**HACKATHON PROJECT**

**PROJECT TEAM NAME: TECHIE SQUAD**

**PROJECT: FINDING HOSPITAL**

**PROJECT MEMBERS**

|  |  |
| --- | --- |
| NAME | EMPLOYEE ID |
| **ANJALI PATEL** | **2081108** |
| **ASHWIN R** | **2081016** |
| **NAVEEDHSHA K** | **2082228** |
| **SATHYA M** | **2080166** |
| **SOWMIYA K K** | **2081205** |
| **VINITH KUMAR** | **2082232** |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Content** | **Page No** |
| 1 | Project Descriptions | 3 |
| 2 | Website used for Automation | 3 |
| 3 | Framework and tools used | 3 |
| 4 | Software Specification | 3 |
| 5 | Browser Specification | 4 |
| 6 | Hardware Specification | 4 |
| 7 | Automation Feasibility | 4 |
| 8 | Key Automation Scope | 4 |
| 9 | Automation Feasibility Checklist Model | 5 |
| 10 | Basic Requirements of Project | 5 |
| 11 | Steps to Automate | 5 |
| 12 | Project Implementations | 5 |
| 13 | Project Structure | 6 |
| 14 | Execution | 8 |
| 15 | Screenshots | 8 |
| 16 | After Execution | 9 |
| 17 | Extent Report | 10 |
| 18 | Console Report | 11 |

**PROJECTS DESCRIPTION**:

1. For Bangalore city, identify Hospitals that is Open 24/7, has Parking facility with rating more than 3.5; Display the hospital names.  
2. In Diagnostics page, pick all the top cities name & store in a List; Display the same.  
3. Go to Corporate Wellness, fill invalid details, schedule & capture the warning message from the alert.

**WEBSITE USED FOR AUTOMATION:**

[*https://www.practo.com/*](https://www.practo.com/)

**FRAMEWORK AND TOOLS USED** **:**

|  |  |
| --- | --- |
| **Framework and Tools Used** | **Version** |
| WebDriverManager | - |
| Maven (Build Tool) | 3.8.4 |
| TestNG | 7.3.0 |
| Extent Report API | 3.1.5 |
| Selenium | 4.00 |
| Apache POI | 3.17 |
| Jenkins | - |

**SOFTWARE SPECIFICATIONS:**

* Programming language used           - JAVA
* IDE used                                           - Eclipse
* Automation framework                - Selenium
* Libraries added                                - Maven dependency Library, TestNG Library
* Browser used                                    - Chrome, MS Edge, Firefox

**BROWSER SPECIFICATIONS:**

|  |  |
| --- | --- |
| **Browser** | **Version** |
| MS Edge | 97.0 |
| Chrome | 95.0 |
| Firefox | 96.0 |

**HARDWARE SPECIFICATIONS:**

* Processor                             - intel core i3 7th Gen
* RAM                                     - 4GB Minimum / 8GB Recommended
* Operating System                 - Windows 10

**AUTOMATION FEASIBILITY:**

Feasible, Selenium is able to locate all web elements and perform actions on them

**KEY AUTOMATION SCOPE:**

* + Handling different browser windows, search option
  + Validation of date controls
  + Capture warning message
  + Extract list items & store in collections
  + Navigating back to home page
  + Scrolling down in web page
  + To capture the screenshot

**AUTOMATION FEASIBILITY CHECKLIST MODEL:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **READINESS CHECK** | **YES/NO** |
| **1** | Product stable | Yes |
| **2** | Any planned enhancement in future | No |
| **3** | Bugs affect the major functionality | Yes |
| **4** | Are the test and precondition are detailed | Yes |
| **5** | Test case and test detail analysis done | Yes |

**BASIC REQUIREMENTS OF PROJECT:**

* Class for driver setup and other utilities.
* Data driven concept is used for the pages to read data and write data.
* This are the basic requirements Initially we need to create a Maven Project.
* In that project we have to create test scenarios and test classes.
* According to this project we need to create methods in that pages.
* We have to use Base of this project.

