tripleten

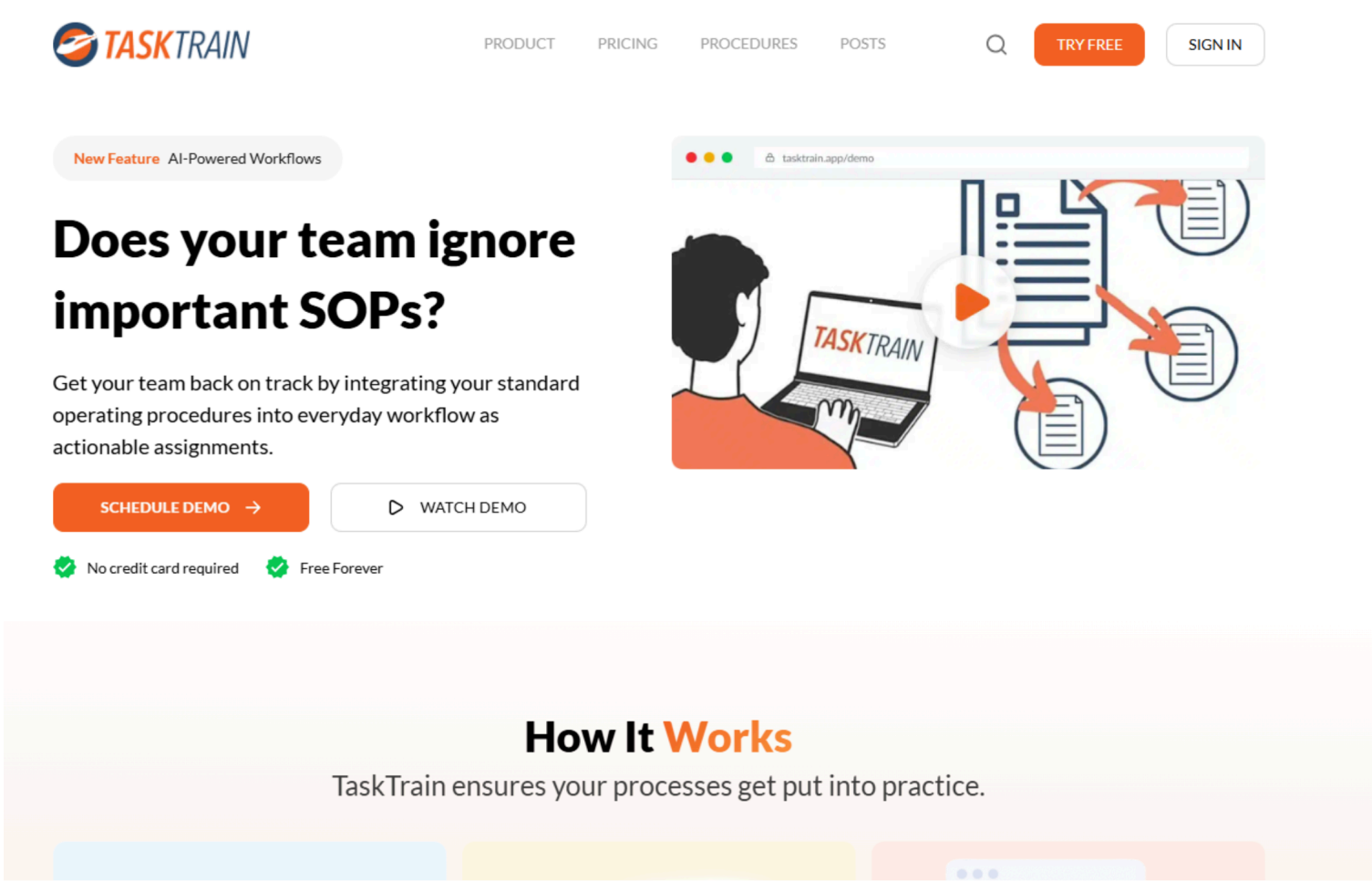
Manual and automated testing for BPM platform

