

## Introduction

You will build upon the previous assignment to add support for rich text and uploading of media items (PDF documents, images, audio files, and videos).

Before you begin this assignment, you must finish your previous assignment. All tasks are to be made “on top” of your previous assignment.

This assignment is worth 10% of your final course grade. If you wish to submit the assignment before the due date, you can do that. If submitted after the due date, **you will receive a 10% deduction for every 24-hour period the assignment is late**. An assignment handed in **more than five days** late will receive a **mark of zero**.

## Getting started

Make sure Visual Studio is closed. In the Windows File Explorer, make a copy of your assignment 5 folder. Rename the folder to **your initials + “2247A6”**. For example, your professor would rename the old folder from “KY2247A5” to “KY2247A6”.

Inside the assignment 6 folder, rename the “.sln” file to the same name, for example “KY2247A6.sln”. Do not rename any other files in this folder. Start Visual Studio and open the solution file for assignment 6.

In the Solution Explorer, you will notice that the solution is named correctly however the project still uses the old name. You can rename the project by right-clicking it and choosing “Rename” or by pressing F2 when it is selected. Change the name to match the solution, for example “KY2247A6”.

Unfortunately, renaming these files will not change the name of the project subfolder or the namespaces used in your assignment. Do not be concerned, continue to use the old namespace as it is a lot of work to change it. The reason we renamed the solution and project was to avoid confusion if you have multiple projects open and so your professor will know they are marking the correct assignment.

Make sure your project still compiles and runs.

## Testing your work

While designing and coding your web app, use the Visual Studio debugger to test your algorithms, and inspect the data that you are working with.

In a browser, test your work by doing tasks that fulfill the use cases in the specifications.

## Customize the app's appearance

You have already customized the appearance of your web app in the previous assignment. You will need to make a few minor changes before submitting this assignment to Blackboard.

- In the “\_Layout.cshtml” file, change the title in the navbar to read “Show Biz II”.
- On the home page:
  - Change the <h1> tag to display “Assignment 6” followed by your name.
  - Change the large “Learn more >>” button to read “Assignment 6 on Azure”
  - Link the button to the URL of your assignment on Azure. Make sure you update this URL to point to the current assignment on Azure.

## Temporarily disable error-handling for HTTP errors 500 and higher

Optionally, you may wish to temporarily disable error-handling routines for HTTP 500 errors. Open the **Global.asax.cs** source code file. In the **Application\_EndRequest()** method, temporarily comment out the following statement:

```
if (code >= 500)...
```

*Remember to enable error-handling routines before publishing your assignment to Azure and submitting on Blackboard.*

## Preparing for rich text editing

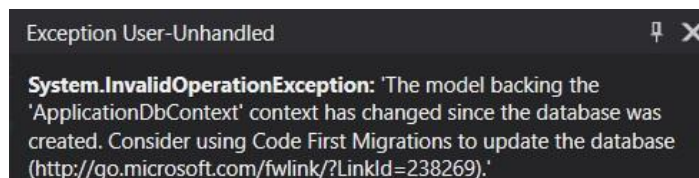
The **Actor**, **Show**, and **Episode** design model classes will need an additional **string** property to hold rich text. In all cases, the new property is not required and will not have a maximum length assigned.

