Universidad ORT Uruguay

Segunda parte del Obligatorio de Programación 3

Entrega correspondiente a la carrera
Analista en Tecnologías de la Información / Analista Programador

Veronica Busiello – 212712
Gonzalo Otheguy – 206324
Grupo N3B

Docente: Adriana Cabella

ÍNDICE

- Diagramas
- Consideraciones
- Modelo

DECLARACIÓN DE AUTORÍA

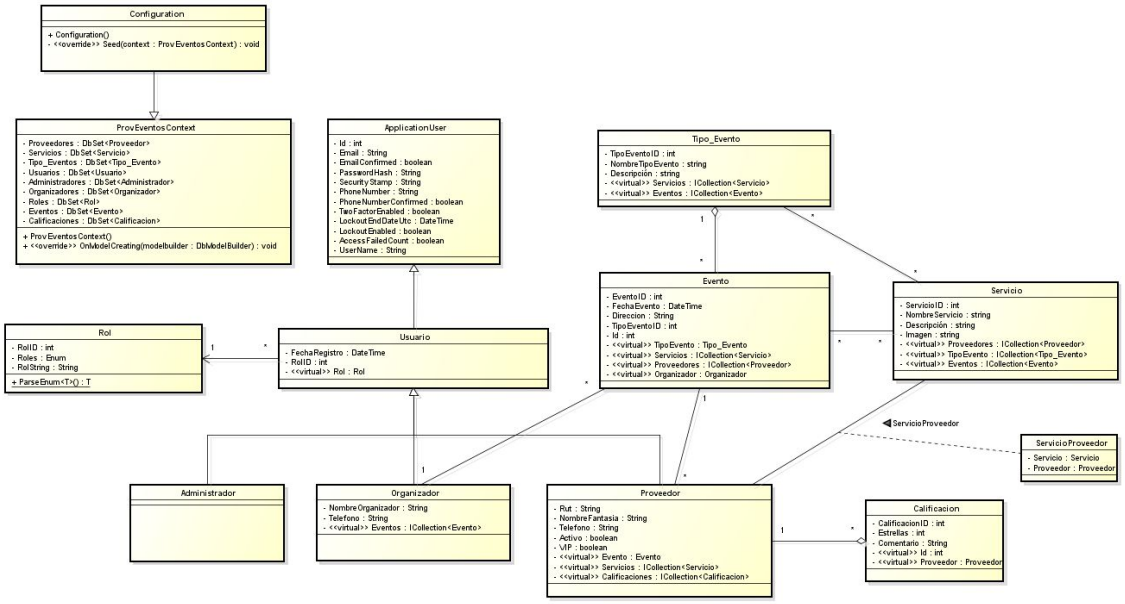
Nosotros, Veronica Busiello y Gonzalo Otheguy declaramos que el presente trabajo es de nuestra autoría. Puedo asegurar que:

- El trabajo fue producido en su totalidad mientras realizaba la primera entrega del obligatorio de Programación 3
- En aquellas secciones de este trabajo que se presentaron previamente para otra actividad o calificación de la universidad u otra institución, se han realizado las aclaraciones correspondientes.
- Cuando he consultado el trabajo publicado por otros, lo he atribuido con claridad.
- Cuando cité obras de otros, he indicado las fuentes. Con excepción de estas citas, la obra es enteramente mía.
- En el trabajo, he acusado recibo de las ayudas recibidas.
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega.

DIAGRAMAS

Diagrama de casos de uso

Diagrama de clases



CONSIDERACIONES

Debido a diversas cuestiones, los dos archivos de proveedores y servicios se levantan a la base de datos desde la carpeta BIN/APP_DATA, por lo que si se borra BIN, no estaría quedando la ruta que se necesita y no se van a poblar las tablas de proveedores y servicios con la información de los archivos.
Otra cuestión, en nuestra implementación la clase Usuario hereda de ApplicationUser, el cual tiene una PK string hasheada, por lo cual al hacer seed de Eventos, la FK que se inserta en cada nuevo Evento debe corregirse cada vez que se levanta la base, porque al estar hasheado ese campo, nunca va a ser el mismo.
Adicionalmente, utilizamos Migrations y se pobla la base para el resto de las tablas desde el file Configuration.cs en vez de utilizar un custom initializer.

