What is Playwright?

- Browser automation tool
  - Chromium, Firefox and WebKit
- Node, Python, Java, .NET
- https://playwright.dev/python/docs/api/class-playwright
- https://playwright.dev/docs/intro
- https://github.com/microsoft/playwright ### Use cases
- Automated testing
  - End-to-end and UI testing
    - https://playwright.dev/python/docs/writing-tests
  - Performance testing
    - https://www.artillery.io/docs/guides/guides/playwright
- Web scraping ### Compared to Selenium
- Playwright
  - Out-of-the-box, has
    - Multi-browser support
    - Debugging tools
    - Code gen tool
  - Has simpler installation and setup
  - Is generally faster
- Selenium
  - Has an older and larger community and more comprehensive docs
  - Has broader language support ### Installing playwright:

1. `pip install playwright`
   - Installs the sync and async APIs
2. `playwright install`
   - Installs web browsers
     - Chromium, Webpack, Firefox ##### FYI: Installing playwright on DERMS required us to install an additional linux package (libasound2) on our base image. ### Writing tests We used LiveServerTestCase with the playwright_sync library and Playwright's built-in assertions (expect)

- https://docs.djangoproject.com/en/4.1/topics/testing/tools/#liveservertestcase
- https://playwright.dev/python/docs/api/class-playwright

```
from django.test import LiveServerTestCase
from playwright.sync_api import sync_playwright, expect


class BasePlaywrightTestCase(LiveServerTestCase):
  ...
```

Set up

```
os.environ["DJANGO_ALLOW_ASYNC_UNSAFE"] = "true"
```

```
playwright = sync_playwright().start()
browser = playwright.chromium.launch()
page = browser.new_page()
page.goto(...)
```

Clean up

```
page.close()
browser.close()
cls.playwright.stop()
os.environ["DJANGO_ALLOW_ASYNC_UNSAFE"] = "false"
```

Manually https://playwright.dev/python/docs/intro #### Playwright test generator - `playwright codegen <your url>` - `page.pause()` - Beware of - Ambiguous locators - Unnecessary code - Clicking into input fields first

Running tests

Headed vs headless

```
self.browser = self.playwright.chromium.launch(headless=True/False)
```

- Headed mode opens up a live browser window
  - Debugging
  - Code gen
  - Slow
  - Requires additional software*
- Headed mode does not open a live browser window
  - Faster
  - Pipeline friendly
- Configure the mode via an environment variable #### Static files
- For DERMS, a front-end build was needed in the pipeline

```
docker-compose run front_end npm run build
docker-compose run django ./manage.py test
```

- And `collectstatic` needed to be called within the test cases

```
from django.core.management import call_command

setUpClass(cls):
    ...
    call_command("collectstatic", interactive=False)
```

*

The following instructions assume you are using MacOS. ##### 1. Install XQuartz. You only need to do this once. 1. Install XQuartz locally by running:

```
brew install --cask xquartz
```

2. Open XQuartz, go to Preferences -> Security, and check “Allow connections from network clients”.
3. Restart your machine.
4. Open Docker and go to Settings -> Resources -> File sharing, and give access to `/tmp/.X11-unix`. If `/tmp` already has access, this is sufficient. ##### 2. Before running the test cases in headed mode, start XQuartz by running:

```
xhost +localhost
```