Aplicación para compartir fotos de Adventure Works (propuesta)

Documento de planificación detallado

*Insertar fecha aquí*

Autores: *Inserte su nombre aquí*

# 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 | El Modelo de la clase photo representa una foto que los | PhotoID | Integer |
| 2. |  | Usuarios autenticados pueden | Title | String |
| 3. |  | Pueden subir al sitio web. | PhotoFile | Binary |
| 4. |  |  | Description | String |
| 5. |  |  | CreateDate | DateTime |
| 6. |  |  | Owner | Integer |
| 7. | Comment | El modelo de la clase comment representa un comentario que los usuarios autenticados pueden agregar. Pueden Agregar fotos. Esto habilita a los usuarios a discutir sobre 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 |

# Controladores 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 de la Galería de Fotos. La acción obtiene todas las fotos de la base de datos y las pasa a la vista de la DisplayGallery.. |
| 2. |  | DisplayRecent(GET) | Esta acción es similar a la acción DisplayGallery, salvo que sólo se obtienen las fotos más recientes de la base de datos. Esta colección más pequeña de fotos se pasa a la vista de DisplayGallery.. |
| 3. |  | DisplayPhoto(GET) | Esta acción se ejecuta cuando el usuario hace clic en el enlace "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(POST) | Esta acción se ejecuta cuando el usuario hace clic en el enlace "Añadir una foto". La acción crea una nueva instancia de la clase de modelo Photo y la pasa a la vista AddPhoto. |
| 5. |  | DeletePhoto(GET) | Esta acción se ejecuta cuando el usuario hace clic en un enlace "Eliminar esta foto" en la vista DisplayPhoto. La acción muestra la vista DeletePhoto, que pide confirmación para la eliminación.. |
| 6. |  | DeletePhoto(POST) | Esta acción se ejecuta cuando el usuario hace clic en "Delete" en la vista DeletePhoto. La acción elimina la foto actual, con sus comentarios asociados, de la base de datos y redirige al usuario a la vista DisplayGallery. |
| 7. | Comment | DisplayComments(GET) | Esta acción se ejecuta cuando se muestra la vista DisplayPhoto. La acción requiere el PhotoID actual como parámetro y lo utiliza para obtener todos los comentarios de la foto actual de la base de datos. La acción devuelve la vista parcial \_DisplayComments. |
| 8. |  | AddComment(GET) | Esta acción se ejecuta cuando el usuario hace clic en el enlace "Agregar un comentario" en la vista DisplayPhoto. La acción crea una nueva instancia de la clase modelo Comment y establece su PhotoID como el ID de la foto actual. Pasa este nuevo comentario a la vista AddComment. |
| 9. |  | AddComment(POST) | Esta acción se ejecuta cuando el usuario hace clic en "Enviar" en la vista AddComment. La acción guarda los detalles del nuevo comentario en la base de datos y redirige al usuario a la vista DisplayPhoto. |
| 10. |  |  |  |

# 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. | PhotoController | 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 aparecen con un enlace "Añadir un comentario". |
| 3. | PhotoController | AddPhoto | Esta vista muestra un formulario que el usuario puede utilizar para cargar y describir una nueva foto. |
| 4. | PhotoController | 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 | Esta vista parcial, que se utiliza en el formulario DisplayPhoto, muestra todos los comentarios asociados con la foto actual. |
| 6. | CommentController | AddComments | Esta vista muestra un formulario que el usuario puede utilizar para crear un nuevo comentario para una foto. |

|  |  |
| --- | --- |
| **DisplayGalleryView** | **DisplayPhoto** |
|  |  |

# 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

*Inserte aquí las recomendaciones del servidor web.*

El autor recomienda utilizar Microsoft Azure para alojar la aplicación Photo Sharing. Microsoft Azure puede alojar cualquier sitio web ASP.NET, incluido la aplicación ASP.NET Core MVC propuesta en este documento. El escalado es muy simple porque Microsoft’s 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

*Inserte aquí las recomendaciones del servidor de base de datos.*

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