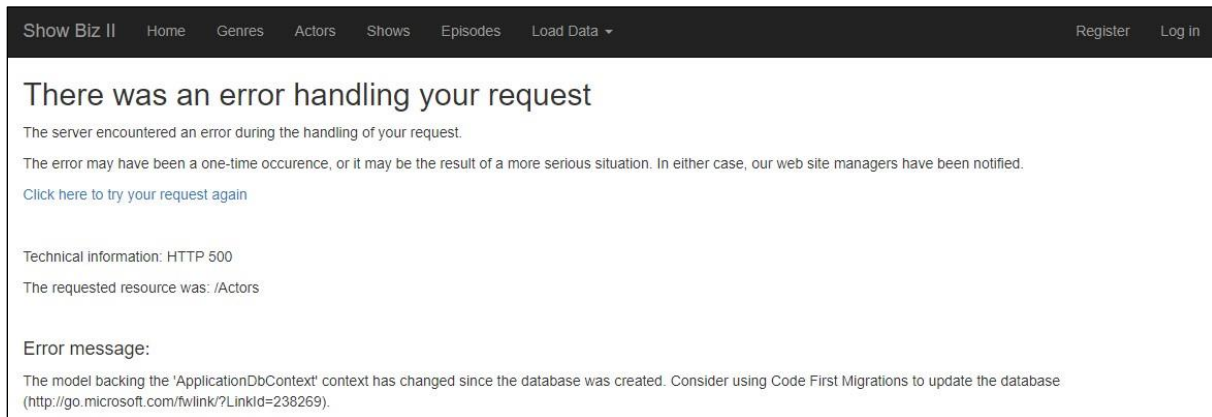
In the **Actor** class, add a new **string** property named “**Biography**”. The property will capture content about the early life and career of the actor.

In the **Show** class, add a new **string** property named “**Premise**”. The property will capture content about the basis for show.

In the **Episode** class, add a new **string** property named “**Premise**”. The property will capture content about the basis for episode.

Build the app and run it in a browser. Navigate to the “Actors” page. You will probably receive an error:





As the error states, the data model has changed. In other words, the schema of your database no longer matches the schema of your design model classes. You can resolve the error using migrations. Review the course notes and code examples.

You already enabled migrations in the previous assignment *before you published it to Azure*. The following commands should add the new migration and update the schema of the database.

1. add-migration "descriptive-name-for-migration"
2. update-database

## Include "Biography" for the "add new" use case for Actor

You already coded the "add new" use case for the **Actor** object but this was before adding the **Biography** property. Make the necessary changes to the view models, controllers, and views to allow adding of rich text.

Remember this "recipe" for enabling rich text editing and display. Refer to the code examples for help.

View Model(s):

- Add the **Biography** property to the **ActorAddViewModel** (and **ActorAddFormViewModel** if you coded it).
- Add **[DataType(DataType.MultilineText)]** to the **Biography** property in the view model class.

View:

- The "Create" view was already scaffolded in the previous assignment. You may choose to scaffold it again, but that may result in a lot of work. Instead, carefully add a new "form-group" with the appropriate controls. You can copy from code examples or other properties in the same view but be careful to rename all instances of the property name.
- In the "Scripts" section, add a reference to the "CKEditor" script on the CKEditor CDN. Add the script below the "~/bundles/jqueryval" line.
- In the "Scripts" section, add code to convert the <textarea> into a rich text editor.

Controller:

- In the controller method that receives the data submitted by the user (the POST method), add the **[ValidateInput(false)]** data annotation.

The screenshot displays a web application interface for creating a new actor. The top navigation bar includes links for 'Show Biz II', 'Home', 'Genres', 'Actors', 'Shows', 'Episodes', and 'Load Data'. The user is logged in as 'Hello exec@example.com!'. The main heading is 'Create Actor'. The form contains the following fields:

- Name**: A text input field.
- Alternate Name**: A text input field.
- Birth Date**: A date picker with a placeholder 'yyyy-mm-dd'.
- Height (m)**: A text input field.
- Image**: A text input field.
- Biography**: A large text area with a rich text editor toolbar. The toolbar includes icons for bold, italic, strikethrough, text color, background color, bulleted list, numbered list, link, unlink, source, and other formatting options.

A 'Create' button is located at the bottom of the form.

## Modify the “get one” use case to display the “Biography”

When displaying rich text, use the **Raw()** HTML Helper. Place the biography in a bootstrap well.

If the **Biography** property is null, an empty string, or whitespace, do not display the biography on the page.