Partner: TaskTrain

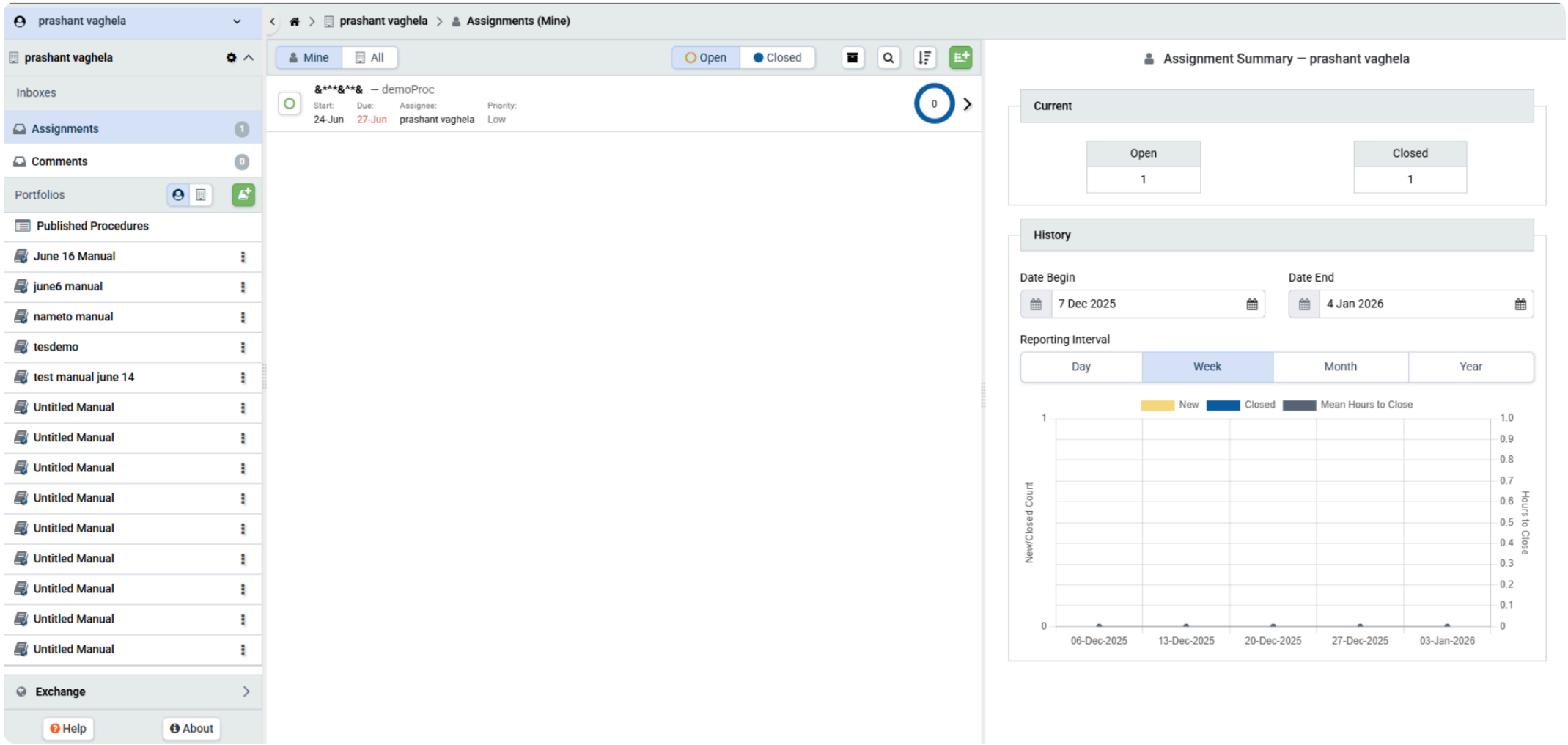
Website: https://www.tasktrain.app/

Project description: Students conducted exploratory testing, reported bugs, and built manual and automated test cases for a Business Process Management platform. Students also improved test reliability by refactoring automation and setting up CI/CD pipelines.

Screenshot of the TaskTrain home page.



Screenshot of a landing page of the TaskTrain application.



