UD6 Desarrollo de aplicaciones web con ASP.NET Core.

6.2 Aplicación de ejemplo.

Vamos a continuar en esta parte con el desarrollo de nuestra aplicación, pasando al elemento más importante, que son los propios materiales inventariables, y que son el motivo de la aplicación. Como veremos, va a ser también la parte que más dificultades va a plantear.

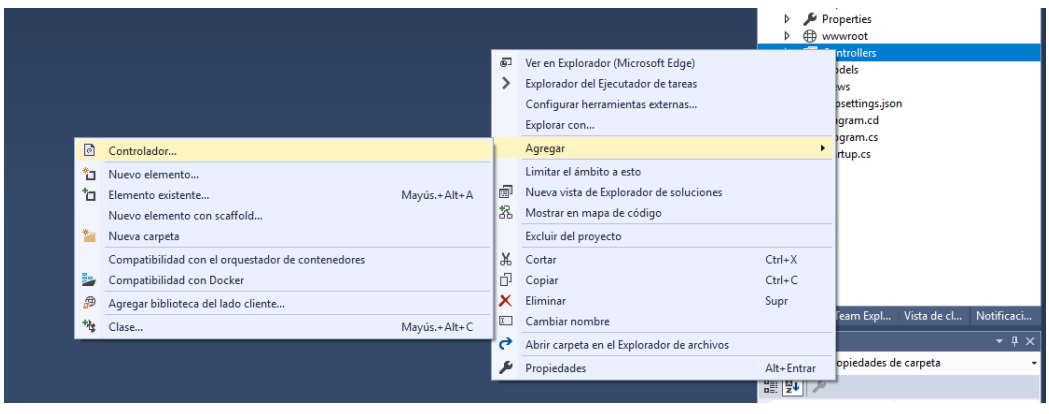
Controlador y enlace al inventario.

Comenzaremos del mismo modo que hemos hecho en la gestión de departamentos y ubicaciones, añadiendo un nuevo enlace en la plantilla de la aplicación y creando el controlador y las vistas con el asistente, ya que como se ha demostrado en la parte anterior, nos ahorra mucho trabajo en tareas repetitivas.

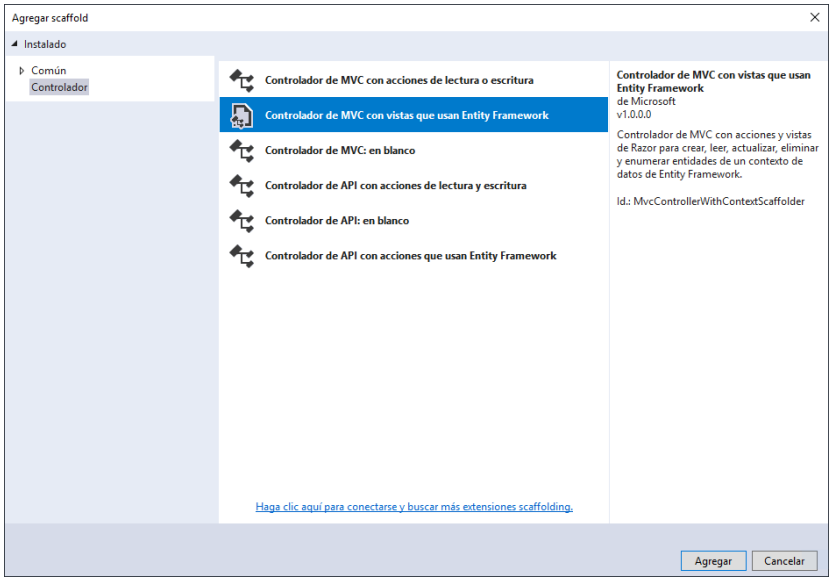
1. Añadimos un nuevo enlace en el fichero *_Layout.cshtml*, en este caso antes de los anteriores:

```
<div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row-reverse">
    <ul class="navbar-nav flex-grow-1">
        <li class="nav-item">
            <a class="nav-link text-dark" asp-area=""
                asp-controller="Inventario" asp-action="Index">
                <i class="fas fa-clipboard-list"></i> Inventario</a>
        </li>
        <li class="nav-item">
            <a class="nav-link text-dark" asp-area=""
                asp-controller="Departamentos" asp-action="Index">
                <i class="fas fa-graduation-cap"></i> Departamentos</a>
        </li>
        <li class="nav-item">
            <a class="nav-link text-dark" asp-area=""
                asp-controller="Ubicaciones" asp-action="Index">
                <i class="fas fa-building"></i> Ubicaciones</a>
        </li>
    </ul>
</div>
```

2. Creamos un controlador para la gestión del inventario haciendo click derecho en nuestra carpeta Controllers, y seleccionando la opción **Agregar | Controlador...**



3. En el asistente, seleccionamos el controlador de MVC con vistas que usan Entity Framework.



4. Como clase de modelo seleccionaremos Material y en el segundo desplegable CIFPACarballeiraContext. Llamaremos al controlador *InventarioController*. Finalmente pulsamos en **Agregar**.