[Show Biz II](#) [Home](#) [Genres](#) [Actors](#) [Shows](#) [Episodes](#) [Load Data ▾](#) Hello exec@example.com! [Log off](#)

---

## Bryan Cranston

**Name**

Bryan Cranston

**Alternate Name**


**Birth Date**

1956-03-07

**Height (m)**

1.79

**Image**



**Executive**

exec@example.com

---

**Appeared In**

Breaking Bad  
2 shows  
Better Call Saul

---

**Biography**

**Bryan Lee Cranston** (born March 7, 1956) is an American actor, director, and producer who is mainly known for portraying *Walter White* in the AMC crime drama series *Breaking Bad* (2008–2013) as well as its prequel series *Better Call Saul* (2015-2022) and *Hal* in the Fox sitcom *Malcolm in the Middle* (2000–2006). He has received a number of awards—including six Primetime Emmy Awards, four Screen Actors Guild Awards, two Tony Awards, and a Golden Globe Award—with a nomination for an Academy Award and a BAFTA Award.

[Add New Show](#) | [Back to List](#)

[Show Biz II](#) [Home](#) [Genres](#) [Actors](#) [Shows](#) [Episodes](#) [Load Data ▾](#) Hello exec@example.com! [Log off](#)

---

## Dwayne Douglas Johnson

**Name**

Dwayne Douglas Johnson

**Alternate Name**

Dwayne "The Rock" Johnson


**Birth Date**

1972-05-02

**Height (m)**

1.96

**Image**



**Executive**

exec@example.com

---

**Appeared In**

0 shows

The biography is not displayed since it is null, empty, or just whitespace.

[Add New Show](#) | [Back to List](#)

## Collect and display the “Premise” for Show and Episode use cases

Mimic the steps outlined in the previous sections to collect and display the **Premise** property for the **Show** and **Episode** objects.

## Show

Show Biz II

Home

Genres

Actors

Shows

Episodes

Load Data ▾

Hello exec@example.com!

Log off

## Add Show for Bryan Cranston

Name

Release Date

📅

Image

Genre

Biography

▾

Actors

☒ Bryan Cranston

☐ Dwayne Douglas Johnson

☐ Julia Louis-Dreyfus

Premise

🗑️

📄

📁

📅

📅

↶

↷

🔍

🔗

🔗

🚩

🖼️

📊

📊

Ω

🔗

🔗

📄

Source

**B**

*I*

S

I<sub>x</sub>

≡

≡

≡

≡

”

Styles ▾

Format ▾

?

Create

[Show Biz II](#) [Home](#) [Genres](#) [Actors](#) [Shows](#) [Episodes](#) [Load Data](#) [Hello exec@example.com!](#) [Log off](#)

## Breaking Bad

**Name**

Breaking Bad


**Genre**

Drama

**Release Date**

2008-01-20

**Image**



**Coordinator**

coord@example.com

**Cast**

Bryan Cranston

1 actors

**Episodes**

Pilot

3 episodes

Cat's in the Bag...

...And the Bag's in the River

**Premise**

Set in *Albuquerque, New Mexico*, between 2008 and 2010,[7] **Breaking Bad** follows *Walter White*, a modest high school chemistry teacher who transforms into a ruthless kingpin in the local methamphetamine drug trade, driven to financially provide for his family after being diagnosed with inoperable lung cancer. Initially making only small batches of meth with his former student *Jesse Pinkman* in a rolling meth lab, Walter and Jesse eventually expand to make larger batches of special blue meth that is incredibly pure and creates high demand. Walter takes on the name "Heisenberg" to mask his identity. Because of his drug-related activities, Walter eventually finds himself at odds with his family, the Drug Enforcement Administration (DEA) through his brother-in-law *Hank Schrader*, the local gangs, and the drug cartels, putting him and his family's lives at risk.

Do not display the premise if it is null, empty, or just whitespace.

Show Biz II

Home

Genres

Actors

Shows

Episodes

Load Data ▾

Hello coord@example.com!

Log off

## Add Episode to Breaking Bad

Name

Season

Episode

Date Aired

📅

Image

GenreBiography▾

Premise

✂️ 📄 🗒️ 📖 📝 ⬅️ ➡️ ↻ 🔍 🚩 🖼️ 🏠 📺 ⌕ 🔧 🔄 📁 Source

B I S T<sub>x</sub> ¶ ⑆ ⑈ ⑉ Styles ▾ Format ▾ ?