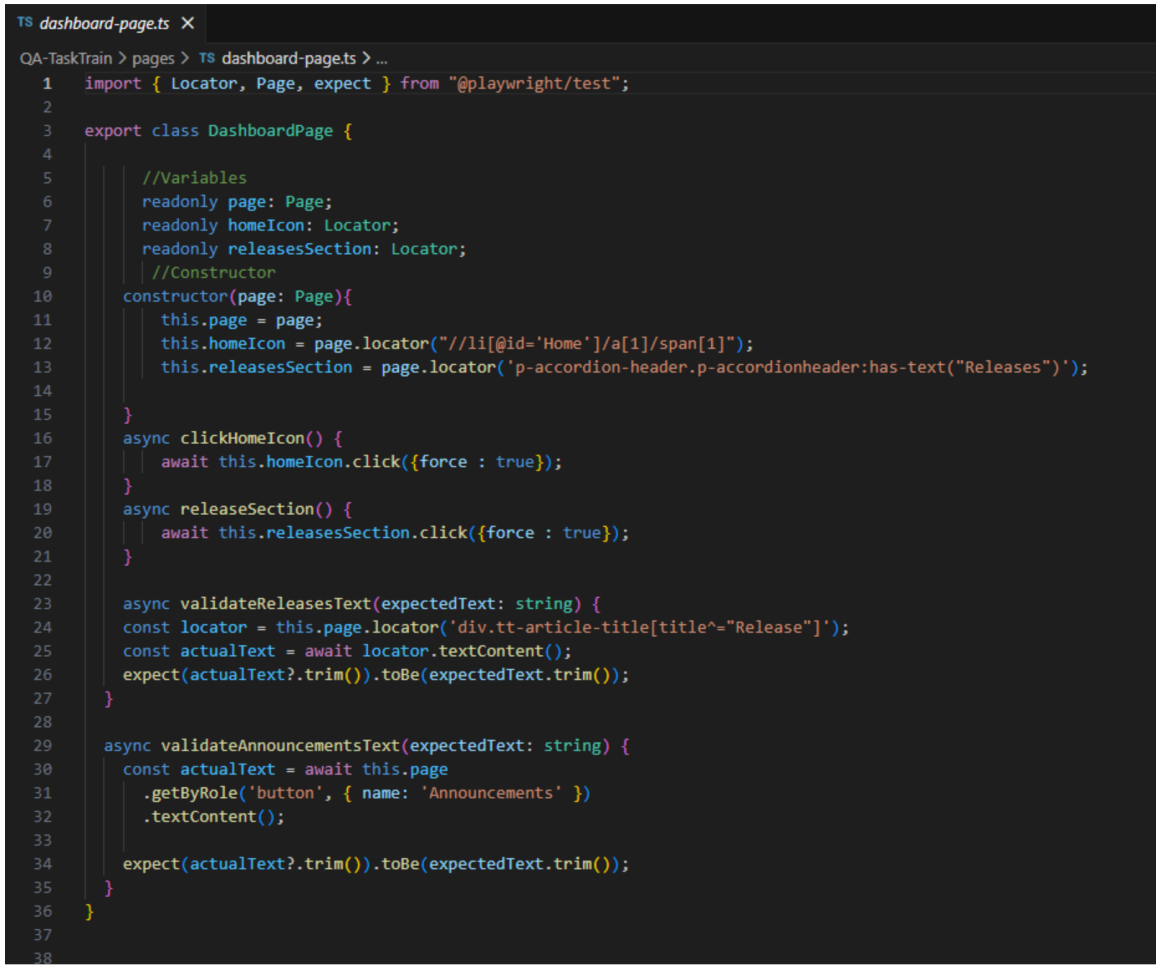
During the externship, engineers used tags to categorize tests.

```
// TESTS BEGIN
test.describe("Feature: Account Update Display Name", () => {
  test.describe("Scenario: Update Display Name", () => {
    test("Change display name to Arthur Dent @regression", async ({ page, navOrganization, organizationSettings }) => {
      // Click on Organization Settings button
      await navOrganization.manageOrganizationButton.click({ force: true });
      // Change Display Name to 'Arthur Dent'
      await organizationSettings.updateDisplayName("Arthur Dent");
      // Assert Display Name has been changed to Arthur Dent
      await expect(organizationSettings.displayNameInput).toHaveValue("Arthur Dent");
    });
  });
});
```

During the externship, engineers used tags to categorize tests.

```
//smoke tests
test.describe('Feature: Assignment', () => {
  test('Create Assignment @smoke @regression', async ({ page, navOrganization }, testInfo) => {
    testInfo.annotations.push({ type: 'tag', description: 'smoke'});

    await navOrganization.getProcedureName.click();
    await navOrganization.actionMenu.click();
    await navOrganization.procedurePublishButton.click();

    await navOrganization.assignButton.click();
    await navOrganization.assignmentName.fill('assignName');
    await navOrganization.procedureAssignment();
    await navOrganization.assignmentAssignButton.click();

    await expect(page.getByText('Procedure assigned')).toBeVisible();
  });

});
```

Engineers learned the value of abstracting reused code into Page Object Model classes or other data classes used to manipulate data.