Alta de material.

Este caso nada tiene que ver con lo que hemos hecho hasta el momento. Tanto las entidades Departamento como Ubicacion eran muy sencillas, y por lo tanto también el código generado, que hemos podido aprovechar por completo. En este caso tendremos mucho trabajo que hacer para que todo quede finalmente como podría ser de esperar. Los materiales de inventario se relacionan con otras entidades, de modo que es necesario especificar donde se encuentran o a quien pertenecen. Además de contener propiedades de diversos tipos de datos, que deberán ser tratados de forma diferente.

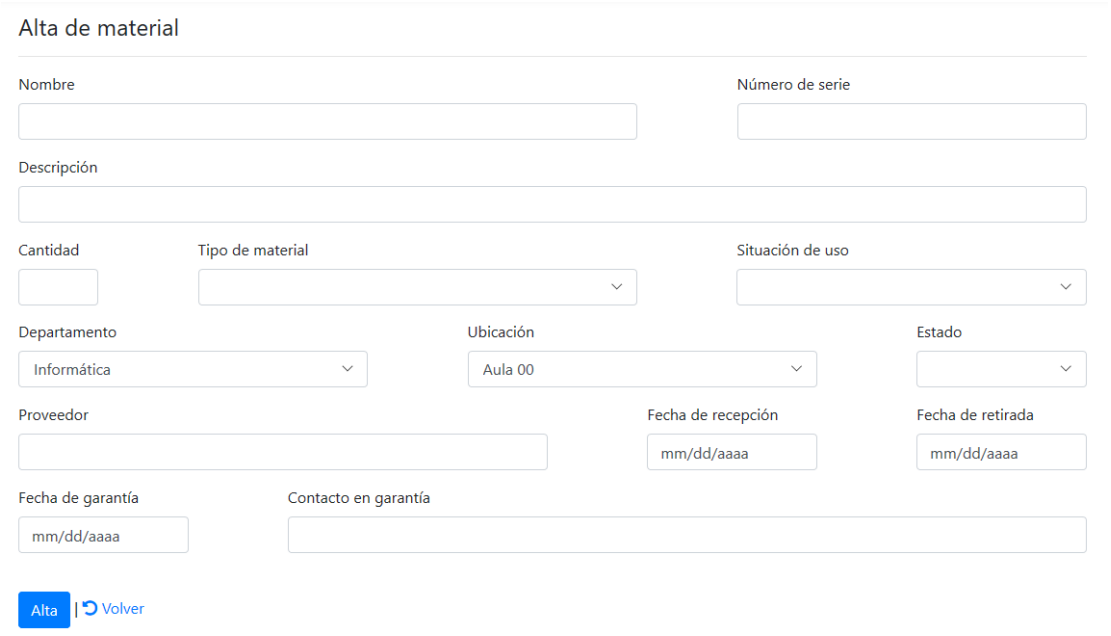
1. En primer lugar vamos a empezar por modificar un poco la presentación, editando la vista *Create.cshtml*, para que el formulario sea más cómodo de utilizar y de visualizar, ya que la plantilla por defecto se limita a mostrar un control en cada fila.

```
@model CIFPACarballeiraWeb.Models.Material

@{
    ViewData["Title"] = "Alta de material";
}

<h4>Alta de material</h4>
<hr />
<div class="row">
    <div class="col-md-12">
        <form asp-action="Create">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-row">
                <div class="form-group col-md-7">
                    <label asp-for="Nombre" class="control-label"></label>
                    <input asp-for="Nombre" class="form-control" />
                    <span asp-validation-for="Nombre" class="text-danger"></span>
```

```
        </div>
        <div class="form-group col-md-4 offset-1">
            <label asp-for="NumSerie" class="control-label"></label>
            <input asp-for="NumSerie" class="form-control" />
            <span asp-validation-for="NumSerie" class="text-danger"></span>
        </div>
    </div>
    <div class="form-group">
        <label asp-for="Descripcion" class="control-label"></label>
        <input asp-for="Descripcion" class="form-control" />
        <span asp-validation-for="Descripcion" class="text-danger"></span>
    </div>
    <div class="form-row">
        <div class="form-group col-md-1">
            <label asp-for="Cantidad" class="control-label"></label>
            <input asp-for="Cantidad" class="form-control" />
        </div>
        <div class="form-group col-md-5 offset-1">
            <label asp-for="TipoMaterial" class="control-label"></label>
            <select asp-for="TipoMaterial" class="form-control"></select>
        </div>
        <div class="form-group col-md-4 offset-1">
            <label asp-for="TipoUso" class="control-label"></label>
            <select asp-for="TipoUso" class="form-control"></select>
        </div>
    </div>
    <div class="form-group" style="margin-top: -17px;">
        <span asp-validation-for="Cantidad" class="text-danger"></span>
    </div>
    <div class="form-row">
        <div class="form-group col-md-4">
            <label asp-for="Departamento" class="control-label"></label>
            <select asp-for="DepartamentoId" class="form-control"
                asp-items="ViewBag.DepartamentoId"></select>
        </div>
        <div class="form-group col-md-4 offset-1">
            <label asp-for="Ubicacion" class="control-label"></label>