MODELO

Usuario

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel;

namespace ProvEventos.Models
{
   [Table("Usuario")]
   public class Usuario : ApplicationUser
   {
      [Required]
      [DataType(DataType.Date)]
      [DisplayName("Fecha de registro")]
      [DisplayFormat(DataFormatString = "{0: dd/MM/yyyy}", ApplyFormatInEditMode = true)]
      [Column("FechaRegistro", Order = 2, TypeName = "date")]
      public DateTime FechaRegistro { get; set; }

      public int RolID { get; set; }
      public virtual Rol Rol { get; set; }
   }
```

```
}
```

Organizador

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel;

namespace ProvEventos.Models
{
    [Table("Organizador")]
    public class Organizador : Usuario
    {
        [Key]
        public override string Id { get; set; }

        [Required]
        [StringLength(100, ErrorMessage = "El nombre no puede ser mayor a 100 caracteres")]
        [DisplayName("Nombre")]
        [Column("NombreOrganizador", Order = 2, TypeName = "varchar")]
        public string NombreOrganizador { get; set; }

        [Required]
        [DataType(DataType.PhoneNumber)]
        [DisplayName("Teléfono")]
        [RegularExpression(@"^([0-9]{9})$", ErrorMessage = "Teléfono inválido")]
        [Column("Telefono", Order = 3, TypeName = "varchar")]
        public string Telefono { get; set; }

        public ICollection<Evento> Eventos { get; set; }
    }
}
```

Administrador

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

```
namespace ProvEventos.Models
{
    [Table("Administrador")]
    public class Administrador : Usuario
    {
    }
}
```

Proveedor

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel;

namespace ProvEventos.Models
{
    [Table("Proveedor")]
    public class Proveedor : Usuario
    {
        [Key]
        public override string Id { get; set; }

        [Required(ErrorMessage = "El RUT no puede estar vacío")]
        [Column("Rut", Order = 2, TypeName = "varchar")]
        public string Rut { get; set; }

        [Required]
        [StringLength(100, ErrorMessage = "El nombre de fantasía no puede ser mayor a 100
caracteres")]
        [Column("NombreFantasia", Order = 3, TypeName = "varchar")]
        public string NombreFantasia { get; set; }

        [Required]
        [DataType(DataType.PhoneNumber)]
        [DisplayName("Teléfono")]
        [RegularExpression(@"^([0-9]{9})$", ErrorMessage = "Teléfono inválido")]
        [Column("Telefono", Order = 4, TypeName = "varchar")]
        public string Telefono { get; set; }

        [Column("Activo", Order = 5, TypeName = "bit")]
```

```
        public bool Activo { get; set; }

        [Column("VIP", Order = 6, TypeName = "bit")]
        public bool VIP { get; set; }

        public virtual Evento Evento { get; set; }

        public virtual ICollection<Servicio> Servicios { get; set; }

        public virtual ICollection<Calificacion> Calificaciones { get; set; }
    }
}
```

Evento

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Web;

namespace ProvEventos.Models
{
    [Table("Evento")]
    public class Evento
    {
        [Required]
        [DatabaseGeneratedAttribute(DatabaseGeneratedOption.Identity), Key()]
        [Column("EventoID", Order = 1, TypeName = "int")]
        public int EventoID { get; set; }

        [Required]
        [DataType(DataType.Date)]
        [DisplayName("Fecha de evento")]
        [DisplayFormat(DataFormatString = "{0: dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        [Column("FechaEvento", Order = 3, TypeName = "date")]
        public DateTime FechaEvento { get; set; }

        [Required(ErrorMessage = "Ingrese una direccion")]
        [DisplayName("Dirección")]
```

```
        [StringLength(100, ErrorMessage = "La direccion debe tener menos de 100
caracteres")]
        [Column("Direccion", Order = 2, TypeName = "varchar")]
        public String Direccion { get; set; }

        public int TipoEventoID { get; set; }
        public virtual Tipo_Evento TipoEvento { get; set; }

        public virtual List<Servicio> Servicios { get; set; }

        public virtual List<Proveedor> Proveedores { get; set; }

        public string Id { get; set; }
        public virtual Organizador Organizador { get; set; }
    }
}
```