Create

Page 8 of 16



## Prepare for non-text media items

In this app, two scenarios will be implemented for non-text media type handling.

In the simple scenario, the existing **Episode** entity will be modified to handle a video media item.

In the more complex scenario, a new **ActorMediaItem** entity will be created. This new entity will be dedicated to the description and storage of various types of media items. The existing **Actor** entity will be modified to handle a collection of **ActorMediaItem** objects.

## Add a video media item to the “Episode” design model class

The existing **Episode** entity will be modified to handle a video media item. You will need two new properties, **VideoContentType** and **Video**.

Follow the guidance in the lecture notes and in the **PhotoProperty** code example. Remember to add a migration and update the database after changing your design model classes.

## Modify the “add new” episode use case to include uploading a video

Locate the appropriate “add form” view model class and add a **VideoUpload** property. Note that the data type of the property is a **string** and that the **DataType** attribute specifies **DataType.Upload**.

Next, locate the view and add the form control. *Don’t forget to change the form encoding to accept multipart form data.*

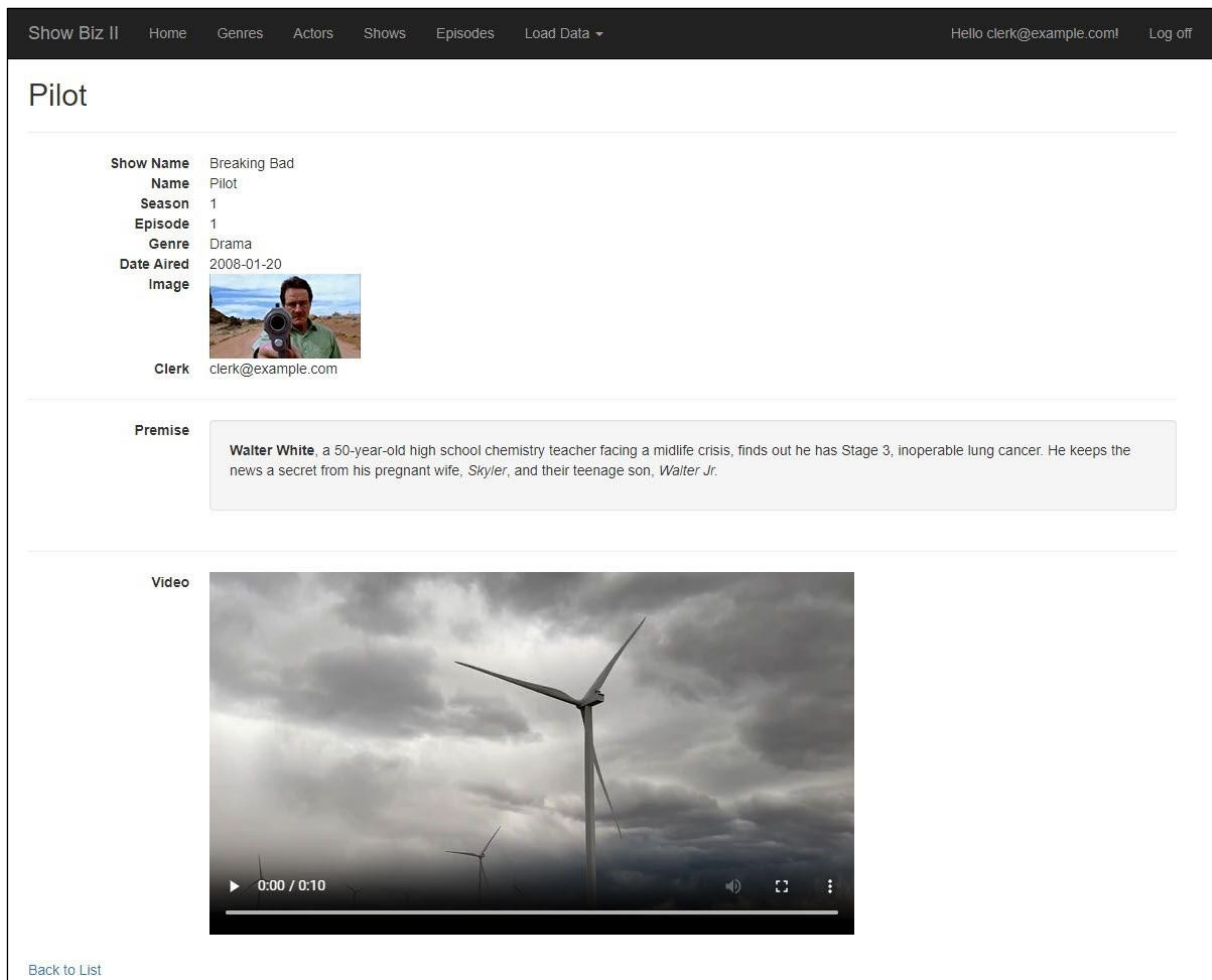
The screenshot shows a web application interface for adding a new episode to the show 'Breaking Bad'. The form is titled 'Add Episode to Breaking Bad'. It contains several input fields: 'Name' (a text box), 'Season' (a dropdown menu set to '1'), 'Episode' (a dropdown menu set to '1'), 'Date Aired' (a date picker set to '2023-04-02'), 'Image' (a text box), and 'Genre' (a dropdown menu set to 'Biography'). Below these is a 'Premise' section with a rich text editor. At the bottom, there is a 'Video Attachment' section with a 'Choose File' button and a 'Create' button. A red arrow points to the 'Choose File' button.

