This document includes my assumptions and decisions taken for the MVC application that I have built.

#### The tasks and how I achieved them

This section includes my initial ideas of how to achieve each task The green coloured text are comments I added at the end describing how I decided to achieve each task.

1.Allow the user to enter a variable amount of numbers, of any integer value and in random order.

* Input must be integers
  + Needs to include validation
  + This application includes client-side and server-side validation as described in point 4
* Can be inputted unordered
* How will these numbers be inputted?
  + Delimiter separated
  + Row of input boxes
  + Decided to get the user to input a comma separated set of numbers as it's the easiest to implement. If I had a bit more time, I would allow the user to input 1 number at a time and this number would be added to an array of numbers which would then get submitted

2. Sort these numbers in either ascending or descending order – the user should choose the order.

* SORTING METHOD IS IMPORTANT
  + Choose which sorting method is best
  + Quicksort was decided upon as the sorting algorithm is fast when used with larger datasets and easy to implement.
* Ascending/Descending toggle
  + Dropdown or radio button
  + Decided to use a Drop-down when submitting the dataset

3. The ordered sequence should be inserted into the database along with the direction that the sequence was sorted in and the time taken to perform the sort.

* Insert into MS SQL database table
* Database design
  + ID
    - bigint/long
  + Ordered Numbers
    - How are these being displayed?
      * Delimiter separated?
    - Decided to store as comma-separated integer array as a varchar
  + Sort Direction
    - bit/boolean
    - sort\_direction use a smallint for the final
  + Time taken to perform the sort
    - Time taken in milliseconds
    - bigint?
    - Used the time\_taken field as bigint
  + Is sorted?
    - Has the array been sorted?
    - Required for asynchronous requests
    - Hidden field in the final version as request currently isn't asynchronous
* Final Design
  + Table - SortedNumbers
    - id - bigint
      * The Primary Key for the table
    - sorted\_array - varchar
      * I decided to store this as a comma-separated string. This allows the sorted array to be displayed easily in the List page
    - sorted\_direction - smallint
      * Displays the sort direction for the sort. This is stored as a 0 (Descending) or a 1 (Ascending). I considered using the 'bit' type but an integer will be easier to read within the SQL table.
    - time\_taken - bigint
      * Displays the time taken to sort the array in milliseconds. This is calculated using the Stopwatch class starting before the sort and stopping after the sort. This often shows as 0ms as

4. Feedback to the user the result of the operation (i.e., whether the operation was successful, any validation issues with the submission or any errors that occurred).

* Output whether the operation was successful or not
  + Output the error if not unsuccessful with details on what went wrong
* Client-side validation
  + For this, there is a check whether the text box isn't blank, such as setting the text box as required. This way, the data isn't submit and indicates the required data that's missing
* Server-side validation
  + For this, the application checks whether the input is correctly formatted. This is done by including Data Annotations and Regex with the appropriate variables within the Model object. If this fails, the user is returned back to page with a message saying that the input isn't valid. There's also an additional check to this to make sure that the inputted integers, as a string, can be parsed to an integer and returns an error if not.

5. Display the results of all sorts including the sort direction and time taken.

* Output details of rows
  + Display in a table
  + Assuming a single JSON is outputted, include 'Actions' at the end of each row with an option to export the row data
* Stored data in a table and retrieves the data as a list to output to the web page in a table. This allows the User to view all of the sorts as well as the time taken and the sort direction effectively.

6. Allow the user to export all of the sorts as JSON.

* Transform table into JSON object
* Is this a batch output or one at a time?
* Is the export outputting to a JSON file?
* Incomplete as I had ran out of time to complete this task

#### Improvements that can be made

* The main improvement that can be made is to improve how the user inputs the array on the New page. For this, I would display the current array of numbers as a label and a text box which allows the user to input a single integer and submit that number to the array. When the number is submitted, the number is added to the number array in the label. The reason that this method wasn't implemented was due to time constraints and lack of experience in using ASP.NET. If given a bit more time, I would be able to make this improvement and it would make the application a lot better as the user doesn't have to input in a specific format.
* The next improvement would be to handle the validation from the client-side through the use of Regex. This would mean that the form wouldn't have to be submitted before a message is displayed to show an error.
* Another improvement that can be made is to run the sort on a separate thread that runs in the background. This is something that I originally planned to include but time constraints made it difficult to achieve.

#### Final Thoughts

I enjoyed creating this application and learning about ASP.NET MVC and Razor pages but I think there's a lot that I could have improved upon and, with more experience using ASP.NET MVC and Razor pages, it could be much better. I'll use this application to continue my learning of ASP.NET and be able to show a better version in the future.