            <select asp-for="UbicacionId" class="form-control"
                asp-items="ViewBag.UbicacionId"></select>
        </div>
        <div class="form-group col-md-2 offset-1">
            <label asp-for="Estado" class="control-label"></label>
            <select asp-for="Estado" class="form-control"></select>
        </div>
    </div>
    <div class="form-row">
        <div class="form-group col-md-6">
            <label asp-for="Proveedor" class="control-label"></label>
            <input asp-for="Proveedor" class="form-control" />
            <span asp-validation-for="Proveedor" class="text-danger"></span>
        </div>
        <div class="form-group col-md-2 offset-1">
            <label asp-for="FechaRecepcion" class="control-label"></label>
            <input asp-for="FechaRecepcion" class="form-control" />
            <span asp-validation-for="FechaRecepcion"
                class="text-danger"></span>
        </div>
        <div class="form-group col-md-2 offset-1">
            <label asp-for="FechaRetirada" class="control-label"></label>
            <input asp-for="FechaRetirada" class="form-control" />
            <span asp-validation-for="FechaRetirada"
                class="text-danger"></span>
        </div>
    </div>
```

```
                <div class="form-row">
                    <div class="form-group col-md-2">
                        <label asp-for="FechaGarantia" class="control-label"></label>
                        <input asp-for="FechaGarantia" class="form-control" />
                        <span asp-validation-for="FechaGarantia"
                            class="text-danger"></span>
                    </div>
                    <div class="form-group col-md-9 offset-1">
                        <label asp-for="InfoGarantia" class="control-label"></label>
                        <input asp-for="InfoGarantia" class="form-control" />
                        <span asp-validation-for="InfoGarantia"
                            class="text-danger"></span>
                    </div>
                </div>

                <br />
                <div class="form-group">
                    <input type="submit" value="Alta" class="btn btn-primary" /> |
                    <a asp-action="Index"> <i class="fas fa-undo-alt"></i> Volver</a>
                </div>
            </form>
        </div>
    </div>

    @section Scripts {
        @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
    }
```

Nuestro formulario tendrá ahora un aspecto similar al siguiente:



Lo que hemos hecho es únicamente reordenar los elementos para conseguir una distribución más compacta de los controles del formulario, pero hay una gran cantidad de cosas que el asistente ha hecho por nosotros. Vamos a analizar las cuestiones más interesantes.

Lo primero que podemos observar son los títulos o nombres de los controles del formulario. Podemos ver que los textos son los que hemos especificado con el atributo *Display* de nuestro modelo. El taghelper *asp-for* de los elementos label hace que ASP.NET Core busque en la propiedad de la entidad.