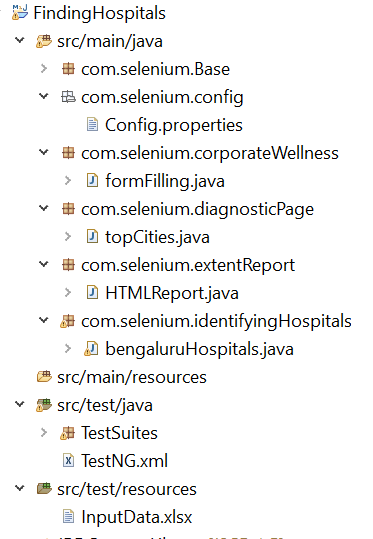
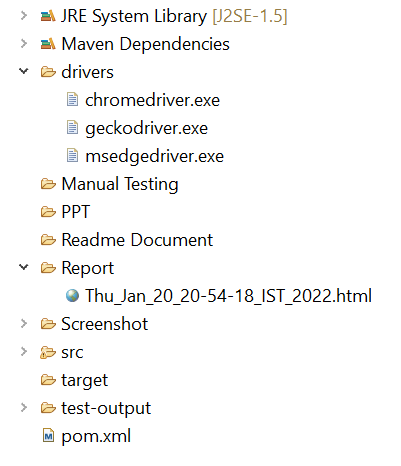
**STEPS TO AUTOMATE:**

* Extract the zip file and save it in folder.
* Import the extracted folder from the eclipse as existing project.
* Right click on the **testng.xml**file and click **run as** then choose **TestNG Suite** it will run on Chrome browser, MS Edge browser and Firefox browser.
* After the result, check the passes and failed testcase on the testing console.
* And then check the screenshot in folder.
* Finally, the automated concept is over.

**PROJECT IMPLEMENTATIONS:**

* **Maven:** To simplify Build Process and adding of dependencies.
* **TestNG Library:** For separate execution of each test case in Functional requirement.
* **Page object Model (POM):** To get the returned objects of each page class during automation.
* **Apache POI:** To read excel file and extract values from it during run time.
* **Config Properties:** To read user properties during test execution.

**PROJECT STRUCTURE :**



**src/main/java**

* **com.selenium.Base :**
* **Base.java:**  It is implemented to setup the driver, invoke the browser and to do some functionalities.
* **com.selenium.config:**
* **Config.properties:** Element locators that are required to automate the tests are declared.
* **com.selenium.corporateWellness:**
* **formFilling.java:** This class contains the methods to fill the form in ‘Corporate Wellness’ page **.**
* **com.selenium.diagnosticPage:**
* **topCities.java:** This class contains the methods to store and print the top cities name in ‘Book Diagnostic Tests’ page and to navigate back to the home page.
* **com.selenium.extentReport:**
* **HTMLReport.java :** This class contain a method to generate the html report.
* **com.selenium.identifyingHospitals:**
* **bengaluruHospitals.java :** This class contains the methods to store and print the Bangalore hospitals names which satisfy the given condition and navigate back to the home page.

**src/test/java**

* **TestSuites:**
* **TestCases.java :** It contains all testcases for the execution**.**
* **TestNG.xml :** This file contains ordered way of execution of the full project.

**src/test/resources**

* **InputData.xlsx :** It contains input value for the testcases.

**drivers**: It contains the chrome, edge and firefox drivers.

**Report**: The html report generated will be stored here.

**Screenshots :** The Screenshots of the project are saved here.

* **FailedCases:** Stores the screenshots of failed cases.

**Pom.xml:**  > The dependency needed for the project will be stored here.

**EXECUTION:**

Right click on the **testng.xml page** and click **run as** then choose **TestNG Suite.**

