PlanIt Questions and Answers

Q1: What other possible scenario’s would you suggest for testing the Jupiter Toys application?

Answer: below scenarios are missing:

1. **Login functionality:** several tests can be added for this functionality such as positive login case, negative can, cancel login, data validations etc.
2. **Home page:**  go to Shop page from “Start Shopping” button is missing, UI and content tests are also missing, verification of hyperlinks on menu, logo etc
3. **Shop page:** pagination if number of products are more, look and feel, image size, font etc.
4. **Contact page:** full form submission, email format validation, number format validation, top alert message validation ( in my test cases, I have added this verification), message length limitations, invalid input validations, sql injections, Cross Site Request Forgery tests, back button functionality, etc
5. **Cart Page:** checkout functionality, Empty cart, back button after empty cart, remove item from cart, change the quantity(positive and negative cases), data and security validation on Quantity textbox, links on top sentence verification, functional scenarios based on quantity modification and verification of sub total and grand total etc.
6. **Checkout:**  delivery details form validation test cases, payment details verification and validations on card numbers, submit validation, order processing, verification of order number from backend etc.

Q2: Jupiter Toys is expected to grow and expand its offering into books, tech, and modern art. We are expecting the of tests will grow to a very large number.

1. What approaches could you used to reduce overall execution time?

Answer: creation of reusable components will be the key, create commonly used reusable component which can work on variety of products irrespective of different offering. Parallel execution, parallel distributed execution across multiple machines can reduce the overall time. Apart from this writing optimized code for execution will be the key such as give appropriate time for waits, different conditions etc.

1. How will your framework cater for this?

Answer: Current provided framework supports many things to reduce the time, some reusable components are created in a such a way that, many different test cases can use them, and it worked seamlessly. This framework can be easily scalable to accommodate not only UI but web services, database, file-based automation. We can add selenium grid to support parallel or distributed execution. Even we can write one (this I have done in current company). Using testing xml also we can execute the tests in parallel which will reduce the execution time.

Q3: Describe when to use a BDD approach to automation and when NOT to use BDD

Answer: for BDD, many people have different opinions, I saw many organizations just go with buzz word BDD, they think that BDD is best and can easily automate the test etc.

For me, As Longform says Behaviour Driven Development (BDD), mainly should be used if Software development follows BDD and not only Test Automation. We also need to think about nature of application, technology and most important complexity of test cases, application features and workflow. For overly complex application BDD is not recommended. We can use BDD for small to medium size project.

BDD is mainly to bridge the gap between System Analyst/ Product analyst and Development team ( DEV and QA), for Software development BDD, feature files are written by System Analyst/ Product analyst, we cannot take that feature file and automation as we can create many tests from that single feature file, System Analyst/ Product analyst usually do not write very low level feature file which directly can be consumed by the automation team. Ultimately, automation engineer or manual team member has to write those feature files which creates an overhead for team to write test cases as well as feature file.

In terms of framework, we write one more file on top. Example:

BDD files for similar automation framework:

1. Feature file
2. Step definition file
3. Business component
4. Page factory

In normal framework:

1. Test class file
2. Business component
3. Page factory

For complex application, I will recommend generating more readable results of test script execution which help everyone to read the same and understand.

Lot of things can be talk about BDD.