Si encuentra un atributo *Display* muestra el texto indicado, y en caso contrario utiliza el nombre de la propiedad. De esta forma podemos definir los textos que deseamos que se muestren en nuestros formularios, o en tablas, como veremos más adelante, en un único lugar y de forma sencilla, en lugar de en cada una de las vistas donde estos se utilicen.
Otra cosa que ASP.NET Core hace por nosotros es utilizar los atributos *DataType* para generar código html que muestre los controles adecuados para cada tipo de dato. Por ejemplo vemos que el control de *Cantidad* es de tipo numérico, y los de las fechas son controles de fecha, que no tienen en cuenta la hora, tal y como indicamos en el atributo correspondiente.
Por último, y esto es aún más interesante, podemos observar como el framework ha detectado que la ubicación y el departamento son propiedades de navegación, y por lo tanto, en lugar de crear un simple campo para introducir un Id, que es muy incómodo para el usuario, genera todo el código necesario para que muestre un desplegable con los departamentos y ubicaciones, permitiendo seleccionar estos por su nombre, mientras que la información que almacenará será en realidad su identificador.
El único problema que observamos a simple vista es que no es capaz de hacer lo propio con los tipos de enumeración. En este caso, tal y como adelantamos en el documento anterior, las enumeraciones nos obligarán a realizar una serie de tareas adicionales para que todo funcione correctamente, aunque como vamos a ver, tampoco será demasiado complicado.

2. Antes de solucionar el problema de las enumeraciones, observaremos por un momento cómo funcionan las listas de selección con la entidades relacionadas, es decir, departamentos y ubicaciones. El concepto clave es que un control de tipo lista debe obtener de algún modo tanto los elementos que debe mostrar como los valores que pasará al aceptar el formulario. Esto nos permite mostrar una información amigable al usuario, como pueden ser los nombres de departamento, mientras que el valor que utilizará será el identificador del mismo.
   Si echamos un ojo a los métodos Create del controlador podemos ver que cuando mostramos la vista que contiene el formulario, lo primero que hacemos es cargar esta información, por ejemplo, para los departamentos tenemos lo siguiente:

   ```
   ViewData["DepartamentoId"] = new SelectList(_context.Departamentos, "ID", "Nombre");
   ```

   Un objeto SelectList permite almacenar la información que requiere un control select de html. En nuesto caso podemos ver que este objeto se almacena en el ViewData / ViewBag para poderlo pasar a la vista. Para la creación del mismo especificamos el origen desde el que obtendrá los datos, siendo aquí la lista de departamentos que obtenemos de la BD gracias a EF, y especificamos también los nombres de los campos que se utilizarán para almacenar los datos que se utilizarán internamente, así como lo que se mostrará. Podemos ver lo sencillo que es especificar que queremos utilizar los Id de todos los departamentos en la BD, mientras que mostraremos los nombres de los mismos.
   Por último, desde la vista, utilizamos el taghelper *asp-items* para indicar al control select los elementos que va a utilizar:

   ```
   <select asp-for="DepartamentoId" class="form-control"
       asp-items="ViewBag.DepartamentoId"></select>
   ```

3. Para los controles necesitamos algo parecido, ya que lo que queremos es indicar los controles select los valores que utilizará para almacenar / mostrar información.
   En este caso vamos a realizar esta tarea directamente de la vista, en lugar de en el controlador. Al no necesitar obtener los datos de la BD ya que están definidos directamente en una enumeración se trata de código que sólo tiene que ver con la presentación y es perfectamente correcto hacerlo de este modo. De hecho, ASP.NET Core nos ofrece un Html Helper para realizar esta tarea. Este

helper permite obtener los datos para un select desde un tipo de enumeración. Los datos que se almacenarán son los valores de la enumeración, mientras que, para el texto que se muestra al usuario, buscará los atributos *Display* de cada valor de la enumeración.
A continuación se muestra el código para el tipo de material:

```
<select asp-for="TipoMaterial" class="form-control"
     asp-items="Html.GetEnumSelectList<TiposMaterial>()"></select>
```

Especificamos como objetos para el select el resultado del helper, que obtiene un objeto *SelectList* a partir de un tipo de enumeración. Como podemos ver resulta bastante sencillo. Hacemos a continuación lo mismo para el resto de tipos enumerados de nuestro formulario.

```
<select asp-for="TipoUso" class="form-contro
    asp-items="Html.GetEnumSelectList<TiposUso>()"></select>


<select asp-for="Estado" class="form-control"
     asp-items="Html.GetEnumSelectList<Estados>()"></select>
```

Si observamos los datos almacenados en la BD podremos ver que los valores que se almacenan en los campos de tipos enumerados son en realidad números enteros, que representan el orden del elemento de la enumeración, y no el valor del texto que se muestra. De este modo, estamos almacenando el número de elemento de una lista de posibles valores. ASP.NET Core se encarga de hacer toda la transformación por nosotros, liberándonos de una gran cantidad de tareas tediosas.

