***Day 5 – Snapdeal Assignment***

1. **Created a snapdeal.spec.ts for all the test cases to be automated via playwright typescript**

test.only('Verify End to End flow', async ({homePage, productsPage}) => {

        await homePage.verifyUserIsOnSnapdealHomePage();

        await homePage.userSearchForAProduct("Shoes for Women");

        await productsPage.verifyUserIsOnSnapdealProductsPage();

        const productDetailsTab = await productsPage.userSelectProduct();

        await productDetailsTab.waitForLoadState();

        const productDetailsPage = new ProductDetailsPage(productDetailsTab);

        await productDetailsPage.verifyUserIsOnSnapdealProductDetailsPage();

        await productDetailsPage.userAddToCart();

        const cartQuantity = await productDetailsPage.verifyCartQuantity();

        await expect(cartQuantity).toBe(1);

        await productDetailsPage.userClickOnCartIcon();

    })

1. **Created a snapdeal.fixture.ts for the fixtures to be implemented**

type SnapdealFixtures = {

    homePage: HomePage;

    productsPage: ProductsPage;

    productDetailsPage: ProductDetailsPage;

}

1. **Created Page Objects for the interaction with the UI elements in a structured and well efficient manner**

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

import { expect } from './Snapdeal.fixture';

export class HomePage {

    private readonly searchInput: Locator;

    constructor(public readonly page: Page) {

        this.searchInput = page.locator("#inputValEnter");

    }

    async navigate() {

        await this.page.goto("/");

    }

    async verifyUserIsOnSnapdealHomePage() {

        await expect(this.page).toHaveTitle("Shop Online for Men, Women & Kids Clothing, Shoes, Home Decor Items");

    }

    async userSearchForAProduct(productName : string) {

        await this.searchInput.isVisible();

        await this.searchInput.fill(productName);

        await this.searchInput.press("Enter");

        await this.searchInput.press("Enter");

    }

}

1. **Implement switching to another tab in a browser context using Promises and switch backing to the original page as well**

export async function switchToNewTab(context: BrowserContext, action: () => Promise<void>): Promise<Page> {

    const [newPage] = await Promise.all([

        context.waitForEvent('page'),

        action()

    ]);

    await newPage.waitForLoadState();

    return newPage;

}

export async function switchBackToPage(page: Page) {

    await page.bringToFront();

}