TASK - 4 - Task Form Inputs Validation (Asynch approx 3 hours)

- **All form fields are validated on form submission(Name, Description, AssignedTo, DueDate, Status).**
- **A meaningful error message is displayed when a form field is invalid.**
- **The JavaScript code is in a separate file and the file is included in the HTML page with no errors.**

Details are below

Implement a form that captures the fields required to create a task.

Requirements:

- **Create** a JavaScript function called "validateTaskForm" that verifies that the inputs inserted by the user in the task form are correct:
    - Name -> Not Empty and longer than 8 characters
    - Description -> Not Empty and longer than 15 characters
    - AssignedTo -> Not Empty and longer than 8 characters
    - DueDate -> Not Empty and not less than current date

Verify that your code works as expected.

TASK - 5 - Displaying Date (Asynch approx 3 hours)

- **A span element is incorporated into the HTML seemlessly and in a user friendly spot.**
- **A Date object is created as soon as the page loads of the current time.**
- **The Date object is then displayed in a user friendly format of day, month, year and displayed inside the span element.**

Details are below

Implement a method which converts the date in to a human readable format (All the dates which has to be displayed - Date)

Example: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date

TASK - 6 - Create a Class Using JavaScript and Add Tasks programmatically (Asynch approx 3 hours)

- **A TaskManager class using JavaScript is created in a separate JS file.**

- **When a new task is added with valid information, the data should be stored inside a JavaScript object. Each task object should be added to and stored in an array variable. They should each have a unique incremented id.**

- **The added task should be visible on the current tasks list and should display the task information.**

Details are below:

Implement a JavaScript function and logic that will handle the tasks model.

Requirements:

- **Modify** the TaskManager class by adding a method that takes as an argument a task JSON object and creates a Card Layout HTML as defined on previous tasks (Take a look at this code as reference).
- **Add** an id attribute to the content list group created in previous tasks and **write** the code to add the card to your HTML element (take a look at the addItem function as reference).
- **Verify** that your code works as expected

How to achieve Requirements: (Provided as reference only, You can use your own thought process here. **this is just an EXAMPLE**)

- **Define** a the object structure to represent a task using JavaScript with the following fields:

    - ID -> Int
    - Name -> String
    - Description -> String
    - AssignedTo (person responsible for completing the task) -> String

    - DueDate -> Date when the task is due
    - Status (TODO, IN PROGRESS, REVIEW and DONE) -> String

- **Define** a TaskManager Class that will implement the following functions:

    - Get Tasks -> returns the list of ALL tasks
        - *function getAllTasks()*
    - Get all Tasks with a given status -> returns a list of all tasks where a status equal to the status passes as an argument:
        - *function getTasksWithStatus(status)*
    - Add Task -> a task to existing Tasks List
        - *function addTask(task)*

Note: Input the data from form, Validate the data and store that data in object format in local storage.

TASK - 7 - Display Tasks (Asynch approx 3 hours)

- **Successfully implemented createTaskHTML() function and render() method.**

- **Each time a new task is added, the render() method is called to display the new task.**

- **The added task should be visible on the current tasks list and should display the task information.**

Details are below:

Implement a JavaScript function that allows tasks to display once created.

Requirements:

- **Add** a card once created with all the details of task.
- **Add** a delete button that will call the *deleteTask* function (Note: just a button, you do not need to add functionality).
- **Verify** that your code works as expected.