Modify the controller to accept the file transfer and save the information to the database.

Finally, modify the details view to show the video clip. To implement this functionality, there are a few things you will need to do:

- Create an **EpisodeVideoViewModel** that contains the properties needed to deliver the video clip to the browser.
- Create a **Manager** method and call it something like **EpisodeVideoGetById()**. This method will return the video information in the form of an **EpisodeVideoViewModel**.
- Add an action to the **EpisodesController** to deliver the video clip to the browser. You can use attribute routing and set up a route like `"/Episodes/Video/{id}"`. Return a **File()** response instead of a **View()** response.
- Modify the view and add a `<video>` tag configured like `<video controls src="">`. Only render the video tag if a video exists. You may need to add the **VideoContentType** property to allow for this check but do not add the **byte array**. Use **Url.Action()** to build the URL.

When finished, your page will look the following.



## Add a media item collection to the Actor entity

In this scenario, the design and coding approach will be like the one demonstrated in the **PhotoEntity** code example. The main difference between the code example and your assignment is that you will need to permit the user to upload several types of media items. In this assignment, ensure you can support photos (\*.jpg, \*.jpeg, \*.png), audio files (\*.mp3), video files (\*.mp4), or PDF documents (\*.pdf).

You will begin by creating a new **ActorMediaItem** entity. Be sure to include the properties: **Id**, **ContentType**, **Content**, and **Caption**. Refer to the code example if you are unsure about the data types or how to implement these properties.

Modify the **Actor** entity to hold a collection of **ActorMediaItem** objects. Ultimately, you are building a one-to-many relationship, that is a single actor can hold many media items.

Don't forget to add the **DbSet<>** property to the **ApplicationDbContext**.

