Task 1

* Where would I start? What would be your first steps?

* As a process, the following events should be done:

1. Planning to review and understand the requirements
2. Suppose Backend team implemented the APIs

* During the FE development activity, we may:

1. Review user stories and do ambiguity analysis
2. Review APIs documentation (example: swagger) or any reference to find the input of each API (headers, params, authentication way, form data, body, etc,… ) and the expected response (status code or response data in case of GET)
3. If It’s applicable to test the APIs back-to-back, we should do that to reduce the cost of fixing the APIs issues before any integration
4. After fixing APIs issues "if exist", start testing the integration between the FE and BE through the application UI components

* Regarding process would I establish around testing new functionality and How would I want the features to be tested?

* Suppose we have the new features documented in form of user stories

* Testing new features Process:
  + Suppose: Dev team will deliver the application URL and up and running testing environment then,

1. We should design the test plan to identify the objective, new scop, time frame
2. Preparing test data
3. Test cases should be created and linked to every new feature's user story
4. Review the designed testcases to check the coverage "filling the traceability matrix"
5. Prioritize the testcases and execute
6. Bug fixing according it’s priority and severity
7. Regression test

* If I would do test automation, which techniques or best practices would you use?

* I will answer this point according my automation experience

First of all, we should analyze and determine if we need to automate the UI or APIs or both of them.

* The following points "conditions" is mandated to stablish UI test automation:

1. We should calculate the ROI "return on investment" of automating our tests, if it's OK,
2. We should measure the team capabilities to kick-off the automation activity, if it's Ok,
3. We should determine the suitable tool which will be used based on the team capabilities
4. We should determine if the requirements are being frequently changed
5. We should determine if we have stable application
6. We should engage to the front end team to get good design (ids, classes, …. )
7. We should determine if we have stable environment to execute our automation
8. We should classify the **high frequently used** testcases "regression suite"
9. We should determine the **tests that may cause human errors**
10. We should follow techniques which assist us to create automation scripts **Resistant to Changes in the UI** example: POM design pattern in case of using OOP for example.

* The APIs automation will be more useful in case of the violation of the above conditions. As if we could get successful UI automation testing, we can cover a lot of APIs in little UI test scenarios

Task 2

* Selected API:
* [Search for Breeds by name - /breeds - API Reference - Home - TheCatApi // Developer Experience](https://docs.thecatapi.com/api-reference/breeds/breeds-search)
* API testcases:

|  |  |  |  |
| --- | --- | --- | --- |
| TC ID | Description | Expected Result | Automated |
| 1 | search with **valid** full name: query parameter: ?q=”Siberian” | Status code: 200  Response data: Should be valid and contains **name** attribute = [Siberian] | Yes |
| 2 | search with **sub-valid** full name: query parameter? q=” Siber” | Status code: 200  Response data: Should be valid and contains **name** attribute = [Siberian] and all names contains **Siber** |  |
| 3 | Search with invalid name | Empty response |  |
| 4 | search with **valid** sub name: query parameter ?q=”Norwegian Fo” | Status code: 200  Response data: Should be valid and contains **name** attribute = [Siberian] and all names contains **Norwegian Fo** |  |
| 5 | search with special characters like \* | Empty list | Yes |
| 6 | search without enter any value in the query param | Empty list |  |

Response Scheme” should be validated”:

[ {

"weight": {

"imperial": "8 - 16",

"metric": "4 - 7"

}, "id": "sibe",

"name": "Siberian",

"cfa\_url": "http://cfa.org/Breeds/BreedsSthruT/Siberian.aspx",

"vetstreet\_url": "http://www.vetstreet.com/cats/siberian",

"vcahospitals\_url": "https://vcahospitals.com/know-your-pet/cat-breeds/siberian",

"temperament": "Curious, Intelligent, Loyal, Sweet, Agile, Playful, Affectionate",

"origin": "Russia", "country\_codes": "RU", "country\_code": "RU",

"description": "The Siberians dog like temperament and affection makes the ideal lap cat and will live quite happily indoors. Very agile and powerful, the Siberian cat can easily leap and reach high places, including the tops of refrigerators and even doors. ",

"life\_span": "12 - 15",

"indoor": 0,

"lap": 1,

"alt\_names": "Moscow Semi-longhair, HairSiberian Forest Cat",

"adaptability": 5,

"affection\_level": 5,

"child\_friendly": 4,

"dog\_friendly": 5,

"energy\_level": 5,

"grooming": 2,

"health\_issues": 2,

"intelligence": 5,

"shedding\_level": 3,

"social\_needs": 4,

"stranger\_friendly": 3,

"vocalisation": 1,

"experimental": 0,

"hairless": 0,

"natural": 1,

"rare": 0,

"rex": 0,

"suppressed\_tail": 0,

"short\_legs": 0,

"wikipedia\_url": "https://en.wikipedia.org/wiki/Siberian\_(cat)",

"hypoallergenic": 1,

"reference\_image\_id": "3bkZAjRh1" }]

Task 3

* Task Name: Search Functionality UI Automation.
* Testcases:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Expected Condition | Test Data | Is Automated |
| 1 | Validate the display of search field | The Field should be displayed as “attach attachment” | - | Yes |
| 2 | Validate that the user can enter search key | User should be able to enter search target in the search field and find valid result | Example:  Namshi | Yes |
| 3 | Check the case the search keyword matches the full name of some of the search result | System should list all stores which have the EXACT name of the search key | Namshi | Yes, validate at least one |
| 4 | Check the case the search keyword matches the sub name of some of the search result | System should list all stores which CONTAINS name of the search key “store name” |  | Yes, validate at least one |
| 5 | Check invalid inputs | System should return empty list |  |  |
| 6 | Check listing after deleting the search words | System should list the featured stores list again” default list” |  |  |
| 7 | Check cancel search “clicking the x button in the right side of the search box” | System should list the featured stores list again” default list” |  |  |

* Note: There is read me file attached with every automation task inside the project itself.