**SCREENSHOTS:**

Graphical user interface, application

Description automatically generated**HOME PAGE:**

**Graphical user interface, text, application

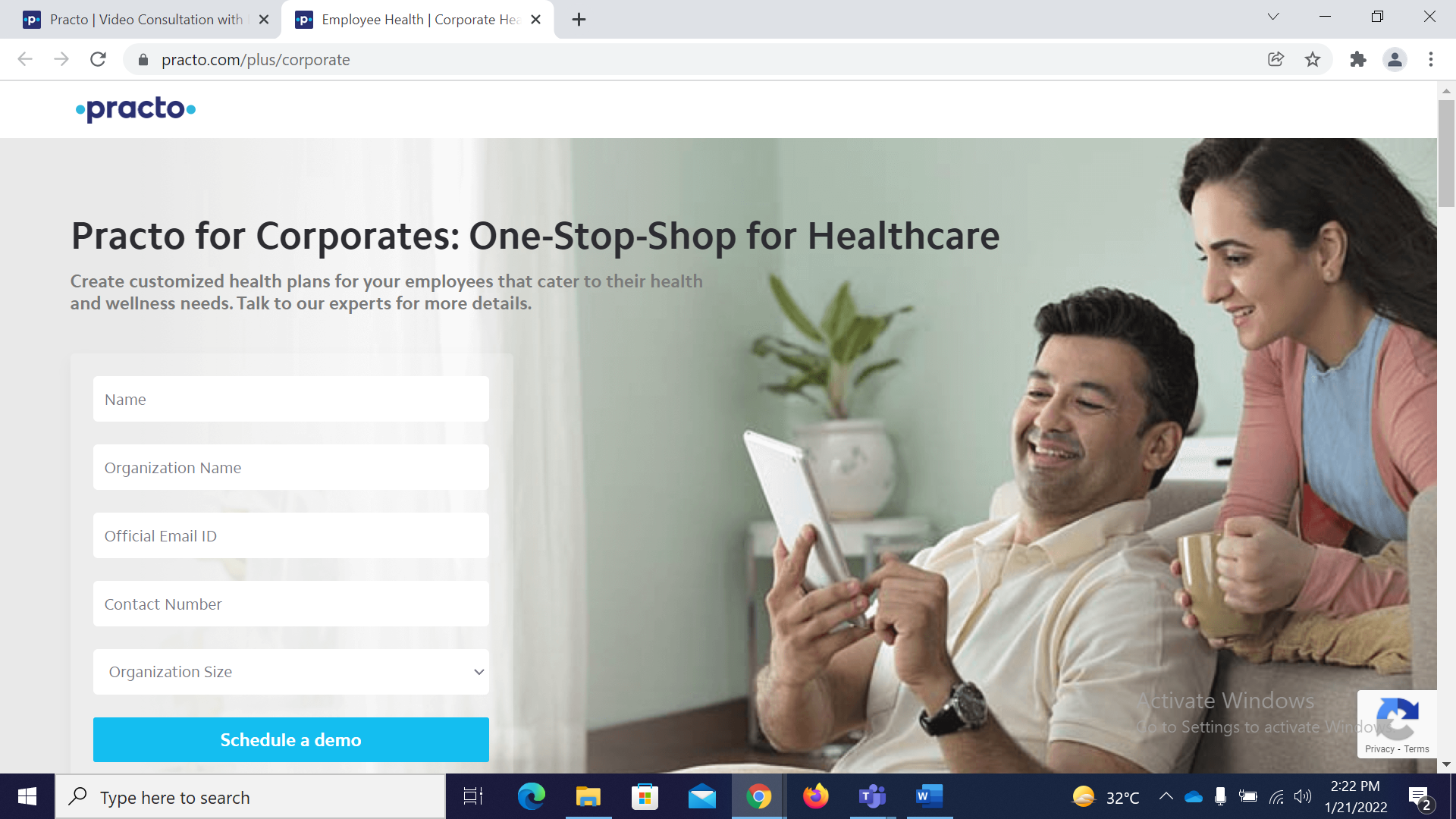
Description automatically generatedBANGALORE HOSPITALS**:

