**Date: 13/05/2019**

1. Refactored project Name/Artifact Id in pom.xml and settings.gradle files(codoingRound → codingRound)
2. Built using Maven No issues observed
3. Deleted Chrome Driver with no file extension which is breaking the execution
4. Updated Chromedriver for Chrome 74 (please consider changing this if the browser version in target machine is different)
5. Removed unused imports which affects the performance. This reduces the time of compilation.
6. Implemented **Public Library**(Provides utilities for automating any web-application) and Application Library (Provides utilities for automating specific application)
7. Implemented driver handling methods like closeWebBrowser, initialiseBrowser, getDriverInstance in Public Library
8. Implemented **Factory design pattern** (To choose variety of browsers like chrome, IE, Firefox etc.,) and **Singleton design pattern** for creating unique driver instance in PublicLibrary in initialiseBrowser.
9. Implemented method to Explicitly wait for an element which can be used instead of hard waits.
10. Implemented **AfterTest** and **BeforeTest** tags for setting up and cleaning up of test objects like drivers, exceldata.
11. Implemented **MiscConstants** class to maintain constant values used across the Application.
12. Implemented xpath locator for auto-suggestions in FlightBookingTest and SignInTest to be suitable for Explicit wait.
13. Updated Locator for To field in FlightBookingTest test (toTag → ToTag)
14. Implemented **Logger** which helps in understanding test status while running. Helps especially in case of running scripts in any **Server/BrowserStack/SauceLabs** and Headless execution.
15. Correction: Added break statement in switch in initialiseBrowser method.
16. Implemented PropertFileManager class to read data from properties files. Implemented **Singleton, Factory and Flyweight design patterns** for handling property file objects
17. Implemented getByObject method in Public library to **dynamically choose type of locator**. Advantages are locators will be present in a sharable file and Type of locator can be easily modified in case of changing of locators while application is changing.

**Date: 14/05/2019**

1. Updated Journey Type Loc as to pick it up from **Object Repository** (properties file)
2. Removed Hard waits and replaced them with **Explicit waits**. (waitFor → waitForElementToDisplay)
3. **Important** : Refactored code to handle auto-suggestions. Changed findElements to findElement. Reason is whenever only one element is used out of a list of elements its better to go with findElement only. findElements can be used when we use more than 2 elements out of list like handling Web Tables etc., This affects the **performance** of the code. We can observe the performance variations when the list of elements contacts 1000+ of elements(Ex: 1000+ rows of a webtable)
4. **Handled Iframe** for the login popup in SignInTest.java
5. Added **PageFactory** initialiser statement in order to access Page Objects in HotelBookingTest.java
6. Added separate **Page object** for HotelBookingPage
7. Added method to **handle datepicker by selecting date dynamically** instead of picking hardcoded date (19)
8. **Added missing steps** like selecting Location suggestions, Selecting Check-In date & Check-out date and verifying results are displayed in HotelBookingTest.java.
9. Implemented date handling API's like LocalDate etc.,
10. Created **MiscValues properties file** to store Miscellaneous strings like Application URL, file paths
11. Added code to handle MiscValues property file in PropertFileManager.
12. Added code in tests to read application URL from MiscValues property file (Suggested approach)
13. Implemented custom Click, Set Text elements in Public Library which internally waits for the element to appear before performing any action
14. Updated scripts with Reusable custom methods created in **ApplicationLibrary** and **PublicLibrary**
15. Added **Java docs and comments** wherever required
16. Created API to read data from excel based on TestCase Name. This enables Data driven testing approach. Used **Singleton, Factory and FlyWeight Design Patterns**.
17. Created TestData.xlsx sheet in FileUtils folder to store data required for tests.
18. Implemented Fillo API to read data from excel.
19. Added Fillo dependency to both **pom.xml** file and **build.gradle** file.
20. Created TestCaseTemplate and move all the test objects, Set Up methods (@**BeforeTest**) and Clean Up methods (@**AfterTest**). This allows more **readability** through the code in tests. The Before and After test methods can be modified easily across multiple tests.
21. Added log statements whenever required.