Questions

1. What other possible scenarios would you suggest for testing the Jupiter Toys application?

1. Test input fields of Contact page, for XSS , SQL injection, LDAP injection vulnerabilities.
2. Test accessibility standards are covered for the elements in the pages.
3. Test in different browsers, platforms (premium, tablets, mobile) and different screen sizes need to ensure the functions are working fine.
4. If there is network failure, better to see how the functions are working and what sort of error messages are given.
5. When products are at the cart and before click check out, if prices and available quantities are changed in the system, need to see how the system behaves when the customer clicks the check out button.
6. Using blackbox testing techniques (boundary value analysis, equivalence partitioning, decision tables), we can decide input combinations for test cases.
7. When the user's culture is changed, need to verify the text contents of the web site is changed.
8. We can check the page load times having different number of products in the page to increase performance if possible.

2. 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?

1. Incorporate implicit wait and fluent wait mechanisms to reduce execution time.
2. Select only most frequently used web browsers to execute automation tests to reduce time.
3. Parallel execution of test cases, which will reduce time.
4. Prioritize test cases and execute high priority ones.
5. Without using sleep statements can reduce execution time.

2. How will your framework cater for this?

1. I have incorporated implicit wait, fluent wait mechanisms.
2. I have designed page classes in which can be extended easily for new tests to be added.
3. I have created WebDriver extension methods, which can be reused by new tests to be added.
4. Sleep statements are not used within test cases.
5. Code is structured to test based on different languages. But resource files for different languages are yet to be added to the solution.
6. Different browser types, platform types (premium, tablet, mobile) have been incorporated into the solution. In CI/CD server, we can give browser and platform type for the project to build and those settings are used to execute tests.

3. Describe when to use a BDD approach to automation and when NOT to use BDD

When to use BDD:

1. With BDD for complex systems with complex requirements it’s easy to understand tests mapped with requirements. This will make changing tests easier when requirements are changed.
2. When many stakeholders contribute for requirements and test cases, having BDD would help even non-technical people to understand test cases, in which all can contribute to setup tests.

When not to use BDD:

1. When business analysts are not collaborating with developers, testers, they may write scenarios with huge content and try to cover all possibilities. In these situations, BDD for automation is not suitable.
2. If the client is not frequently available to clear ambiguities it’s not good to use BDD for automation.