Servicio

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ProvEventos.Models
{
    [Table("Servicio")]
    public class Servicio
    {
        [Required]
        [DatabaseGeneratedAttribute(DatabaseGeneratedOption.Identity), Key()]
        [Column("ServicioID", Order = 1, TypeName = "int")]
        public int ServicioID { get; set; }

        [Required]
        [StringLength(50, ErrorMessage = "El nombre no puede ser mayor a 50 caracteres")]
        [Display(Name = "Nombre")]
        [Column("NombreServicio", Order = 2, TypeName = "varchar")]
        public string NombreServicio { get; set; }

        [StringLength(250, ErrorMessage = "La descripción no puede ser mayor a 250
caracteres")]
        [Display(Name = "Descripción")]
        [Column("Descripcion", Order = 3, TypeName = "varchar")]
```

```
        public string Descripcion { get; set; }

        [StringLength(200)]
        [Column("Imagen", Order = 4, TypeName = "varchar")]
        public string Imagen { get; set; }

        public virtual ICollection<Proveedor> Proveedores { get; set; }

        public virtual ICollection<Tipo_Evento> TipoEvento { get; set; }

        public virtual ICollection<Evento> Eventos { get; set; }
    }
}
```

Rol

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using Microsoft.AspNet.Identity.EntityFramework;

namespace ProvEventos.Models
{
    public enum Roles
    {
        Administrador,
        Organizador,
        Proveedor
    }

    [Table("Rol")]
    public class Rol
    {
        [Key]
        [Required]
        [Column("RolID", Order = 1, TypeName = "int")]
        public int RolID { get; set; }

        [Column("Rol", Order = 2)]
        public string RolString
```

```
        {
            get { return Roles.ToString(); }
            private set { Roles = value.ParseEnum<Roles>(); }
        }

        [NotMapped]
        public Roles Roles { get; set; }
    }
    public static class StringExtensions
    {
        public static T ParseEnum<T>(this string value)
        {
            return (T)Enum.Parse(typeof(T), value, true);
        }
    }
}
```

Tipo_Evento

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel;

namespace ProvEventos.Models
{
    [Table("Tipo_Evento")]
    public class Tipo_Evento
    {
        [Required]
        [DatabaseGeneratedAttribute(DatabaseGeneratedOption.Identity), Key()]
        [Column("TipoEventoID", Order = 1, TypeName = "int")]
        public int TipoEventoID { get; set; }

        [Required]
        [DisplayName("Evento")]
        [StringLength(100, ErrorMessage = "El nombre no puede ser mayor a 100 caracteres")]
        [Column("NombreTipoEvento", Order = 2, TypeName = "varchar")]
        public string NombreTipoEvento { get; set; }

        [StringLength(250, ErrorMessage = "El nombre no puede ser mayor a 250 caracteres")]
        [Column("Descripcion", Order = 3, TypeName = "varchar")]
        public string Descripcion { get; set; }
```

```
        public virtual ICollection<Servicio> Servicios { get; set; }

        public virtual ICollection<Evento> Eventos { get; set; }
    }
}
```

ServicioProveedor

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace ProvEventos.Models
{
    public class ServicioProveedor
    {
        public Servicio Servicio { get; set; }
        public Proveedor Proveedor { get; set; }
    }
}
```

Calificacion

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Web;

namespace ProvEventos.Models
{
    [Table("Calificacion")]
    public class Calificacion
    {
        [Required]
        [DatabaseGeneratedAttribute(DatabaseGeneratedOption.Identity), Key()]
        [Column("CalificacionID", Order = 1, TypeName = "int")]
        public int CalificacionID { get; set; }
```

```
        [Required(ErrorMessage = "Debe ingresar una calificación")]
        [DisplayName("Estrellas")]
        [Range(1, 5, ErrorMessage = "El puntaje debe ser entre 1 y 5")]
        [Column("Estrellas", Order = 2, TypeName = "int")]
        public int Estrellas { get; set; }

        [DisplayName("Comentario")]
        [StringLength(200, ErrorMessage = "El comentario debe tener menos de 200
caracteres")]
        [Column("Comentario", Order = 3, TypeName = "varchar")]
        public string Comentario { get; set; }

        public string Id { get; set; }
        public virtual Proveedor Proveedor { get; set; }
    }
}
```