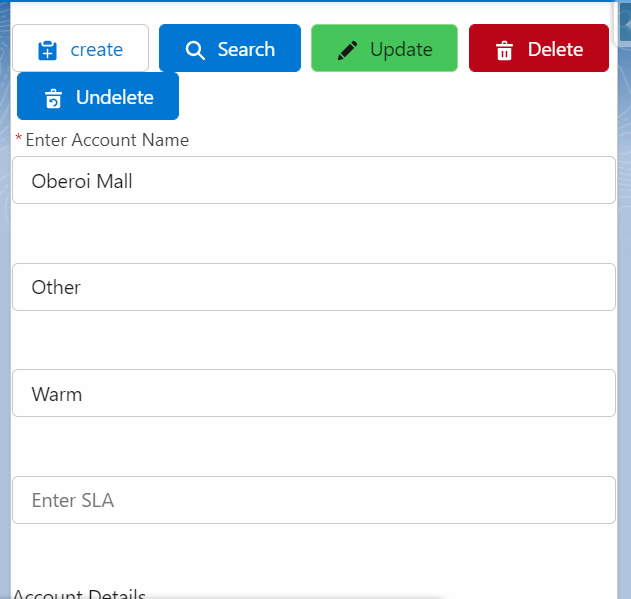
58. CRUD Operation - 25 June 2022

Today we will learn:

1] CRUD Operation

CRUD = Create, Read/Search, Update/Edit, Delete



Java Script Confirm:

let text = "Do you want to delete this record ? ";

if (confirm(text) == true) {

Business Logic 1;

} else {

Business Logic 2;

}

</script>

<template>

    <lightning-card title="CRUD Operation">

        <lightning-button variant="neutral" icon-name="utility:record\_create" label="Create" onclick={createNewRecordHandler}> </lightning-button> &nbsp;

        <lightning-button variant="brand" icon-name="utility:search" label="Search" onclick={searchRecordHandler}> </lightning-button> &nbsp;

        <lightning-button variant="success" icon-name="utility:edit\_form" label="Update" onclick={updateRecordHandler} disabled={disableUpdateButton}> </lightning-button> &nbsp;

        <lightning-button variant="destructive" icon-name="utility:delete" label="Delete" onclick={deleteRecordHandler} disabled={disableDeleteButton}> </lightning-button> <br/>

        <lightning-button variant="brand-outline" icon-name="utility:undelete" label="Undelete" onclick={undeleteRecordHandler}> </lightning-button> <br /><br/>

        <lightning-input type="text" data-formfield="accName" label="Enter Account Name" name="accName"> </lightning-input> <br />

        <template if:true={showInfoFlag} >

                <lightning-input type="text" data-formfield="accType" label="Enter Type" name="accType"> </lightning-input> <br />

                <lightning-input type="text" data-formfield="accSla" label="Enter SLA" name="accSLA"> </lightning-input> <br />

                <lightning-button variant="brand" icon-name="utility:save" label="Save" onclick={saveRecordHandler}> </lightning-button>

    </template>

    <template if:true={showSearchFields}>

        <lightning-input type="text" data-formfield="accType" label="Type" name="accType" value={objAccount.Type} disabled={disableSearchFlag}> </lightning-input> <br />

        <lightning-input type="text" data-formfield="accSLA" label="SLA" name="accsla" value={objAccount.SLA\_\_c} disabled={disableSearchFlag}> </lightning-input> <br />

    </template>

 <template if:true={showFindRecordFlag}>

    <lightning-button variant="brand" icon-name="utility:search" label="Find Record" onclick={findRecordHandler}> </lightning-button> <br/>

    </template>

    <template if:true={showUpdateButtonFlag}>

        <lightning-button variant="brand" icon-name="utility:edit" label="Update Record" onclick={updateSearchedRecordHandler}> </lightning-button> <br/>

        </template>

    </lightning-card>

</template>

import { LightningElement } from 'lwc';

import createNewAccount from  '@salesforce/apex/AccountProvider.createNewAccount';

import searchAccountRecord from '@salesforce/apex/AccountProvider.searchAccountRecord';

import updateAccountRecord from  '@salesforce/apex/AccountProvider.updateAccountRecord';

import deleteAccountRecord from '@salesforce/apex/AccountProvider.deleteAccountRecord';

import undeleteAccountRecord from  '@salesforce/apex/AccountProvider.undeleteAccountRecord';

export default class CrudCompo extends LightningElement {

   objAccount = {'sObjectType' : 'Account'};

   showInfoFlag = true

   disableSearchFlag = true

   showSearchFields = false;

   disableUpdateButton = true;

   showUpdateButtonFlag = false;

   showFindRecordFlag = false;

   disableDeleteButton = true;

    createNewRecordHandler(){

        this.showInfoFlag = true;

        this.showSearchFields = false;

    }

    searchRecordHandler(){

        this.showInfoFlag = false;

        this.showUpdateButtonFlag = false;

        this.showFindRecordFlag = true;

        this.showSearchFields = false;

        this.disableSearchFlag = true;

