Part 1: Trigger

Assumptions/out of scope

- Assuming formula field is not an option.
- Scope is only the contact created/updated, not all contacts tied to the same account.
- Assuming the ‘Industry’ field will be pulled from primary Account.
- Assuming logic will transfer any (including null) to contact.
- Assuming the logic runs on any update (including a dummy update, with no real change to any fields)
- Assuming the ‘Account Industry’ field in Contact cannot be edited by user or other processes, otherwise there is a risk of value getting overwritten.
    - The only place this field is updated is in Trigger while creation and update.
- Keeping scope only to system administrators
    - I have kept access settings to default ones selected at the time of creation.
    - I have not created runAs tests.
- Did not consider the undelete scenario.

Implementation Details

- Created field
    - Name: Account Industry
    - Type: Text
    - Length: 255 → Size of standard picklist, since we are assigning data from standard Industry(picklist field)
- Created a simple version of Kevin O’Hara’s TriggerHandler framework.
    - A virtual class which has a concrete implementation of ‘run’ method, which resolves the Trigger Context and calls the virtual methods (beforeInsert, beforeUpdate etc)
    - Extending the TriggerHandler class overrides the corresponding virtual Methods.
- Created Trigger on Contact Object.
- Create a Trigger handler class ‘ContactTriggerHandler’
    - Implemented ‘beforeInsert’ and ‘beforeUpdate’, since we are pulling the information from a related object, we can set the value in before context without explicitly running a DML.
    - The implementation creates a Map<AccountId,List<Contact>>, queries the Account for Industry field and assigns the value in Contact’s ‘Account Industry’ field.
- Created a Test Class ‘ContactTriggerHandlerTest’.

Part 2: lwc

Assumptions

1) Assumed it was okay to use lightning:datatable.
    a) Lightning:datatable allows Selectable Row.
    b) Lightning:datatable highlights the row selected in different color
    c) There was no explicit requirement between Next button and the row selection, so I thought it was okay to use row action.

Implementation Details

1) The Visualforce page is exposed as a Tab. ‘LWC Demo’
2) ‘accountList’ lwc component is also exposed directly through tab ‘Account List’

Component	Type	Remarks
LWCDemo	Visualforce page	Uses lightning:out to render lwc component
lwcdemocontainer	Aura App	Used in lightning:out
accountList	lwc	Wrapper component that lists the Account and also has the child component (accountDetails)
accountDetails	lwc	1) Child component that shows details of account. 2) Record Id from parent is passed through @api property. 3) Publishes event 2 event for parent. a) ‘detailloaded’ -> used to turn off the spinner b) ‘backevent’ -> used to hide ‘detail’ view and show the list view. 4) Everytime this component is rendered a call is made to get the Account information
AccountController	Apex	@AuraEnabled methods to support lwc