***Day 4 – Amazon Test Case Assignment***

**Best Sellers Page Tests**

This tests validates various functionalities of the **Amazon Best Sellers** page using Playwright and a bestSellersPage page object with the help of fixtures. The tests are configured to run **in parallel** for efficiency.

test.describe.configure({ mode: 'parallel' });

**Test Coverage Overview:**

1. **Navigation:**
   * Open the Best Sellers page before each test.

test.describe('Best Sellers Page Tests', () => {

  test.beforeEach(async ({ bestSellersPage }) => {

    await bestSellersPage.navigate();

  });

1. **Search Functionality:**
   * Search for products like **Laptop**, **Mobile**, and **Camera**.
   * Verify search results are displayed.

[

    { searchKeyword: 'Laptop' },

    { searchKeyword: 'Mobile' },

    { searchKeyword: 'Camera' },

  ].forEach(({ searchKeyword }) => {

    test(`verify and search results for "${searchKeyword}"`, async ({ bestSellersPage }) => {

      await bestSellersPage.searchProduct(searchKeyword);

      await expect(bestSellersPage.products.first()).toBeVisible();

    });

  });

1. **Page Title Validation:**
   * Ensure the page title matches:  
     "Amazon.in Bestsellers: The most popular items on Amazon"

test('Verify Page Title', async ({ bestSellersPage }) => {

    await bestSellersPage.verifyPageTitleHasTitle("Amazon.in Bestsellers: The most popular items on Amazon");

  });

1. **Category Display:**
   * Check that categories are visible.
   * Specifically verify presence of **Electronics** and **Books** categories.

test('Verify Categories Are Displayed', async ({ bestSellersPage }) => {

    await bestSellersPage.verifyCategoriesDisplayed();

  });

1. **Header Check:**
   * Confirm the presence of the **"Bestsellers"** header.

async verifyHasHeader(headervalue: string) {

    await expect(this.header.first()).toBeVisible();

    const headerText = await this.header.first().textContent();

    expect(headerText).toContain(headervalue);

  }

1. **Product Listings:**
   * Verify that products are listed on the page.

async verifyProductsDisplayed() {

    await expect(this.bestproducts.first()).toBeVisible();

  }

1. **Product Navigation:**
   * Simulate clicking on a product.
   * Confirm that product details (like an "Add to Cart" button) are displayed.

test('Verify Clicking a Product Opens Product Page', async ({ bestSellersPage }) => {

    await bestSellersPage.userClicksAProduct();

    await expect(bestSellersPage.addToCartButton.first()).toBeVisible();

  });

1. **Category Navigation:**
   * Navigate to a different category (e.g., **Books**) and verify the header updates appropriately.

test('Verify Navigation to a Different Category', async ({ bestSellersPage }) => {

    await bestSellersPage.verifyCategoriesDisplayed();

    await bestSellersPage.clickCategory('Books');

    await expect(bestSellersPage.verifyCategoriesDisplayed).toBeTruthy();

    await bestSellersPage.verifyHasBooksHeader('Bestsellers in Books');

  });

1. **Sorting/Filtering Options:**
   * Search for "camera" and apply a sorting filter (e.g., **Price: Low to High**).

async applySortOption(optionValue: string) {

    console.log(this.sortDropdown);

    if (await this.sortDropdown.isVisible()) {

      await this.sortDropdown.selectOption(optionValue);

      const priceValue1 = await this.price.first().textContent();

      const price1 = parseFloat(priceValue1 ?? "0");

      const priceValue2 = await this.price.nth(1).textContent();

      const price2 = parseFloat(priceValue2 ?? "0");

      console.log(`Price of first product: ${price1}`);

      if (price1 < price2) {

        console.log("Products sorted successfully in ascending order.");

      }

    }

  }

1. **Pagination:**
   * Navigate to the next page within a category.

async goToNextPage() {

    if (await this.paginationNext.isVisible()) {

      await this.paginationNext.click();

      await expect(this.page).toHaveURL(  /pg=2/);

    }

  }

1. **Add to Cart Flow:**
   * Add a product to the cart directly from the Best Sellers page after navigating to the product details page.

async addFirstProductToCart() {

    const addToCartButton = this.page.locator('#add-to-cart-button');

    if (await addToCartButton.isVisible()) {

      await addToCartButton.first().click();

      await expect(this.cartCount).not.toHaveText('0');

    }

  }

1. **Verify user can navigate each department**

* To ensure the user can navigate through every department listed on the Best Sellers page.
* After navigation, verify that each click leads to the correct page.

async validateEachDepartmentNavigateCorrectPage() {

    const departmentCount = await this.categories.count();

    console.log('Number of departments:', departmentCount);

    for (let i = 0; i < departmentCount; i++) {

      await this.categories.nth(i).click();

      await this.page.waitForLoadState();

      await expect(await this.page.locator('#zg\_banner\_text')).toBeVisible();

      await this.page.goBack();

    }

  }