4. Con esto ya tenemos lista la posibilidad de dar de alta nuevos elementos del inventario.

Actualizar inventario.

Para la opción de edición, al igual que para la creación, todo nuestro código está listo para utilizar. Lo único que necesitamos es reubicar los elementos del formulario y cargar los datos necesarios para los select correspondientes a tipos enumerados.

```
@model CIFPACarballeiraWeb.Models.Material

@{
    ViewData["Title"] = "Actualizar inventario";
}

<h4>Actualizar inventario</h4>
<hr />
<div class="row">
    <div class="col-md-12">
        <form asp-action="Edit">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <input type="hidden" asp-for="ID" />
            <div class="form-row">
                <div class="form-group col-md-7">
                    <label asp-for="Nombre" class="control-label"></label>
                    <input asp-for="Nombre" class="form-control" />
                    <span asp-validation-for="Nombre" class="text-danger"></span>
                </div>
                <div class="form-group col-md-4 offset-1">
                    <label asp-for="NumSerie" class="control-label"></label>
                    <input asp-for="NumSerie" class="form-control" />
                    <span asp-validation-for="NumSerie" class="text-danger"></span>
```

```
        </div>
    </div>
    <div class="form-group">
        <label asp-for="Descripcion" class="control-label"></label>
        <input asp-for="Descripcion" class="form-control" />
        <span asp-validation-for="Descripcion" class="text-danger"></span>
    </div>
    <div class="form-row">
        <div class="form-group col-md-1">
            <label asp-for="Cantidad" class="control-label"></label>
            <input asp-for="Cantidad" class="form-control" />
        </div>
        <div class="form-group col-md-5 offset-1">
            <label asp-for="TipoMaterial" class="control-label"></label>
            <select asp-for="TipoMaterial" class="form-control"
               asp-items="Html.GetEnumSelectList<TiposMaterial>()"></select>
        </div>
        <div class="form-group col-md-4 offset-1">
            <label asp-for="TipoUso" class="control-label"></label>
            <select asp-for="TipoUso" class="form-control"
                asp-items="Html.GetEnumSelectList<TiposUso>()"></select>
        </div>
    </div>
    <div class="form-group" style="margin-top: -17px;">
        <span asp-validation-for="Cantidad" class="text-danger"></span>
    </div>
    <div class="form-row">
        <div class="form-group col-md-4">
            <label asp-for="Departamento" class="control-label"></label>
            <select asp-for="DepartamentoId" class="form-control"
                asp-items="ViewBag.DepartamentoId"></select>
        </div>
        <div class="form-group col-md-4 offset-1">
            <label asp-for="Ubicacion" class="control-label"></label>
            <select asp-for="UbicacionId" class="form-control"
                asp-items="ViewBag.UbicacionId"></select>
        </div>
        <div class="form-group col-md-2 offset-1">
            <label asp-for="Estado" class="control-label"></label>
            <select asp-for="Estado" class="form-control"
                  asp-items="Html.GetEnumSelectList<Estados>()"></select>
        </div>
    </div>
    <div class="form-row">
        <div class="form-group col-md-6">
            <label asp-for="Proveedor" class="control-label"></label>
            <input asp-for="Proveedor" class="form-control" />
            <span asp-validation-for="Proveedor" class="text-danger"></span>
        </div>
        <div class="form-group col-md-2 offset-1">
            <label asp-for="FechaRecepcion" class="control-label"></label>
            <input asp-for="FechaRecepcion" class="form-control" />
            <span asp-validation-for="FechaRecepcion"
                class="text-danger"></span>
        </div>
        <div class="form-group col-md-2 offset-1">
            <label asp-for="FechaRetirada" class="control-label"></label>
            <input asp-for="FechaRetirada" class="form-control" />
            <span asp-validation-for="FechaRetirada"
                class="text-danger"></span>
        </div>
    </div>
    <div class="form-row">
        <div class="form-group col-md-2">
```

```
                    <label asp-for="FechaGarantia" class="control-label"></label>
                    <input asp-for="FechaGarantia" class="form-control" />
                    <span asp-validation-for="FechaGarantia"
                        class="text-danger"></span>
                </div>
                <div class="form-group col-md-9 offset-1">
                    <label asp-for="InfoGarantia" class="control-label"></label>
                    <input asp-for="InfoGarantia" class="form-control" />
                    <span asp-validation-for="InfoGarantia"
                        class="text-danger"></span>
                </div>
            </div>

            <br />
            <div class="form-group">
                <input type="submit" value="Actualizar" class="btn btn-primary" /> |
                <a asp-action="Index"> <i class="fas fa-undo-alt"></i> Volver</a>
            </div>
        </form>
    </div>
</div>

@section Scripts {
    @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

Vista de detalle.

Es habitual en una aplicación, que en la página en la que se muestra la lista de objetos de un tipo determinado, no se muestren todos los datos de cada objeto, si no que se ofrecen solamente los más importantes. Es importante por lo tanto tener un modo de obtener una vista completa que muestra toda la información de un objeto. Este es el objetivo de la vista de detalle.
Esta parte es una de las más sencillas de la aplicación, y el código generado por el asistente ya muestra toda la información de cada elemento del inventario, por lo que únicamente vamos a necesitar, como casi siempre, modificaciones en la presentación.

Antes de pasar a modificar la vista, sin embargo, nos pararemos un poco a analizar el código del método de acción *Details*, para explicar la forma en la que obtiene la información.

Cuando almacenamos un objeto inventariable, ya explicamos que en la tabla correspondiente se guarda el Id tanto del departamento al que pertenece como de la ubicación en la que se encuentra. De este modo en la BD se establece una relación entre estos objetos. En una consulta SQL tradicional, para obtener el nombre del departamento y de la ubicación de un objeto, tendríamos que hacer una consulta con un join. Gracias a las propiedades de navegación y EF veremos, sin embargo, que obtener estos datos es sumamente sencillo. Vamos a observar el código en el que se obtiene la información de un objeto para mostrar su detalle:

