Basics

1. How do you deal with flaky tests?

   A. Preventing flaky tests at an early stage.
      - Identifying the red flags at early stages is best.
      - For example:
        - test covers lot of logic
        - use of fixed wait time
        - test asserts randomly generated data
        - reliance on test order etc.

   B. Run test regularly to identify flaky tests based on reports generated after every run.

   C. Isolate the flaky tests.
      - Isolate flaky tests & focus on them separately to avoid any delays

   D. Optimize the test structure.
      - Divide the flaky test into smaller sections to avoid validating a big flow of end to end logic. Keep the flaky tests as simple as possible.

   E. Use dynamic waiting to avoid timeout
      - Using Cypress's network feature we are able to wait exactly as long as our application needs making the tests more stable & less prone to flakiness.

   F. Debugging flaky tests
      - Putting the flaky tests in loop. Running them multiple times to make sure they pass every time. Inspect the output using screenshot & video recording features.

2. Let's suppose test pipeline is taking about 1 hour to finish, what would you do to decrease the time of it?
   - Identify the test cases which are slow & have bigger impact on the execution time. Work on them separately to understand the slowness.
   - Make smart use of waits
   - Optimize SQL queries used in scripts.
   - Capture screenshots & videos to understand what is slowing down the execution.
   - Breakup test cases having large flows into smaller ones.
   - Go through the suit again & focus only on critical scenarios. Remove all other scenarios from suit.
   - Running tests in parallel on multiple remote machines

3. Imagine you have the possibility to ask software engineers to develop tools for you that will increase your productivity as full-stack QA, please describe to them your requirements
    a. In order to develop any tool first I need to explain my current task and how it is time consuming. After identifying that this task can be automated if we create a tool. This tool should save x number of hours for certain time period.
    b. Requirements will focus on user inputs, steps to be automated; auto triggering of tool functionality to start it at any specified time, end report which should include screenshots of failed scenarios & a trigger email to inform everyone that tool did run at specified time with result report as an attachment. End goal must be achieved where time is saved & productivity is improved.

    For example: Assuming that tool is being developed to identify broken link.
        i. As a QA, if I have to click on every single hyperlink on every webpage, it will surely be time consuming.
        ii. Requirements will include:
            1) Focus on creating a tool which can check all hyperlinks on a page whose URL will be provided by QA.
            2) An option to validate hyperlinks on multiple webpages together in different environments.
            3) An option to schedule execution
            4) Any broken link should be sent to QA via triggering a report.
            5) Report should have screenshots of broken links
            6) QA will re-validate only the broken link & report the issue.

Test Case Challenge

Purpose of testing

The purpose of testing is to make sure that user can login without any errors on different platforms.

Test Strategy

Testing will be conducted using waterfall methodology. QA will focus on end to end login flow. User login should work on Android, iOS & web platform. User can pick any platform after which drivers or bundles or packages will be called to trigger browser or native apps on specific platforms.

The login feature will work with different locators to fill details in email field, password field & to click on log in button. Page title will be validated to make sure login is successful.

I will automate the login feature so that it can be implemented based on user’s choice of platform & environment. This will help to test login feature efficiently.

Test Scenarios

- User is able to log into web, android and iOS platforms without any errors.
- Native application on the mobile is able to load login webpage.
- Email address and password fields are displaying error messages when invalid data is entered.
- Email address and password fields are displaying error messages when no data is entered in any of the fields.
- Verify the home page of application once user log in is successful.

Roles

Sagar is the responsible QA for this change.

Out of Scope

Nothing is kept out of scope for this change.

Test Automation Challenge

Resolution for Test Case 3 with Selenium:

- QA will open "https://github.com/appwrite/appwrite/pulls" hyperlink using driver.get() method.
- Successful opening of URL will be validated using page title.
- Create a method (method name =Updatedata) using FileWriter class, BufferWriter class & Write method to write data into CSV file.
- Find out a common attribute (common attribute is [data-hovercard-type="pull_request"]) for all pull requests and get the pull request count. Pull request data is spread over three pages. Make sure to navigate to each page.
- QA will gather PR name, created date & author from pull request table on the web page from the common attribute using locators.
- Store captured values from each <div> tab for PR name, created date & author into three variables.
- Using iterator every time you save values in three variables call Updatedata() method to write recently saved data into CSV file.
- Iterator will run as per pull request count and enter all the data in CSV file.
- QA will open the CSV file in plaintext format to view the PR name, created date & author.