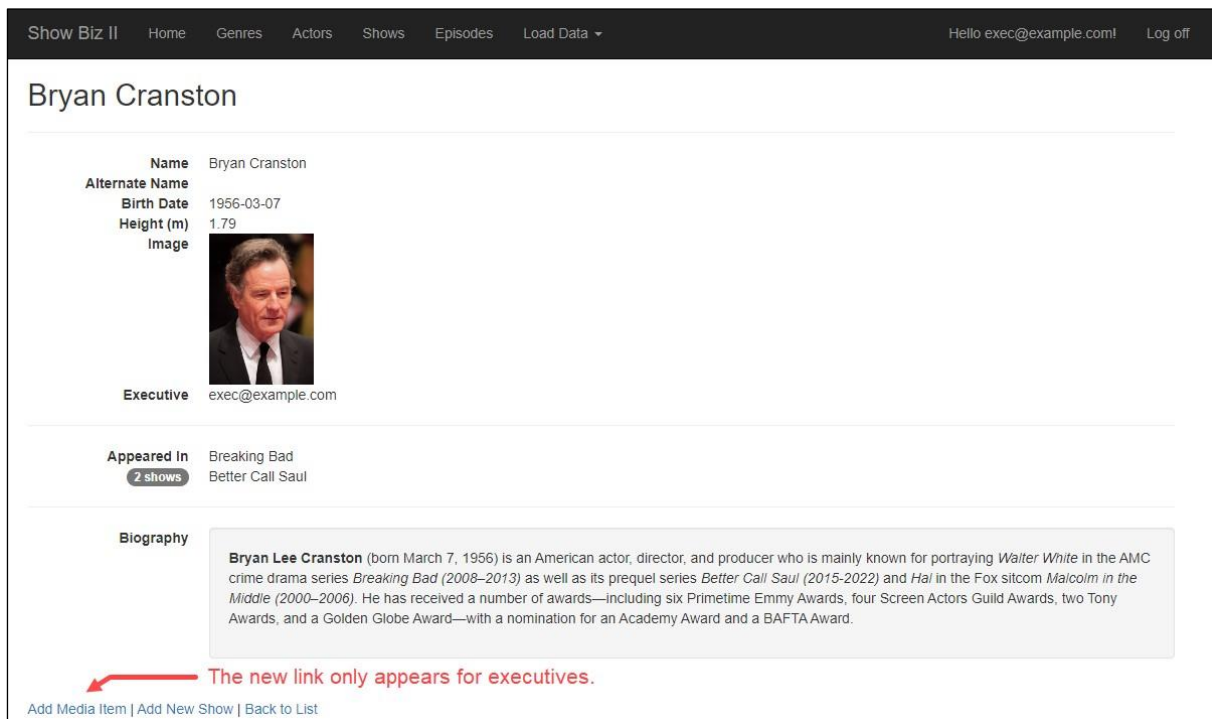
Once again, use migrations to modify the database schema.

## Implement the “add new” use case for “ActorMediaItem” entities

To implement the “add new” use case for **ActorMediaItem** entities, follow these steps. Remember, you should refer to the **PhotoEntity** code example for help.


- You will need to create both the “add form” and “add” view models:
  - ActorMediaItemAddFormViewModel** – Include the **Caption** and **ContentUpload** string properties. You may need an **ActorId** and **ActorName** properties as well.
  - ActorMediaItemAddViewModel** – In this view model, the **ContentUpload** property will have the **HttpPostedFileBase** data type. You will also need the **Caption** and **ActorId** properties as well.
- Modify the **ActorsController**. Include two **AddMediaItem** actions for the GET and POST tasks. Uploading of media items can only be done by executives.
- Add a method to the **Manager** class to add the media item to the database. You can name it **ActorMediaItemAdd()**.
- Scaffold the “add new” view. Modify it to include the name of the actor in the header. The **ActorName** property cannot be edited and the **ActorId** property is stored as a hidden form field. A screenshot is provided below.
- Update the “details” actor view to include a link to the add new media item view. The link should only display for users that are part of the **Executive** role.

### Actor “get one”



Show Biz II Home Genres Actors Shows Episodes Load Data ▾ Hello exec@example.com! Log off

## Bryan Cranston

Name	Bryan Cranston
Alternate Name	
Birth Date	1956-03-07
Height (m)	1.79
Image	
Executive	exec@example.com

Appeared In

- Breaking Bad
- 2 shows** Better Call Saul

Biography

**Bryan Lee Cranston** (born March 7, 1956) is an American actor, director, and producer who is mainly known for portraying *Walter White* in the AMC crime drama series *Breaking Bad* (2008–2013) as well as its prequel series *Better Call Saul* (2015-2022) and *Hal* in the Fox sitcom *Malcolm in the Middle* (2000–2006). He has received a number of awards—including six Primetime Emmy Awards, four Screen Actors Guild Awards, two Tony Awards, and a Golden Globe Award—with a nomination for an Academy Award and a BAFTA Award.