```
var material = await _context.Materiales
                .Include(m => m.Departamento)
                .Include(m => m.Ubicacion)
                .FirstOrDefaultAsync(m => m.ID == id);
```

En esta consulta obtenemos, mediante nuestro contexto de datos, el primer elemento de la tabla Materiales, cuyo valor del campo ID coincida con el id que pasamos como parámetro. De esta forma obtenemos el material inventariable que deseamos.

La parte más interesante de la consulta son los métodos *Include*, que permiten especificar que se incluya automáticamente determinada información al hacer la consulta. De esta forma estamos indicando que cuando obtengamos el objeto, a través del id del departamento obtengamos toda la información del departamento al que pertenece y lo mismo con su ubicación. Para acceder más adelante a cualquier dato de estos elementos lo podremos hacer a través de la propiedad de navegación correspondiente.
Por ejemplo, en la vista de detalle queremos mostrar el nombre del departamento al que pertenece el material, no su id, y esto se consigue de este modo:

```
@Html.DisplayFor(model => model.Departamento.Nombre)
```

Al haber obtenido toda la información del departamento en la consulta, podemos acceder a todas las propiedades del objeto asociado utilizando el operador '.' correspondiente.

Ahora que ya conocemos el mecanismo para obtener la información de los objetos asociados, realizamos las modificaciones en la vista detalle para ordenar la presentación de la información.

```
@model CIFPACarballeiraWeb.Models.Material

@{
    ViewData["Title"] = "Detalle de material";
}

<div>
    <h4>Detalle de material</h4>
    <hr />
    <dl class="row">
        <dt class="col-sm-1">
            @Html.DisplayNameFor(model => model.Nombre)
        </dt>
        <dd class="col-sm-7">
            @Html.DisplayFor(model => model.Nombre)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.NumSerie)
        </dt>
        <dd class="col-sm-2">
            @Html.DisplayFor(model => model.NumSerie)
        </dd>

        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Descripcion)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Descripcion)
        </dd>

        <dt class="col-sm-1">
            @Html.DisplayNameFor(model => model.Cantidad)
        </dt>
        <dd class="col-sm-1">
            @Html.DisplayFor(model => model.Cantidad)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.TipoMaterial)
        </dt>
        <dd class="col-sm-4">
            @Html.DisplayFor(model => model.TipoMaterial)
        </dd>
        <dt class="col-sm-2">
```

```
            @Html.DisplayNameFor(model => model.TipoUso)
        </dt>
        <dd class="col-sm-2">
            @Html.DisplayFor(model => model.TipoUso)
        </dd>