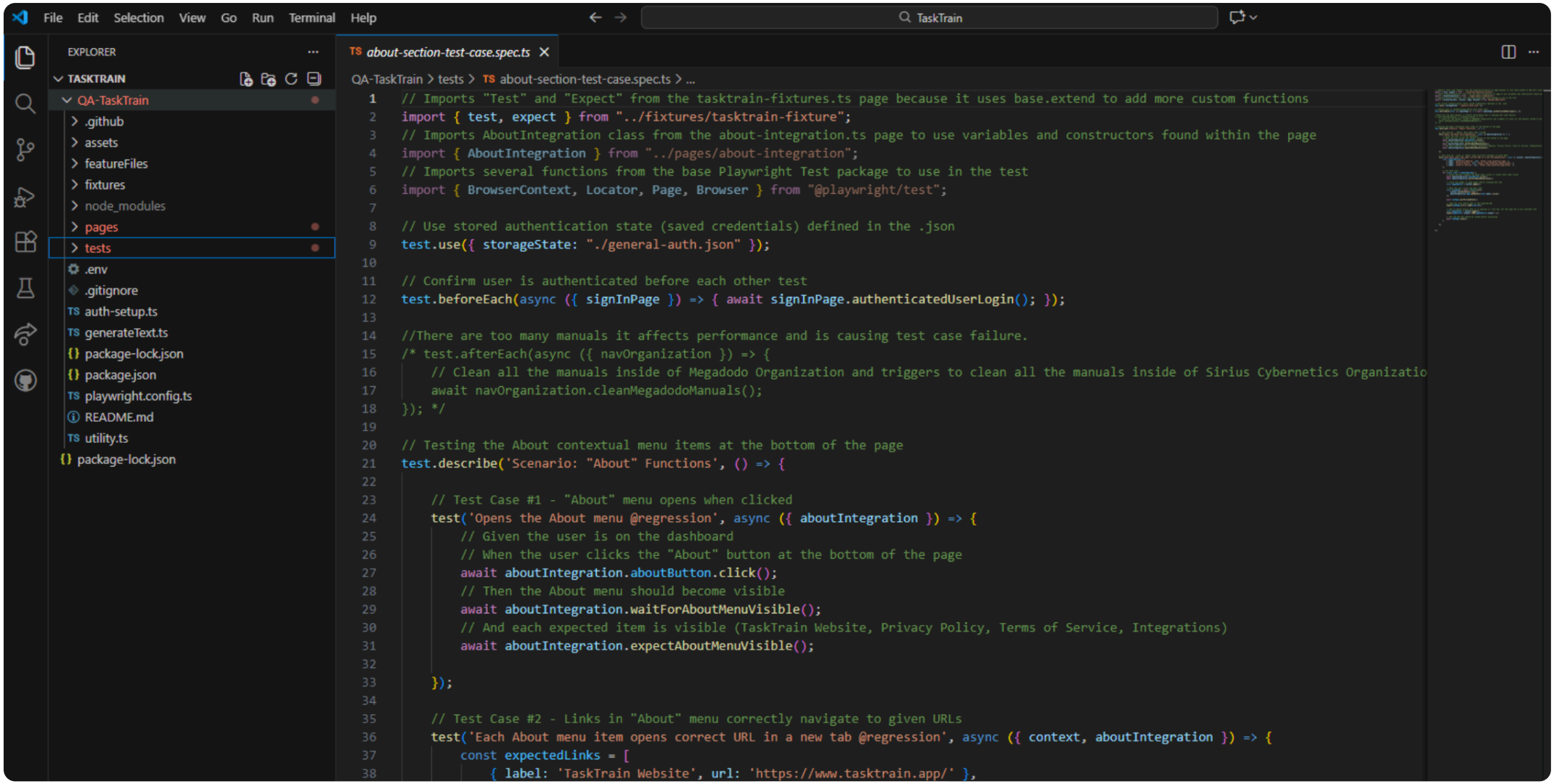


TS dashboard-page.ts

QA-TaskTrain > pages > TS dashboard-page.ts > ...

```ts
import { Locator, Page, expect } from "@playwright/test";

export class DashboardPage {

    //Variables
    readonly page: Page;
    readonly homeIcon: Locator;
    readonly releasesSection: Locator;
    //Constructor
  constructor(page: Page){
    this.page = page;
    this.homeIcon = page.locator("//li[@id='Home']/a[1]/span[1]");
    this.releasesSection = page.locator('p-accordion-header.p-accordionheader:has-text("Releases")');

  }
  async clickHomeIcon() {
    await this.homeIcon.click({force : true});
  }
  async releaseSection() {
    await this.releasesSection.click({force : true});
  }

  async validateReleasesText(expectedText: string) {
  const locator = this.page.locator('div.tt-article-title[title^="Release"]');
  const actualText = await locator.textContent();
  expect(actualText?.trim()).toBe(expectedText.trim());
 }

 async validateAnnouncementsText(expectedText: string) {
  const actualText = await this.page
    .getByRole('button', { name: 'Announcements' })
    .textContent();

  expect(actualText?.trim()).toBe(expectedText.trim());
 }
}

```

TripleTen engineers used Visual Studio Code to write their automated tests and Page Object Model files.



Configured CI/CD pipeline to execute Smoke tests on every Pull Request.