[Add Media Item](#) | [Add New Show](#) | [Back to List](#)

**The new link only appears for executives.**

## ActorMediaItem “add new”

## Show media items on the Actor “get one” page

To implement the changes to the “get one” use case for the **Actor** entity, follow these steps.

- Create an **ActorMediaItemBaseViewModel** that contains the **Id**, **Caption**, and **ContentType** properties.
- Create an **ActorMediaItemWithContentViewModel** that contains the Content byte array. It inherits from the base view model.
- Modify the **ActorWithShowInfoViewModel** to include:
  - A property called **ActorMediaItems** that holds many **ActorMediaItemBaseViewModel** objects.
  - A property called **Photos** that holds many **ActorMediaItemBaseViewModel** objects.
  - A property called **Documents** that holds many **ActorMediaItemBaseViewModel** objects.
  - A property called **AudioClips** that holds many **ActorMediaItemBaseViewModel** objects.
  - A property called **VideoClips** that holds many **ActorMediaItemBaseViewModel** objects.
- In the **Manager** class:
  - Modify the **ActorWithShowInfoGetById()** method to include **ActorMediaItems**.
  - Add a method **ActorMediaItemGetById()** method to return an **ActorMediaItemWithContentViewModel**.
- In the **ActorsController**:
  - **Details** action - Ensure you properly copy the appropriate media items from the **ActorMediaItems** property to the **Photos**, **Documents**, **AudioClips**, and **VideoClips** properties. Media items in each property must be sorted by the **Caption**.
  - Add **MediaItemDownload** action to deliver the media item to the browser. You can use attribute routing and set up a route like “/Actors/MediaItem/{id}”. The **id** refers to the identifier of the **ActorMediaItem** entity. Return a **File()** response instead of a **View()** response.
- Modify the “get one” view as shown in the screenshot.
  - Carefully read and implement any notes/tasks included in the screenshot.
  - You must hide any media item sections where there are no media items present.
  - Remember to sort media items by the **Caption**.
  - The caption can be displayed within a `<small>` html tag.

- All media types will display in the browser window except PDF documents. ▪ Use the **Url.Action()** helper.
- PDF documents will display as a link surrounding the caption and the icon provided. Clicking the link must download the file to your computer and must not open the document within the browser. The name of the file should include the “.pdf” extension.

**Biography**

**Bryan Lee Cranston** (born March 7, 1956) is an American actor, director, and producer who is mainly known for portraying *Walter White* in the AMC crime drama series *Breaking Bad* (2008–2013) as well as its prequel series *Better Call Saul* (2015–2022) and *Hal* in the Fox sitcom *Malcolm in the Middle* (2000–2006). He has received a number of awards—including six Primetime Emmy Awards, four Screen Actors Guild Awards, two Tony Awards, and a Golden Globe Award—with a nomination for an Academy Award and a BAFTA Award.