**DIAGNOSTIC PAGE / TOPCITIES:**

Graphical user interface, application, website

Description automatically generated

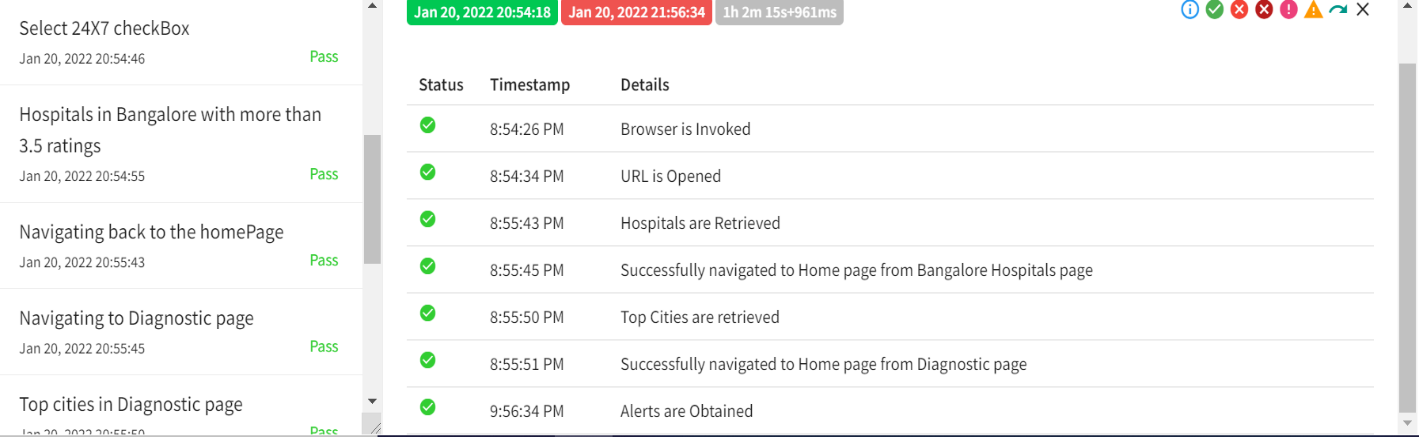
**CORPORATE WELLNESS :**



