Aplicación para compartir fotos de Adventure Works (propuesta)

Documento de planificación detallado

*18/11/2020*

Autores: *Juan Ramón Díaz Fernández*

# Introducción

El autor ha examinado el documento de investigación inicial. Según los casos de uso, los requisitos técnicos y otro contenido de ese documento, el autor ha creado los planes detallados a continuación. La junta ya acordó que la aplicación para compartir fotos se construirá como un sitio web basado en la tecnología ASP.NET MVC de Microsoft. Por lo tanto, los detalles presentados aquí incluyen los nombres y propiedades de las clases de modelo y los controladores que los desarrolladores deben crear. También se han identificado vistas y se han incluido diagramas de prototipo (wireframes [MockFlow](https://mockflow.com/)) para ayudar a visualizar la interfaz de usuario para partes importantes del sitio.

Es probable que el diseño de la aplicación evolucione a lo largo del proceso de desarrollo a medida que cambian los requisitos. El equipo de desarrollo adoptará prácticas ágiles para garantizar que dichos cambios se reflejen en el producto final. Por lo tanto, este documento no debe considerarse una definición completa de la aplicación final.

# Modelo MVC

Los desarrolladores crearán un modelo con las siguientes clases de modelo. Para cada clase de modelo, se han enumerado las propiedades y se han proporcionado descripciones.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Modelo de Clases | Descripción | Propiedades | Tipos de Datos |
| 1. | Photo | La clase de modelo de foto representa una foto que los usuarios autenticados pueden cargar en el sitio web. | PhotoID | Integer |
| 2. | Title | String |
| 3. | PhotoFile | Byte[] |
| 4. | Description | String |
| 5. | CreatedDate | Datetime |
| 6. | Owner | Integer |
| 7. | Comment | La clase de modelo de comentario representa un comentario que los usuarios autenticados pueden agregar a las fotos. Esto permite a los usuarios discutir las fotos de otros. Cada comentario está asociado con una sola foto. | CommentID | Integer |
| 8. | User | String |
| 9. | Subject | String |
| 10. | Body | String |
| 11. | PhotoID | Integer |

# Cotroladores MVC

Los desarrolladores crearán los siguientes controladores. Para cada controlador, se han enumerado las acciones y se han proporcionado descripciones.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Controlador | Acción | Descripción |
| 1. | **PhotoController** | **DisplayGallery (GET)** | La acción se ejecuta cuando el usuario solicita la página Galería de fotos. La acción obtiene todas las fotos de la base de datos y las pasa a la vista DisplayGallery. |
| 2. |  | **DisplayRecent (GET)** | Similar a la anterior, obtiene las fotos más recientes en una lista más pequeña y las pasa a la vista DisplayGallery. |
| 3. |  | **DisplayPhoto (GET)** | "Detalles" de una foto en una galería. La acción obtiene todos los detalles de una sola foto de la base de datos y la pasa a la vista DisplayPhoto. |
| 4. |  | **AddPhoto (GET)** | clic en el enlace "Agregar una foto". La acción crea una nueva instancia de la clase de modelo Photo y la pasa a la vista AddPhoto. |
| 5. |  | **AddPhoto (POST)** | Esta acción se ejecuta cuando el usuario hace clic en "Guardar" en la vista Agregar foto. La acción guarda el archivo y los detalles de la nueva foto en la base de datos y redirige al usuario a la vista DisplayGallery. |
| 6. |  | **DeletePhoto (GET)** | Esta acción se ejecuta cuando el usuario hace clic en el enlace "Eliminar esta foto" en la vista Mostrar foto. La acción muestra la vista Eliminar foto, que solicita confirmación para la eliminación. |
| 7. |  | **DeletePhoto (POST)** | Esta acción se ejecuta cuando el usuario hace clic en "Eliminar" en la vista Eliminar foto. La acción elimina la foto actual, con sus comentarios asociados de la base de datos y redirige al usuario a la vista DisplayGallery. |
| 8. | **CommentController** | **DisplayComments (GET)** | Esta acción se ejecuta cuando se muestra la vista DisplayPhoto. La acción requiere el PhotoID actual como parámetro y lo usa para obtener todos los comentarios de la foto actual de la base de datos. La acción devuelve la vista parcial \_DisplayComments. |
| 9. |  | **AddComment (GET)** | Esta acción se ejecuta cuando el usuario hace clic en el enlace "Agregar un comentario" en la vista Mostrar foto. La acción crea una nueva instancia de la clase de modelo Comment y establece su PhotoID como el ID de la foto actual. Pasa este nuevo comentario a la vista AddComment. |
| 10. |  | **AddComment (POST)** | Esta acción se ejecuta cuando el usuario hace clic en "Enviar" en la vista Agregar comentario. La acción guarda los detalles del nuevo comentario en la base de datos y redirige al usuario a la vista DisplayPhoto. |

# Vistas MVC

|  |  |  |  |
| --- | --- | --- | --- |
|  | Controlador | Vista | Descripción |
| 1. | **PhotoController** | **DisplayGallery** | Esta vista muestra una colección de fotos en tamaño miniatura. Para cada foto, se muestran los valores de Título, Propietario y Fecha de creación |
| 2. |  | **DisplayPhoto** | Esta vista muestra una sola foto en tamaño completo. Los valores de Título y Propietario aparecen encima de la foto. El nombre de la foto, la descripción y otros valores aparecen debajo de la foto. Debajo de estos detalles, todos los comentarios de la foto actual se enumeran con un enlace "Agregar un comentario". |
| 3. |  | **AddPhoto** | Esta vista muestra un formulario que el usuario puede usar para cargar y describir una nueva foto. |
| 4. |  | **DeletePhoto** | Esta vista muestra un formulario que el usuario puede utilizar para confirmar la eliminación de una foto. La vista muestra detalles de la foto actual, como su título y descripción. |
| 5. | **CommentController** | **DisplayComments** | This partial view, which is used on the DisplayPhoto form, displays all the comments associated with the current photo |
| 6. |  | **AddComment** | This view displays a form that the user can use to create a new comment for a photo |

# Recomendaciones de alojamiento

Dado que la aplicación para compartir fotos se desarrollará en ASP.NET Core MVC, debe estar alojada en un servidor web de Microsoft. El autor recomienda la siguiente configuración de alojamiento:

## Servidor web

## *El autor recomienda utilizar Microsoft Azure para alojar la aplicación Photo Sharing. Microsoft Azure puede alojar cualquier sitio web ASP.NET, incluida la aplicación ASP.NET Core MVC propuesta en este documento. El escalado es muy simple porque Microsoft, no Adventure Works, es responsable de agregar recursos del servidor en momentos de alto tráfico. Los costes son mínimos: dependen de la cantidad de datos servidos a los visitantes pero no es necesario mantener nuestro propio hardware.*

## Base de datos

*El autor recomienda utilizar SQL Database, dentro de Microsoft Azure, para alojar la base de datos subyacente de la aplicación Photo Sharing. En cuanto al servidor web, esta recomendación asegura un alojamiento de alta disponibilidad para la base de datos con una buena relación calidad-precio. Esto tiene especial sentido si el sitio web está alojado en Microsoft Azure.*