**Introduction**: Creating an efficient data driven playwright test to login to Asana iterate through JSON test data and verify the expected data is present in the UI.

**Implementation Details**: The script must first login to Asana using the given username and password, this is being handled in the beforeEach section that is executed before each test is run. The test cases are represented as an element in the JSON file and include the following attributes: name, project, task, column, tags[]. The tags attribute is an array of tags. The code utilizes loops to iterate through each JSON element and verify the attributes associated with that element. The array of tags associated with that element are also verified in an inner for loop. Adding a new test case would be as simple as adding a new row to the JSON file.

GitHub Repository: <https://github.com/coanderson/playwrightloop/>

**Challenges and Solutions**:

* Login – the login is a two-step process and can be slow to complete (~ 12 seconds before fully loaded). In order to make sure the page was loaded I used a locator.waitFor to find if the User Settings menu was present to verify the login was a success
* Columns – the columns could be easier to locate in the DOM. I used a selector to find the columns and then used filtering to find items under the correct column
* Project names - The projects were documented with a period at the end. These were removed in order to properly navigate the correct project.

**Results**:

* All test cases failed due to the documented strings in the technical evaluation document. The test cases will pass if the project names and the tags are corrected to the actual values found.
  + Test 1 –
    - Navigation fails - The project is documented as "Cross-functional project plan, Project.", I corrected the project names so that navigation could be verified.
    - The tag 'On track.' on screen is 'On track'
  + Test 2 –
    - Navigation fails - The project is documented as "Cross-functional project plan, Project.", I corrected the project names so that navigation could be verified.
    - The tag 'At risk.' on screen is ‘At risk’
  + Test 3 –
    - Navigation fails - The project is documented as "Cross-functional project plan, Project.", I corrected the project names so that navigation could be verified.
    - The tag 'Off track.' on screen is 'Off track'
  + Test 4 –
    - Navigation fails - The project is documented as "Work Requests.", I corrected the project names so that navigation could be verified.
    - The tags "Medium priority," "Low effort," "New hardware," and "Not Started."' on screen are "Medium priority" "Low effort" "New hardware" and "Not Started"
  + Test 5 –
    - Navigation fails - The project is documented as "Work Requests.", I corrected the project names so that navigation could be verified.
    - The tag 'Low effort,' on screen is 'Low effort'
  + Test 6 –
    - Navigation fails - The project is documented as "Work Requests.", I corrected the project names so that navigation could be verified.
    - The tags "Low effort," "New hardware," "High Priority," and "Done."' on screen are "Low effort," "New hardware," "High Priority," and "Done."

**Recommendations**:

* Login – a loading image that was shown until loading is complete would make it easier to know when the page could be used by waiting for this element to not be found
* Task Preview – the column information is not found on the task preview page
* Login once – instead of logging out and back in for each JSON element, logging in only one time and running all tests would be faster
* Expanded testing possibilities
  + Login
    - Different users – Tests could be run for additional users. The JSON elements could include the username and password to use for each test case
    - Invalid user – check error
    - Invalid password – check error
    - Account locked – check error
  + Verify task preview pane matches the expected values