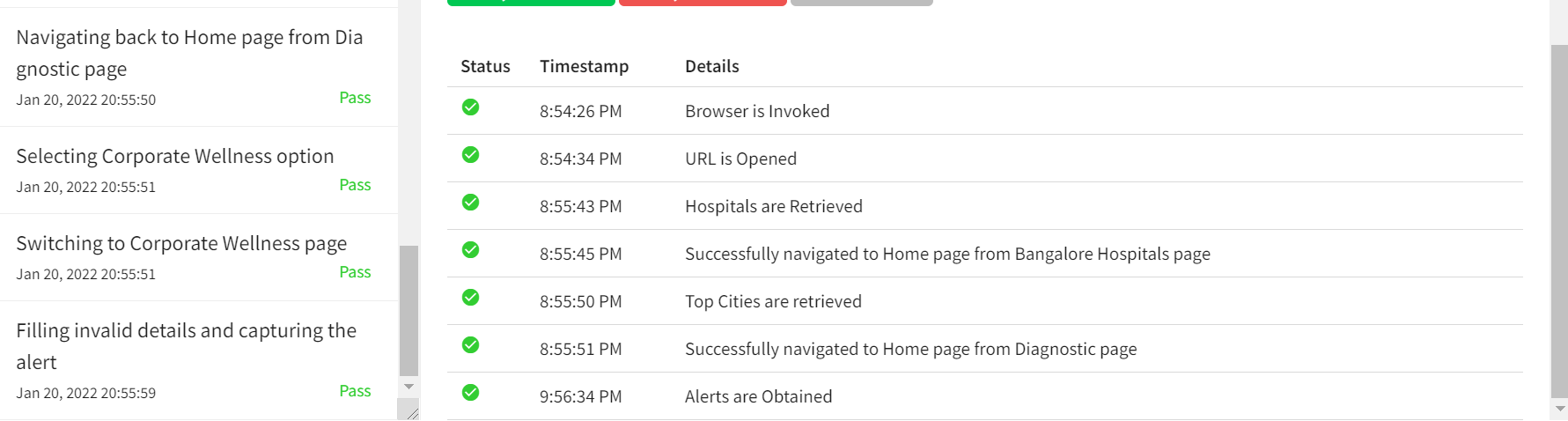
**AFTER EXECUTION:**

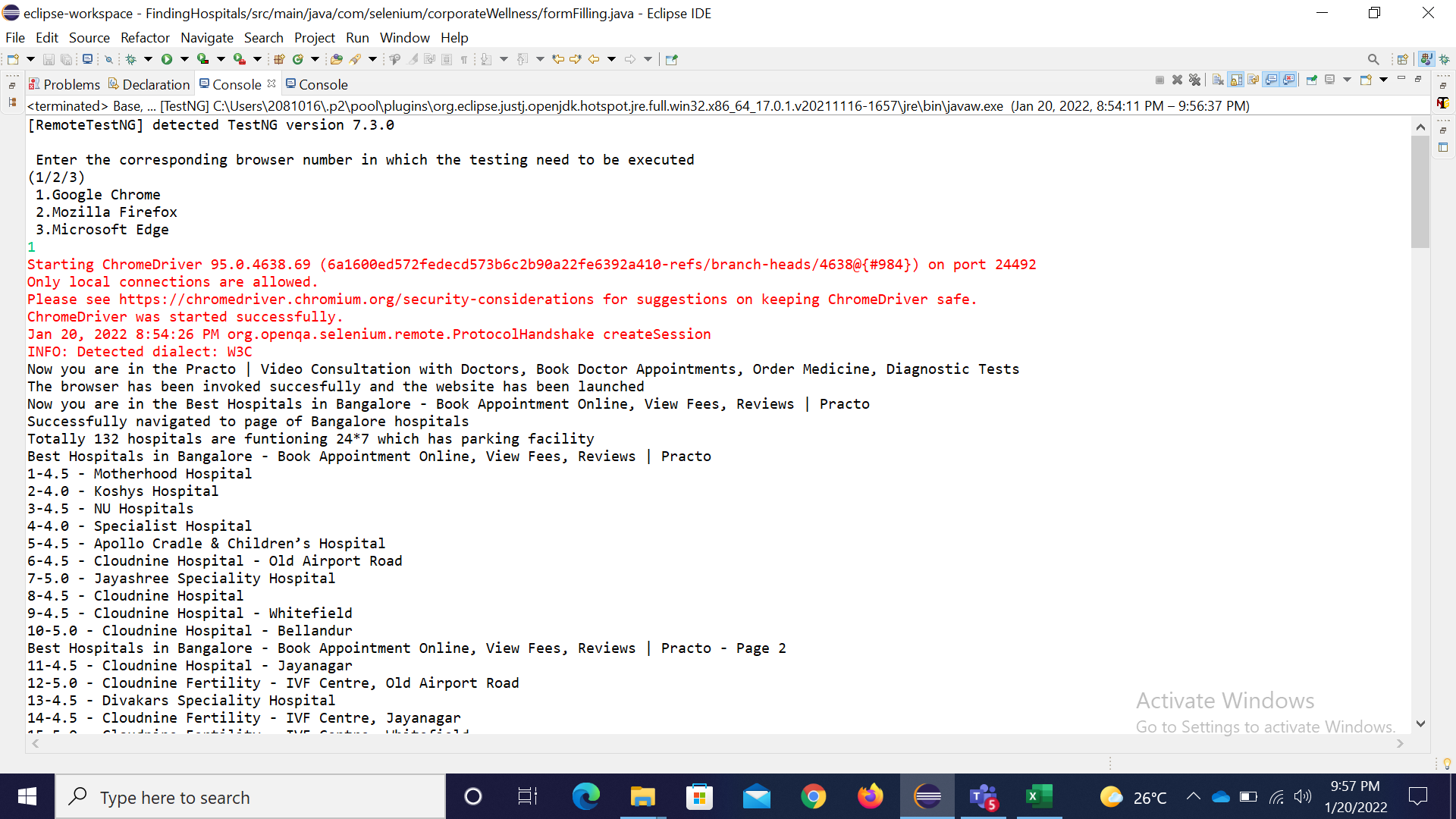
            Refresh the project after completing the full execution of code to get the report file.

