Adventure Works Photo Sharing Application (Proposed)

Detailed Planning Document

*Insert Date Here*

Authors: *Insert Your Name Here*

# Introduction

The author has examined the initial investigation document. Based on the use cases, technical requirements, and other content in that document, the author has created the detailed plans below. The board has already agreed that the photo sharing application will be built as a website based on Microsoft’s ASP.NET Core Razor Pages technology. Therefore the details presented here include the names and properties of model classes and controllers developers must create. Views have also been identified and wireframe diagrams included to help envision the user interface for important parts of the site.

The application design is likely to evolve throughout the development process as requirements change. The development team will adopt Agile practices to ensure such changes are reflected in the final product. Therefore this document should not be considered a complete definition of the final application.

# Model

Developers will create a model with the following model classes. For each model class, properties have been listed and descriptions given.

Table 1: Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Model Class | Description | Properties | Data Types |
| 1. | Photo | The photo model class represents a photo that authenticated users can upload to the website | PhotoID | Integer |
| 2. | Title | String |
| 3. | PhotoFile | Binary |
| 4. | Description | String |
| 5. | CreatedDate | Date |
| 6. | Owner | Integer |
| 7. | Comment | The comment model class represents a comment that authenticated users can add to photos. This enables users to discuss others’ photos. Each comment is associated with just one photo. | CommentID | Integer |
| 8. | User | String |
| 9. | Subject | String |
| 10. | Body | String |
| 11. | PhotoID | Integer |

# Razor Pages

Developers will create the following Razor Pages. For each page, methods have been listed and descriptions given.

Table 2: Razor Pages

|  |  |  |  |
| --- | --- | --- | --- |
|  | Page | Method | Description |
| 1 | Photos\Gallery | Get | The action runs when the user requests the Photo Gallery page. The action obtains all the photos from the database and passes them to the DisplayGallery partialview.  This partial view displays a collection of photos in the thumbnail size. For each photo the Title, Owner, and Created Date values are displayed. There is also a link to display the details of each photo and one to delete it. |
| 2 | Photos\Recent | Get | This action is similar to the Photos\Gallery Get action except that only the most recent photos are obtained from the database. This smaller collection of photos is passed to the DisplayGallery partialview |
| 3 | Photos\Details | Get | This action runs when the user clicks a photo's “Details” link in a gallery. The action obtains full details of a single photo from the database and renders it.  The Page displays a single photo in full size. The Title and Owner values appear above the photo. The Photo Name, Description, and other values appear beneath the photo. Under these details, all the comments for the current photo are listed, along with two textboxes to insert a Title and Body with a “Submit a Comment” button. |
| 4 | Photos\Upload | Get | This action runs when the user clicks an “Add a Photo” link. The action renders a Form to insert the details of the Photo and upload the file |
| 5 | Photo\Upload | Post | This action runs when the user clicks “Upload”. The action saves the file and details of the new photo to the database and redirects the user to the Gallery page. |
| 6 | Photos\Delete | Get | This action runs when the user clicks a “Delete this Photo” link in the Details Page. The action displays a requests confirmation for the deletion. |
| 7 | Photos\Delete | Post | This action runs when the user clicks “Confirm Delete” in the Delete Page. The action deletes the current photo, with its associated comments from the database and redirects the user to the Gallery Page.  This page displays the details of the current photo such as its title and description, along with a form that the user can use to confirm the deletion of a photo. |

# API Controllers

Developers will create the following API Controllers. For each controller, methods have been listed and descriptions given

|  |  |  |  |
| --- | --- | --- | --- |
|  | Controller | Action | Description |
| 8. | CommentsController | GetCommentsForPhoto (GET) | This action runs when the Details Page is displayed. The action requires the current PhotoID as a parameter and uses it to get all the comments for the current photo from the database. |
| 10. | AddComment (POST) | This action runs when the user clicks “Submit” in the Photos\Details view. The action saves the details of the new comment in the database and returns the newly created comment. |

# Blazor Components

Developers will create the following Blazor Components.

Table 3: MVC Views

|  |  |  |
| --- | --- | --- |
|  | Component | Description |
| 5. | DisplayComments | This component, which is used on the Photos\Details page, displays all the comments associated with the current photo |
| 6. | AddComment | This component displays a form that the user can use to submit a new comment for a photo |

# Hosting Recommendations

Since the photo sharing application will be developed in ASP.NET Core MVC, it may be hosted either on a Microsoft Windows or Linux web server. The author recommends the following hosting configuration:

## Web Server

The author recommends using Microsoft Azure to host the Photo Sharing application. Microsoft Azure can host any ASP.NET website, including the ASP.NET Core application proposed in this document. Scaling is very simple because Microsoft, not Adventure Works, is responsible for adding server resources at times of high traffic. Costs are minimal: they depend on the amount of data served to visitors but it is not necessary to maintain our own hardware.

## Database

The author recommends using SQL Database, within Microsoft Azure, to host the Photo Sharing application underlying database. As for the web server, this recommendation ensures high-availability hosting for the database with good value for money. This makes particular sense if the web site is hosted in Microsoft Azure.