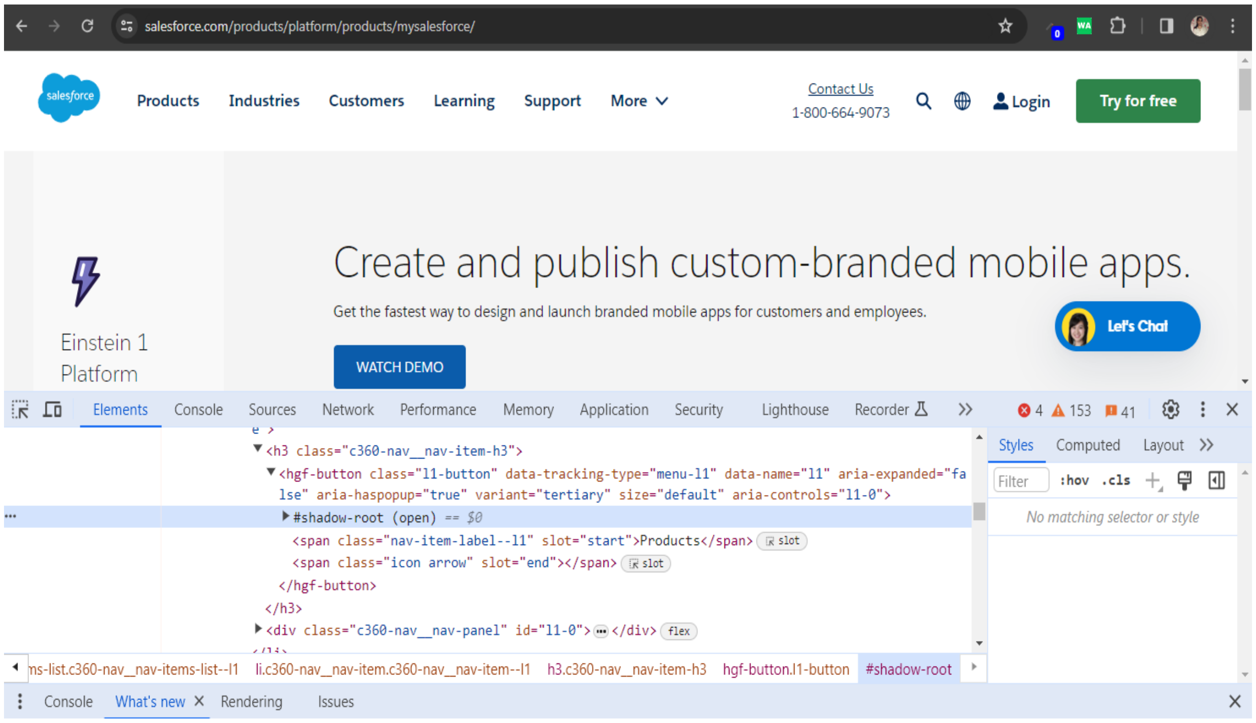
# HOW TO HANDLE SHADOM DOM IN PLAYWRIGHT?

## *What is Shadow DOM?*

Shadow DOM allows developers to create isolated components within a webpage, ensuring styles and scripts are contained without affecting the rest of the page. It acts as a secret compartment for neatly organized code, invisible and non-interfering with other elements.

## *Why Shadow DOM Matters?*

- Isolation: It ensures that the styles and scripts of a component do not leak out or get influenced by the main page, making components portable and reusable across different projects.
- Encapsulation: By keeping a component's internals hidden, Shadow DOM promotes better code organization and maintenance, facilitating a modular approach to web development.



```
▼<h3 class="c360-nav__nav-item-h3">
  ▼<hgf-button class="l1-button" data-tracking-type="menu-l1" data-name="l1" aria-expanded="fa
   lse" aria-haspopup="true" variant="tertiary" size="default" aria-controls="l1-0">
    ▶#shadow-root (open) == $0
     <span class="nav-item-label--l1" slot="start">Products</span> slot
     <span class="icon arrow" slot="end"></span> slot
   </hgf-button>
 </h3>
▶<div class="c360-nav__nav-panel" id="l1-0">…</div> flex
```

```
ms-list.c360-nav__nav-items-list--l1   li.c360-nav__nav-item.c360-nav__nav-item--l1   h3.c360-nav__nav-item-h3   hgf-button.l1-button   #shadow-root
```

## *Challenges with Shadow DOM*

Automating Shadow DOM components poses unique challenges, primarily because traditional automation tools and scripts might not directly interact with the encapsulated elements. The main hurdles include:

1. *Visibility:* Elements within a Shadow DOM are not readily accessible to automation scripts that operate on the global DOM.

2. *Tool Compatibility:* Many automation frameworks and tools have limited support for navigating through Shadow DOM boundaries, complicating direct element interaction and manipulation.

Playwright simplifies working with Shadow DOM. All locators in Playwright by default work with elements in Shadow DOM.

*Automatic Traversal:* Playwright's CSS and text locators pierce the Shadow DOM by default.

You can access shadow elements just like other DOM elements without writing explicit JavaScript code.

For example, page.locator("[text='Products']") directly accesses a Product link within the Shadow DOM. You can locate in the same way as if the shadow root was not present at all.

The exceptions are:

- Locating by XPath does not pierce shadow roots.
- Closed-mode shadow roots are not supported.