**Photos**

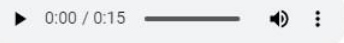
Macbeth on Broadway

← `<img height="200">`← `<small>`**Documents**

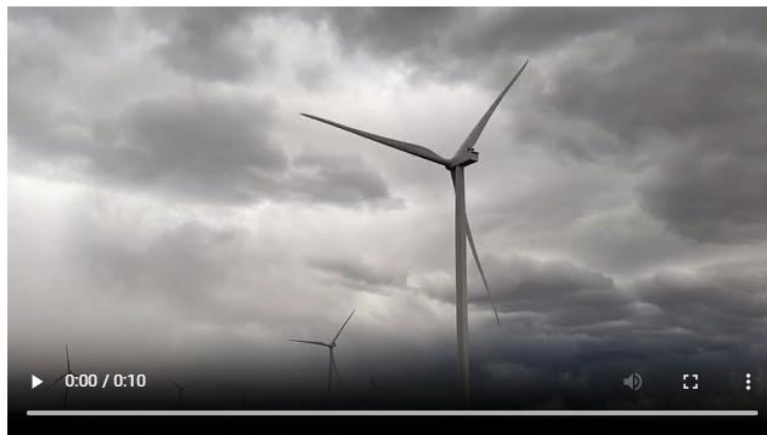
Wikipedia

← `<img>` (icon provided)← `<small>`

Wrapped with an `<a>` tag.  
Clicking the link will  
download the PDF

**Audio Clips**

Autobiography Sample Clip

← `<audio controls>`← `<small>`**Video Clips**

Docuseries Trailer

← `<video controls>`← `<small>`

[Add Media Item](#) | [Add New Show](#) | [Back to List](#)

## Publish to Azure

Follow the guidance in the Azure assignment for help to deploy/publish your web app to Azure. Create a new database and web app for this assignment. Do not reuse any of the services deployed in the previous assignment.

Use the following names for your Azure resources (replacing **kenneth** with your Seneca ID)

- Database server: **kenneth-ds-web524**
- Database: **kenneth-db-web524-a6**
- Web app: **kenneth-wa-web524-a6.azurewebsites.net**

**Reminder:** The “Learn more >>” button on the project home page must be customized to link to your assignment on Azure. Your professor will use this link to test and mark your assignment.

Once published to Azure, make sure you load the roles, genres, actors, shows, and episodes to the database. Create the user accounts as outlined in the previous assignment.

Your professor will test on Azure. If there are malfunctions due to incorrectly publishing to Azure, you may lose up-to 25% of your assignment mark.

## Reminder about academic integrity

You must comply with [Seneca College’s Academic Integrity Policy](#). Although you may interact and collaborate with others, this assignment must be worked on individually and you must submit your own work.

You are responsible to ensure that your solution, or any part of it, is not duplicated by another student. If you choose to push your source code to a source control repository, such as GIT, ensure that you have made that repository private.

A suspected violation will be filed with the Academic Integrity Committee and may result in a grade of zero on this assignment or a failing grade in this course.

## Submitting your work

**Before submitting your assignment, remember to update the home page link so that it points to the project you deployed on MS Azure. Your site will be marked using MS Azure so make sure to test on the cloud as well as locally!**

Make sure you submit your assignment before the due date and time. It will take a few minutes to package up your project so make sure you give yourself a bit of time to submit the assignment.

The solution folder contains extra items that will make submission larger. The following steps will help you “clean up” unnecessary files.

1. Locate the folder that holds your solution files. You can jump to the folder using the Solution Explorer. Rightclick the “Solution” item and choose “Open Folder in File Explorer”.
2. Go up one level and you will see your solution folder (similar to **KY2247A6** but using your initials). Make a copy of your solution and change into the folder where you copied the files. For the remainder of the steps, you should be working in your copied solution!
3. Delete the “packages” folder and all its contents.
4. In the project folder (should be called **KY2247A6** but using your initials) contained within the solution folder, delete the “bin” and “obj” folders.
5. Compress the copied folder into a **zip** file. **Do not use 7z, RAR, or other compression algorithms (otherwise your assignment will not be marked)**. The zip file should not exceed a couple of megabytes in size. If the zip file is larger than a couple of megabytes, do not submit the assignment! Please ensure you have completed all the steps correctly.
6. Login to <https://learn.senecapolytechnic.ca/>. Open the “Web Programming Using ASP.NET” course area and click the “Assignments” link. Follow the link for this assignment.
7. Submit/upload your zip file. The page will accept unlimited submissions so you may re-upload the project if you need to make changes but make sure you make all your changes before the due date. Only the last submission will be marked.
8. It is highly recommended you download the submitted assignment and test it on your computer.