**Use Case:**  
Build an order record page consisting of 2 LWC components

1. one which displays available products and
2. one which displays order products from Order

**Acceptance Criteria:**

1. Orderable product will be displayed in a 2-column list displaying *Name* and *List Price* (component 1)
   1. Products are orderable when they have a Pricebook Entry in the Pricebook related to the current order (standard pricebook for this assignment) and when that Pricebook Entry is active
   2. Each product can only appear once in the list
2. The UI needs to provide the ability for the user to add a product from the list (component 1) to the order
   1. When the same product is not yet added to the order it will be added with a quantity of 1
   2. When the product already exists the quantity of the existing order product be increased by 1
3. All Order Products in the current order will be displayed in a table displaying the *Name*, *Unit Price, Quantity* and *Total Price* (component 2)
   1. When the user adds a new product or updates an existing product on the order (see point 2) the list needs to display the newly added
4. A test coverage of at least 80% for both APEX components is required.
5. The use of Aura components is prohibited.
6. Create a Salesforce Developer login for this assignment and build it as a SFDX project.

Once you build this, create a private repository on GitHub, check in all the components and share the repository path & login details with us.

**Tips**

* Try to use OOTB LWC components provided by Salesforce, see also the component library from Salesforce: <https://developer.salesforce.com/docs/component-library/overview/components>
* Make sure you commit the Order Flexi page to git
* Apply best practices, we will be reviewing both the functionality and implementation of your solution
* To avoid any minus points when your code is not deployable, include short video of your solution that covers the Acceptance Criteria and commit it to git
* Document your code to help us understand the technical choices you made when building the solution

**Extra Acceptance Criteria:**

1. The number of products can exceed 200; the solution needs to be able to handle this while providing a proper user experience.
2. To ensure an optimal user experience the page should not be reloaded and only the changed or new items should be refreshed/added
3. The end user needs to be able to confirm the order in an external system with the click of a button.
   1. The request format expected by the external system should follow the following JSON structure:

{

   "accountNumber": "",

    "orderNumber": "",

    "type": "order type",

    "status": "order status",

    "orderProducts": [{

        "name": "product name",

        "code": "product code",

        "unitPrice": 10.00,

        "quantity": 1

    }]

}

* 1. Request is sent as POST
  2. Order of the JSON fields in the above JSON structure is not relevant but the data type is.
  3. Errors and time-outs of the external system need to be handled
     1. All 200 responses are considered OK
     2. Any non-200 response is handled as ERROR
  4. For this use case generate a new endpoint URL at <https://requestcatcher.com/>

1. After the order is confirmed successfully in the external system the status of the order and order items will be updated to “Activated”
   1. When activated the end user will not be able to add new order items or confirm the order for a second time.
2. A test coverage of at least 80% for both LWC components is required.