        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Departamento)
        </dt>
        <dd class="col-sm-2">
            @Html.DisplayFor(model => model.Departamento.Nombre)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Ubicacion)
        </dt>
        <dd class="col-sm-3">
            @Html.DisplayFor(model => model.Ubicacion.Nombre)
        </dd>
        <dt class="col-sm-1">
            @Html.DisplayNameFor(model => model.Estado)
        </dt>
        <dd class="col-sm-2">
            @Html.DisplayFor(model => model.Estado)
        </dd>

        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Proveedor)
        </dt>
        <dd class="col-sm-4">
            @Html.DisplayFor(model => model.Proveedor)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.FechaRecepcion)
        </dt>
        <dd class="col-sm-1">
            @Html.DisplayFor(model => model.FechaRecepcion)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.FechaRetirada)
        </dt>
        <dd class="col-sm-1">
            @Html.DisplayFor(model => model.FechaRetirada)
        </dd>

        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.FechaGarantia)
        </dt>
        <dd class="col-sm-1">
            @Html.DisplayFor(model => model.FechaGarantia)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.InfoGarantia)
        </dt>
        <dd class="col-sm-7">
            @Html.DisplayFor(model => model.InfoGarantia)
        </dd>
    </dl>
</div>

<br/>
<div>
    <a asp-action="Edit" asp-route-id="@Model.ID">
        <i class="fas fa-edit"></i> Actualizar</a> |
    <a asp-action="Index"> <i class="fas fa-undo-alt">
```

```
        </i> Volver</a>
</div>
```

Dar de baja un material del inventario.

Como era de esperar, la eliminación de objetos del inventario no necesitará más que unos pequeños ajustes en la vista. Por defecto nos muestra la información completa del material en la pantalla de confirmación de eliminación, pero eso más bien excesivo, así que dejaremos únicamente los elementos más importantes.

```
@model CIFPACarballeiraWeb.Models.Material

@{
    ViewData["Title"] = "Baja del inventario";
}

<h3>Baja de inventario</h3>

<h4>Estás seguro de eliminar el material del inventario?</h4>
<div>
    <hr />
    <dl class="row">
        <dt class="col-sm-1">
            @Html.DisplayNameFor(model => model.Nombre)
        </dt>
        <dd class="col-sm-7">
            @Html.DisplayFor(model => model.Nombre)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.NumSerie)
        </dt>
        <dd class="col-sm-2">
            @Html.DisplayFor(model => model.NumSerie)
        </dd>

        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Descripcion)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Descripcion)
        </dd>

        <dt class="col-sm-1">
            @Html.DisplayNameFor(model => model.Cantidad)
        </dt>
        <dd class="col-sm-1">
            @Html.DisplayFor(model => model.Cantidad)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.TipoMaterial)
        </dt>
        <dd class="col-sm-4">
            @Html.DisplayFor(model => model.TipoMaterial)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.TipoUso)
        </dt>
        <dd class="col-sm-2">
            @Html.DisplayFor(model => model.TipoUso)
        </dd>
```

```
            <dt class="col-sm-1">
                @Html.DisplayNameFor(model => model.Estado)
            </dt>
            <dd class="col-sm-1">
                @Html.DisplayFor(model => model.Estado)
            </dd>
            <dt class="col-sm-2">
                @Html.DisplayNameFor(model => model.Departamento)
            </dt>
            <dd class="col-sm-3">
                @Html.DisplayFor(model => model.Departamento.Nombre)
            </dd>
            <dt class="col-sm-2">
                @Html.DisplayNameFor(model => model.Ubicacion)
            </dt>
            <dd class="col-sm-3">
                @Html.DisplayFor(model => model.Ubicacion.Nombre)
            </dd>
        </dl>

