**Tasks Covered :**

* **Reasons for failing the existing tests in your GitHub code**

1. **Chromedriver in your project is not an executable file, so it was throwing error “create process give error 193 %1 is not a valid win32 app”. The basic reason I can say is “a mismatch between a 32-bit executable and a 64-bit”.**
2. **I don’t think we need that file for our execution, so I deleted it from the project and issue got resolved.**
3. **Current (latest) Chrome Browser version is 70, and in your project the Chromedriver.exe version is 2.33 and it is not compatible with chrome browser 70. So, we were facing “java.net.SocketException:Connection reset”.**
4. **Updated the chromedriver.exe from 2.33 version to 2.36 version and issue got resolved.**
5. **After updating to 2.36, I got “SesionNotFoundException” sometimes, which is again a configuration/compatibility issue of browser with chromedriver.exe.**
6. **Again updated chromedriver.exe to 2.43 and it got resolved.**

* **Code Design Issues and Framework issues of your code**

1. **In every test class you are initializing the driver. Since, launching browser is common for every test case, we can add that code in one common method and we can utilize at every test case you need.**
2. **In every test case you have set the path of the driver, which we can do in one common method.**
3. **Before running/After running @Test method, neither killing the browser instances nor killing the chromedriver.exe.**
4. **In Each and every test case everything is hard coded and which is not recommended and which will lead to increase our effort and sometimes failing the test cases.**
5. **Most of the xpaths and ids are not dynamic, once page reloads/relaunch the browser those will get change.**
6. **Page object models are created in one class, but those are not getting initialized at all.**
7. **Hardcoded WebElements and not implemented Page Object models.**
8. **If any WebElement/data changes then we need to change that in each and every place where ever we have used to those. So hard coding is bad practice.**
9. **Thread.sleep() is not recommended at all since we are saying the Thread to sleep for sometime. Instead of that we can use WebDriver waits(Implicit/Explicit & Fluent).**
10. **System.out.println()’s are not recommended.**
11. **Not utilized the Framework default benefits/methods properly in whole project.**
12. **Good practice for testers is to write their code in src/test/java source folder rather than writing in src/main/java. Because src/main/java is recommended for developers. However, code will execute though😊.**
13. **Not used packages properly, we can create page classes in one package and test classes in another package.**
14. **javadocs are not added even for single method. We should write Javadoc for every method. Since, tomorrow someone will work on the same code so he can get to know the functionality easily by seeing java doc.**

**Improvements done by me:**

1. **Updated chromedriver.exe version to 2.43 which is compatible with latest chrome browser.**
2. **Deleted chromedriver file which is not necessary and resolved the error.**
3. **Created src/test/java and deleted src/main/java.**
4. **Created different,different packages for Pages, Tests and For Common methods.**
5. **Created GenericMethods class where I am setting the property of the driver, and initializing there by using @BeforeTest method and made it as always run.**
6. **So there will be no need to write the code for property setting and for driver initialization in each and every test. It will work for ‘n’ number of test cases. Here nothing but we are doing Code Resuability.**
7. **In @BeforeTest, checking is there any browser/driver instances. If there are any open instances I am killing those.**
8. **Created @AfterTest in GenericMethods.class, to kill the browser and driver instances once @Test execution completes.**
9. **Since for all the 3 test we are launching ClearTrip only, I have added that code in @BeforeTest only. So, that no need to write for each n every test case – Code Reusability.**
10. **Added ImplicitWait and PageLoadTimeout before launching the Cleartrip.**
11. **Each and every data we needed for test case execution is declared in class level so that there will be no hard coding and we can change easily whenever we want. (Better way is to take data from Excel/.xml file. Due to lack of time, stored in class level variables).**
12. **Added slf4j logger dependency to overcome sysouts and where ever I need to print something on console, used Loggers.**
13. **Created 3 page classes for 3 tests and implemented the PageObjectModels and initialized those with created driver.**
14. **What ever actions we are performing on web page, has been implemented in page classes not in test classes.**
15. **For every page class implemented private WebDriver.**
16. **Since the application is not stable, implemented ExplicitWait in each n every page class.**
17. **Created the WebElements dynamically. So, if page refreshes/relaunches the website again still those elements will be visible for driver.**
18. **Created 3 test classes for 3 tests in different package and calling the Pages classes methods from test classes – Abstraction.**
19. **For every verification implemented assert statements of TestNG.**
20. **Created testing.xml file to run all the test cases at a single run.**
21. **Added javadocs for every method.**
22. **Of course, all test cases will pass in my code.**

**Github:**

1. **Created Github account.**
2. **Forked your code to my Github account.**
3. **Cloned my Github code into my local using GitBash.**
4. **Created new branch “SandeepTestVagrant”. (It is always recommended to use new branch for every release for application testing ).**
5. **What ever the changes I have done in project has been committed into “SandeepTestVagrant” branch.**
6. **Link “**[**https://github.com/sandeep1817/codingRound/tree/SandeepTestVagrant**](https://github.com/sandeep1817/codingRound/tree/SandeepTestVagrant)**”.**

**Note: Please go through the code which I have implemented. Didn’t written many methods since tests are very small. Implemented many concepts, and if those were big test cases writing many methods will help to achieve Code Reusability.**

**“Please do reply back if any suggestions/Improvements😊”**

**Thankyou,**

**Sandeep Basireddy,**

**8669394794.**