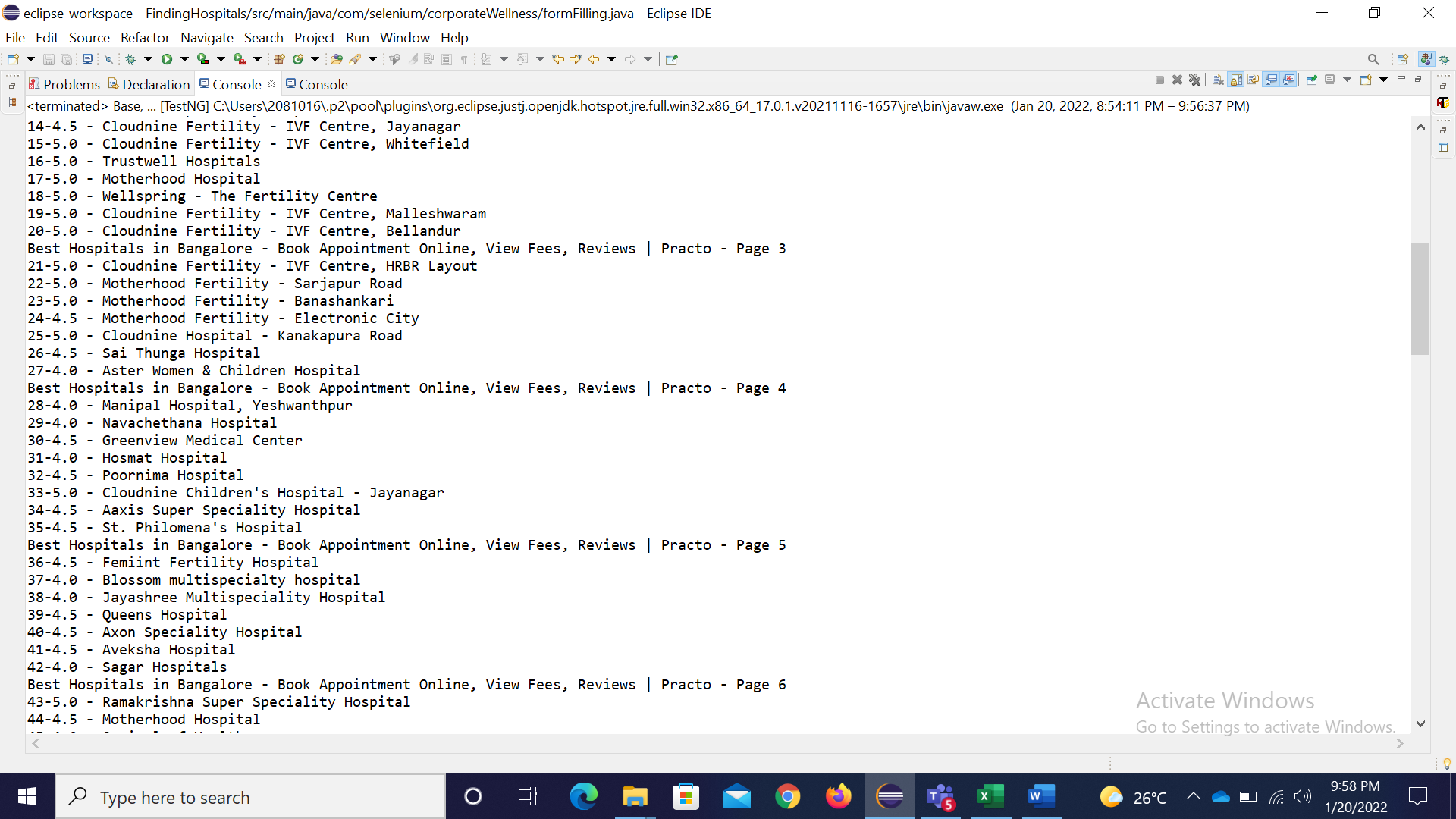
**EXTENT REPORT:**

