import { test, Locator, Page } from '@playwright/test'

export default class AccountingPage {

  readonly page: Page;

  public accounting: {

    accountingLink: Locator;

    reports: Locator;

    dailyTransaction: Locator;

    fiscalYear: Locator;

    load: Locator;

    settings: Locator;

    searchBar: Locator;

    activate: Locator;

    deactivate: Locator;

  };

  constructor(page: Page) {

    this.page = page;

    this.accounting = {

      accountingLink: page.getByRole('link', { name: 'Accounting ' }),

      reports: page.locator('//a[text()=" Reports "]'),

      dailyTransaction: page.locator('//i[text()="Daily Transaction"]'),

      fiscalYear: page.locator('//select[@class="ng-untouched ng-pristine ng-valid"]'),

      load: page.locator('//button[@type="button"]'),

      settings: page.locator('(//a[@href="#/Accounting/Settings"])[2]'),

      searchBar: page.locator('#quickFilterInput'),

      activate: page.locator('//a[text()="Activate"]').first(),

      deactivate: page.locator("//a[text()='Deactivate']"),

    };

  }

  /\*\*

   \* @Test4 Verifies the activation process of the ledger by navigating to the accounting settings, searching for a specific ledger,

   \* and activating it through a confirmation dialog.

   \*

   \* @param {Record<string, string>} data - Not used in this method but typically used to pass additional parameters if needed.

   \* @returns {Promise<void>} - Returns void; logs error if any step fails.

   \*

   \* Steps:

   \* 1. Navigate to the Accounting module.

   \* 2. Go to Settings and search for the "BANK A/C #" ledger.

   \* 3. Trigger the activation process, confirm the activation in the dialog, and finalize the action.

   \*/

  async verifyActivationLedger() {

    const counter = await this.page.locator("//a[text()='X']");

    if (counter) {

      await counter.click();

    }

    await this.accounting.accountingLink.click();

    await this.page.waitForTimeout(2000)

    await this.accounting.settings.click();

    await this.page.waitForTimeout(4000)

    await this.accounting.settings.click();

    const ActiveAccount = await this.page.locator('(//a[text()="Activate"]/../../../div)[1]').innerText();

    await this.accounting.searchBar.fill(ActiveAccount);

    await this.accounting.searchBar.click();

    await this.page.keyboard.press('Enter')

    this.page.once('dialog', async dialog => {

      await dialog.accept();

    })

    await this.accounting.activate.first().click();

  }

  /\*\*

   \* @Test5 Verifies the deactivation process of a ledger entry ("Sundry Debtors (Receivables)")

   \* in the Accounting module settings.

   \*

   \* Steps:

   \* 1. Navigates to the Accounting module and opens Settings.

   \* 2. Searches for a specific ledger using the search bar.

   \* 3. Handles the deactivation confirmation dialog.

   \* 4. Clicks on the deactivate button to trigger the action.

   \*

   \* @returns {Promise<void>} - This method performs UI actions and does not return a value.

   \*/

  async verifyDeactivationLedger() {

    const counter = await this.page.locator("//a[text()='X']");

    if (counter) {

      await counter.click();

    }

    await this.accounting.accountingLink.click();

    await this.page.waitForTimeout(2000)

    await this.accounting.settings.click();

    await this.page.waitForTimeout(4000)

    await this.accounting.settings.click();

    const DeActiveAccount = await this.page.locator('(//a[text()="Deactivate"]/../../../div)[1]').innerText();

    await this.accounting.searchBar.fill(DeActiveAccount);

    await this.accounting.searchBar.click();

    await this.page.keyboard.press('Enter')

    this.page.once('dialog', async dialog => {

      await dialog.accept();

    })

    await this.accounting.deactivate.first().click();

  }

}