LWC Mini Project: Product Management App

Project Overview

This mini project allows users to manage electronic products in Salesforce using LWC.
It includes adding, viewing, and deleting products from a custom object Electronic_Product__c with fields Name and Price__c.

1. Apex Controller - ProductController.cls

```
public with sharing class ProductController {
    @AuraEnabled(cacheable=true)
    public static List<Electronic_Product__c> getProducts() {
              return [SELECT Id, Name, Price__c FROM Electronic_Product__c ORDER BY
CreatedDate DESC];
    }

    @AuraEnabled
    public static Electronic_Product__c createProduct(String name, Decimal price) {
         Electronic_Product__c prod = new Electronic_Product__c(Name = name, Price__c =
price);
        insert prod;
        return prod;
    }

    @AuraEnabled
    public static void deleteProduct(Id productId) {
        delete [SELECT Id FROM Electronic_Product__c WHERE Id = :productId LIMIT 1];
    }
}
```

2. LWC Component: productManager.html

```
<template>
    <lightning-card title="Product Manager" icon-name="custom:custom63">
        <div class="slds-p-horizontal_medium">
                            <lightning-input label="Product Name" value={productName}
onchange={handleNameChange}></lightning-input>
                      <lightning-input label="Price" type="number" value={productPrice}
onchange={handlePriceChange}></lightning-input>
              <lightning-button label="Add Product" variant="brand" onclick={addProduct}
class="slds-m-top_small"></lightning-button>
        </div>
        <template if:true={products}>
                  <lightning-datatable key-field="Id" data={products} columns={columns}
hide-checkbox-column="true"></lightning-datatable>
        </template>
    </lightning-card>
</template>
```

3. productManager.js

```
import { LightningElement, track } from 'lwc';
import getProducts from '@salesforce/apex/ProductController.getProducts';
import createProduct from '@salesforce/apex/ProductController.createProduct';
import deleteProduct from '@salesforce/apex/ProductController.deleteProduct';
import { ShowToastEvent } from 'lightning/platformShowToastEvent';

export default class ProductManager extends LightningElement {
    @track productName = '';
    @track productPrice;
    @track products;

    columns = [
        { label: 'Product Name', fieldName: 'Name' },
        { label: 'Price', fieldName: 'Price__c', type: 'currency' },
        {
            type: 'button',
            label: 'Delete',
            typeAttributes: {
                label: 'Delete',
                name: 'delete',
                title: 'Delete',
                variant: 'destructive',
                iconName: 'utility:delete'
            }
        }
    ];

    connectedCallback() {
        this.fetchProducts();
    }

    fetchProducts() {
        getProducts().then(data => {
            this.products = data;
        }).catch(error => {
            this.showToast('Error', error.body.message, 'error');
        });
    }

    handleNameChange(event) {
        this.productName = event.target.value;
    }

    handlePriceChange(event) {
        this.productPrice = event.target.value;
    }
```

```
    addProduct() {
        if (!this.productName || !this.productPrice) {
               this.showToast('Validation Error', 'Please fill in both name and price',
'warning');
            return;
        }
           createProduct({ name: this.productName, price: parseFloat(this.productPrice)
}).then(() => {
            this.showToast('Success', 'Product added', 'success');
            this.productName = '';
            this.productPrice = '';
            this.fetchProducts();
        }).catch(error => {
            this.showToast('Error', error.body.message, 'error');
        });
    }

    handleRowAction(event) {
        const action = event.detail.action;
        const row = event.detail.row;

        if (action.name === 'delete') {
            deleteProduct({ productId: row.Id }).then(() => {
                this.showToast('Deleted', 'Product deleted', 'success');
                this.fetchProducts();
            }).catch(error => {
                this.showToast('Error', error.body.message, 'error');
            });
        }
    }

    showToast(title, message, variant) {
        this.dispatchEvent(new ShowToastEvent({ title, message, variant }));
    }
}
```

4. productManager.js-meta.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>60.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__AppPage</target>
        <target>lightning__RecordPage</target>
        <target>lightning__HomePage</target>
    </targets>
```

```
</LightningComponentBundle>
```