

# Ejecutar en desarrollo
dotnet run --launch-profile "https"

# Publicar aplicación
dotnet publish -c Release
```