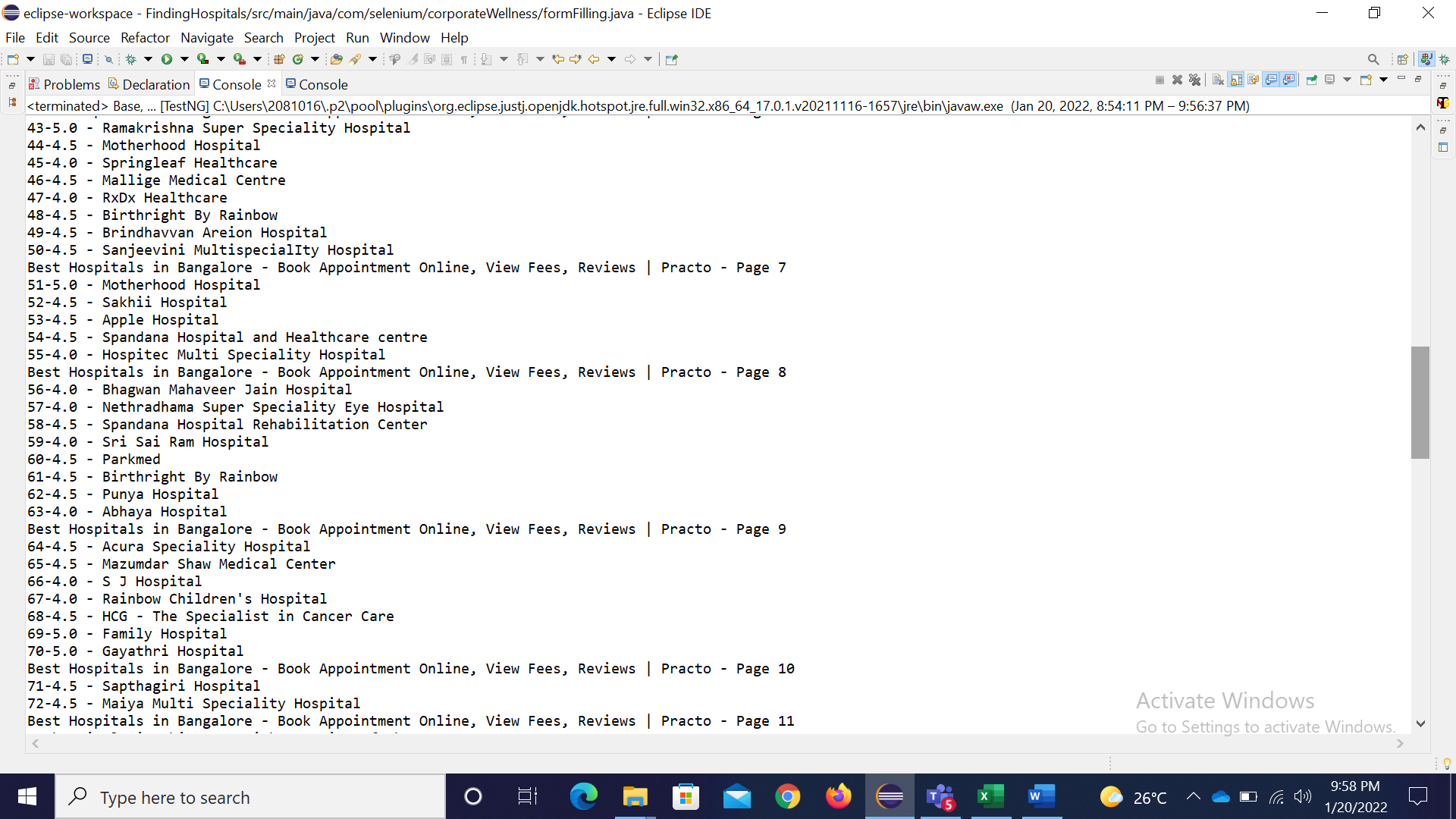
**Graphical user interface, text, application