    }

    updateRecordHandler(){

        console.log("Update");

        this.disableSearchFlag = false;

        this.showInfoFlag = false;

        this.showUpdateButtonFlag = true;

        this.showFindRecordFlag = false;

        this.showSearchFields = true;

    }

    deleteRecordHandler(){

        let text = "Do you want to delete this record ? ";

        if (confirm(text) == true) {

            console.log("Delete");

            this.showSearchFields = false;

            deleteAccountRecord({objAcc : this.objAccount})

            .then((result) =>{

                console.log(JSON.stringify(result) );

                alert('Record Deleted...!!!');

                this.objAccount =null;

            })

            .catch((error)=>{

                console.log(JSON.stringify(error));

            });

        }

        else {

        }

    }

    undeleteRecordHandler(){

        this.objAccount.Name = this.template.querySelector('lightning-input[data-formfield="accName"]').value;

        undeleteAccountRecord({objAcc : this.objAccount})

        .then((result) =>{

            console.log(JSON.stringify(result) );

            alert(JSON.stringify(result));

            this.objAccount =null;

        })

        .catch((error)=>{

            console.log(JSON.stringify(error));

        });

    }

    saveRecordHandler(){

        this.objAccount.Name = this.template.querySelector('lightning-input[data-formfield="accName"]').value;

        this.objAccount.Type = this.template.querySelector('lightning-input[data-formfield="accType"]').value;

        this.objAccount.SLA\_\_c = this.template.querySelector('lightning-input[data-formfield="accSla"]').value;

        createNewAccount({objAcc : this.objAccount})

        .then((result) =>{

            console.log(JSON.stringify(result) );

            alert('Record Created...!!!');

        })

        .catch((error)=>{

            console.log(JSON.stringify(error));

        });

    }

    findRecordHandler(){

        this.objAccount.Name = this.template.querySelector('lightning-input[data-formfield="accName"]').value;

        searchAccountRecord({objAcc : this.objAccount})

        .then((result) =>{

            console.log(JSON.stringify(result) );

            this.objAccount = result;

            this.showSearchFields = true;

            this.disableUpdateButton = false;

            this.disableDeleteButton = false;

            alert('Record Found');

        })

        .catch((error)=>{

            console.log(JSON.stringify(error));

            this.showSearchFields = false;

            this.disableUpdateButton = true;

        });

    }

    updateSearchedRecordHandler(){

        debugger;

        this.objAccount.Name = this.template.querySelector('lightning-input[data-formfield="accName"]').value;

        this.objAccount.Type = this.template.querySelector('lightning-input[data-formfield="accType"]').value;

        this.objAccount.SLA\_\_c = this.template.querySelector('lightning-input[data-formfield="accSLA"]').value;

        updateAccountRecord({objAcc : this.objAccount})

        .then((result) =>{

            console.log(JSON.stringify(result) );

            this.objAccount = result;

            alert('Record Updated');

        })

        .catch((error)=>{

            console.log(JSON.stringify(error));

        });

    }

}

public with sharing class AccountProvider {

@AuraEnabled

public static string createAccount(String accName){

    try {

                if(!String.isBlank(accName)){

                Account objAcc = new Account(Name = accName);

                insert objAcc;

                return 'Account Record Created successfully...';

    }

            else{

                return 'Locha..!!!';

            }

    } catch (Exception e) {

        throw new AuraHandledException(e.getMessage());

    }

}

@AuraEnabled

public static string createNewAccount(Account objAcc){

    try {

        if(objAcc != null){

            Insert objAcc;

            return 'New Account Record Created Successfully...!!!';

        }

        else{

           return 'Locha, something went wrong...!!!';

        }

    } catch (Exception e) {

        throw new AuraHandledException(e.getMessage());

    }

}

@AuraEnabled

public static Account searchAccountRecord(Account objAcc){

    try {

        if(objAcc !=null){

            return [select Id, Name, Type, SLA\_\_c from Account where Name=: objAcc.Name];

        }

        else{

            return null;

        }

    } catch (Exception e) {

        throw new AuraHandledException(e.getMessage());

    }

}

@AuraEnabled

public static Account updateAccountRecord(Account objAcc){

    try {

        if(objAcc !=null){

            Database.update(objAcc, false);

            return [select Id, Name, Type, SLA\_\_c from Account where Name=: objAcc.Name];

        }

        else{

            return null;

        }

    } catch (Exception e) {

        throw new AuraHandledException(e.getMessage());

    }

}

@AuraEnabled

public static string deleteAccountRecord(Account objAcc){

    try {

        if(objAcc !=null){

            delete objAcc;

            return 'Record Deleted';

        }

        else{

            return 'Locha, something went wrong...!!!';

        }

    } catch (Exception e) {

        throw new AuraHandledException(e.getMessage());

    }

}

@AuraEnabled

public static string undeleteAccountRecord(Account objAcc){

    try {

        if(objAcc !=null){

            undelete [select Id from Account where Name=: objAcc.Name LIMIT 1 ALL ROWS];

            return 'Record Undeleted successfully...!!';

        }

        else{

            return 'Record not found in Trash';

        }

    } catch (Exception e) {

        throw new AuraHandledException(e.getMessage());

    }

}

}