        <br/>
        <form asp-action="Delete">
            <input type="hidden" asp-for="ID" />
            <input type="submit" value="Baja del inventario" class="btn btn-danger" /> |
            <a asp-action="Index"><i class="fas fa-undo-alt"></i> Volver</a>
        </form>
    </div>
```

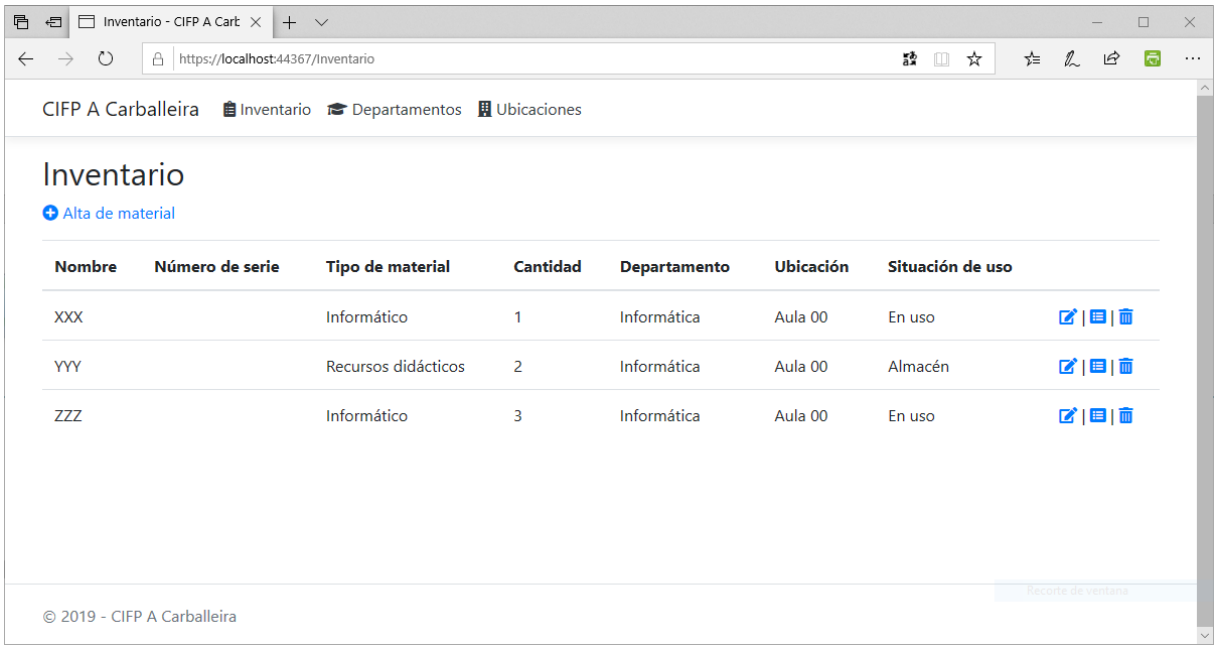
Página principal del inventario.

Sólo nos queda por arreglar la vista principal, donde se muestra la lista de objetos inventariados. Por defecto se muestran todas las propiedades del objeto en la tabla, pero eso no es operativo, al ser demasiada información para ser mostrada en la tabla. Si queremos ver toda la información de un objeto utilizaremos la vista de detalle, pero para la tabla dejaremos sólo la información más significativa:

```
@model IEnumerable<CIFPACarballeiraWeb.Models.Material>

@{
    ViewData["Title"] = "Inventario";
}

<h2>Inventario</h2>

<p>
    <a asp-action="Create"><i class="fa fa-plus-circle"></i> Alta de material</a>
</p>
<table class="table">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.Nombre)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.NumSerie)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.TipoMaterial)
            </th>
```

```
                <th>
                    @Html.DisplayNameFor(model => model.Cantidad)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Departamento)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Ubicacion)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.TipoUso)
                </th>
                <th></th>
            </tr>
        </thead>
        <tbody>
@foreach (var item in Model) {
            <tr>
                <td>
                    @Html.DisplayFor(modelItem => item.Nombre)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.NumSerie)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.TipoMaterial)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Cantidad)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Departamento.Nombre)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Ubicacion.Nombre)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.TipoUso)
                </td>
                <td>
                    <a asp-action="Edit" asp-route-id="@item.ID">
                        <i class="fas fa-edit" title="Editar"></i></a> |
                    <a asp-action="Details" asp-route-id="@item.ID">
                        <i class="fas fa-list-alt" title="Detalle"></i></a> |
                    <a asp-action="Delete" asp-route-id="@item.ID">
                        <i class="fas fa-trash-alt" title="Eliminar"></i></a>
                </td>
            </tr>
}
        </tbody>
    </table>
```

Si ejecutamos ahora nuestra aplicación y vamos a la página inicial del inventario podemos observar como ahora la información es mucho más clara.



Con esto terminamos la segunda parte de la unidad, y nuestra aplicación es completamente funcional. En los apartados siguientes veremos como añadir nuevas funcionalidades como búsqueda y ordenamiento en la página principal de inventario. Implementaremos también un mecanismo de autenticación, de modo que sólo un usuario registrado y logueado pueda utilizar la aplicación.