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 MVC 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.

# MVC Model

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

Table 1: MVC Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Model Class | Description | Properties | Data Types |
| 1. | Photo | Класс фотомодели представляет собой фотографию, которую аутентифицированные пользователи могут загрузить на веб-сайт | PhotoID | Integer |
| 2. | Title | String |
| 3. | PhotoFile | Binary |
| 4. | Description | String |
| 5. | CreatedDate | Date |
| 6. | Owner | Integer |
| 7. | Comment | Класс модели комментариев представляет собой комментарий, который пользователи, прошедшие проверку подлинности, могут добавлять к фотографиям. Это позволяет пользователям обсуждать фотографии других пользователей. Каждыйкомментарий связан только с одной фотографией. | CommentID | Integer |
| 8. | User | String |
| 9. | Subject | String |
| 10. | Body | String |
| 11. | PhotoID | Integer |

# MVC Controllers

Developers will create the following controllers. For each controller, actions have been listed and descriptions given.

Table 2: MVC Controllers

|  |  |  |  |
| --- | --- | --- | --- |
|  | Controller | Action | Description |
| 1. | PhotoController | DisplayGallery (GET) | Действие выполняется, когда пользователь запрашивает страницу Фотогалереи. Действие получает все фотографии из базы данных и передает их в View (DisplayGallery) |
| 2. | DisplayRecent (GET) | Это действие похоже на действие DisplayGallery, за исключением того, что только самые последние фотографии получены из базы данных. Эта уменьшенная коллекция фотографий передается на View (DisplayGallery). |
| 3. | DisplayPhoto (GET) | Это действие выполняется, когда пользователь нажимает на ссылку "Подробности" фотографии в галерее. Действие получает полную информацию о одной фотографии из базы данных и передает ее на просмотр DisplayPhoto. |
| 4. | AddPhoto (GET) | Это действие выполняется, когда пользователь нажимает на ссылку "Добавить фотографию". Действие создает новый экземпляр класса модели Photo и передает его в представление AddPhoto. |
| 5. | AddPhoto (POST) | Это действие выполняется, когда пользователь нажимает кнопку "Сохранить" в представлении AddPhoto. Действие сохраняет файл и детали новой фотографии в базу данных и перенаправляет пользователя на представление DisplayGallery. |
| 6. | DeletePhoto (GET) | Это действие выполняется, когда пользователь нажимает ссылку "Удалить эту фотографию" в представлении DisplayPhoto. Действие отображает представление DeletePhoto, которое запрашивает подтверждение для удаления. |
| 7. | DeletePhoto (POST) | Это действие выполняется, когда пользователь нажимает кнопку "Удалить" в представлении DeletePhoto. Действие удаляет текущее фото с связанными с ним комментариями из базы данных и перенаправляет пользователя на представление DisplayGallery. |
| 8. | CommentController | DisplayComments (GET) | Это действие выполняется при отображении представления DisplayPhoto. Действие требует текущий PhotoID в качестве параметра и использует его, чтобы получить все комментарии для текущей фотографии из базы данных. Действие возвращает частичное представление \_DisplayComments представление. |
| 9. | AddComment (GET) | Это действие выполняется, когда пользователь нажимает на ссылку "Добавить комментарий" в представлении DisplayPhoto. Действие создает новый экземпляр класса модели Комментарий и устанавливает его PhotoID в качестве идентификатора текущей фотографии. Он передает этот новый комментарий к представлению AddComment. |
| 10. | AddComment (POST) | Это действие выполняется, когда пользователь нажимает кнопку "Отправить" в представлении AddComment. Действие сохраняет детали нового комментария в базе данных и перенаправляет пользователя на представление DisplayPhoto. |
|  |  |  |  |

# MVC Views

Developers will create the following views. Each view has been listed together with the controller it is associated with.

Table 3: MVC Views

|  |  |  |  |
| --- | --- | --- | --- |
|  | Controller | View | Description |
| 1. | PhotoController | DisplayGallery | Это представление отображает коллекцию фотографий в размере эскиза. Для каждой фотографии отображаются значения Заголовка, Владельца и Созданной Даты. |
| 2. | DisplayPhoto | В этом представлении отображается одна фотография в полном размере. Значения заголовка и владельца отображаются над фотографией. Имя фотографии, описание и другие значения отображаются под фотографией. В соответствии с этими деталями, все комментарии к текущей фотографии перечислены со ссылкой "Добавить комментарий". |
| 3. | AddPhoto | Это представление отображает форму, которую пользователь может использовать для загрузки и описания новой фотографии |
| 4. | DeletePhoto | В этом представлении отображается форма, которую пользователь может использовать для подтверждения удаления фотографии. Представление отображает детали текущей фотографии, такие как ее название и описание |
| 5. | CommentController | DisplayComments | Это частичное представление, которое используется в форме DisplayPhoto, отображает все комментарии, связанные с текущей фотографией |
| 6. | AddComment | Это представление отображает форму, которую пользователь может использовать для создания нового комментария для фотографии |

# Hosting Recommendations

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

## Web Server

*Insert web server recommendations here.*

## Database

*Insert database server recommendations here.*