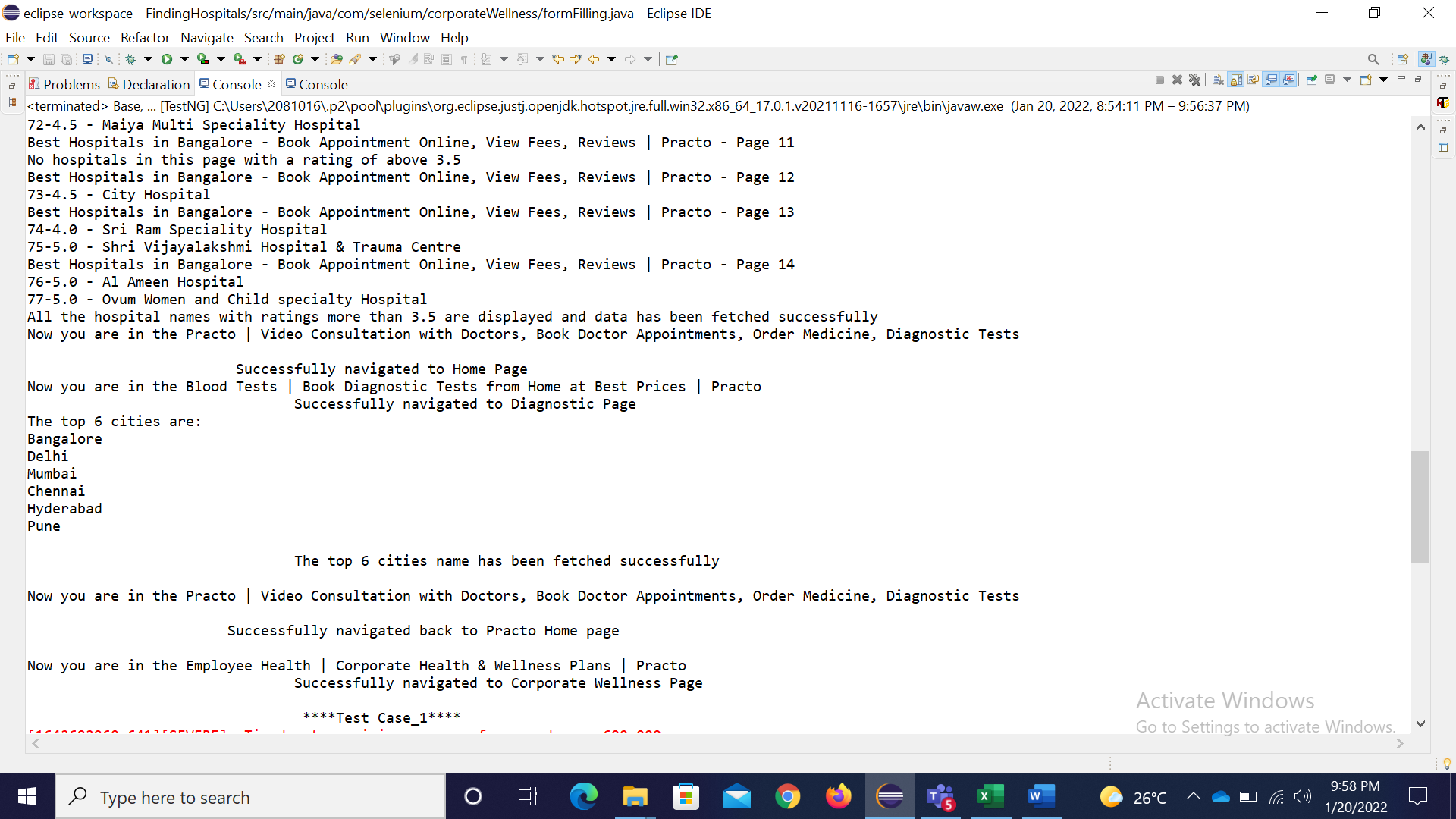
Description automatically generated**

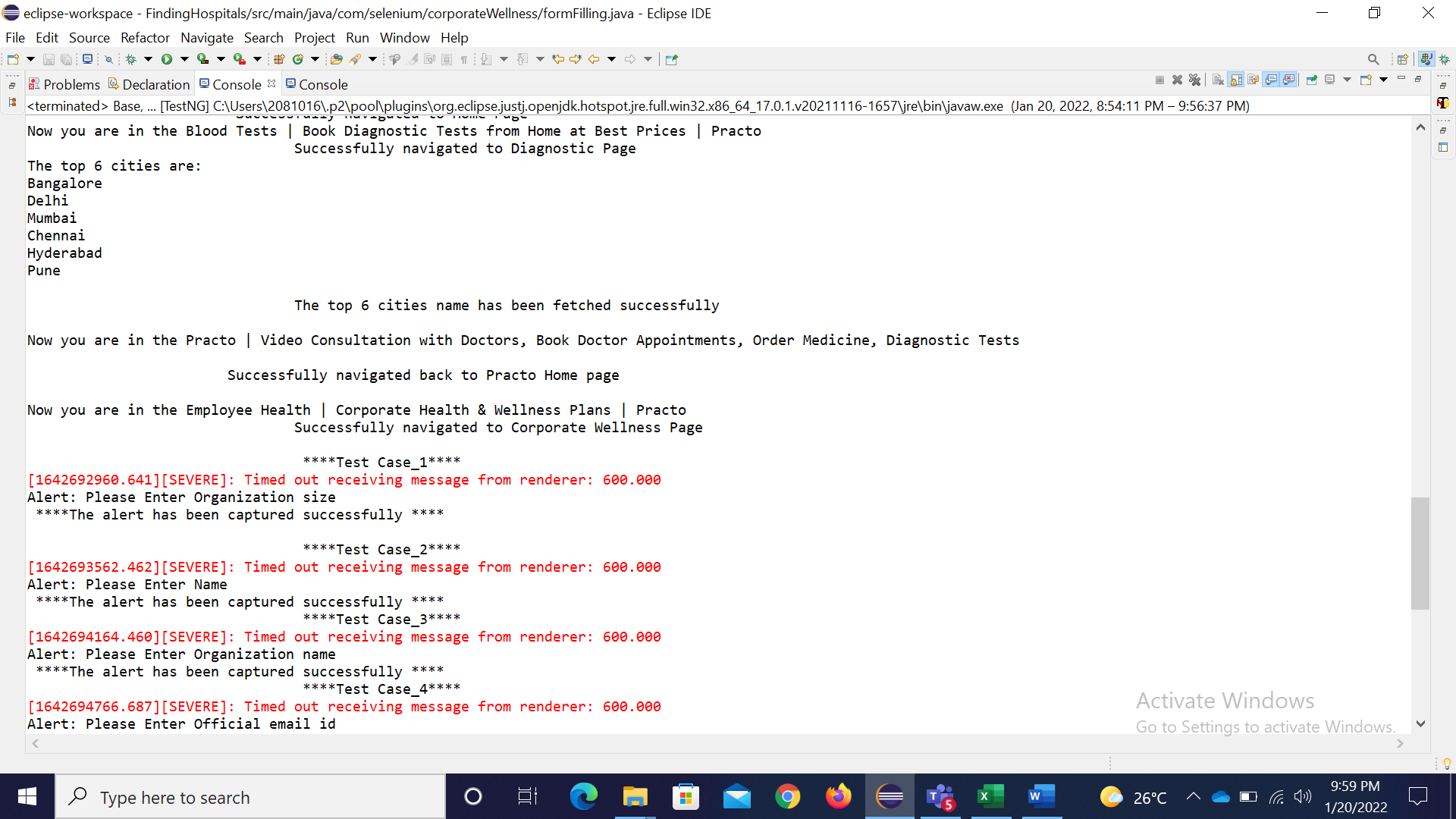


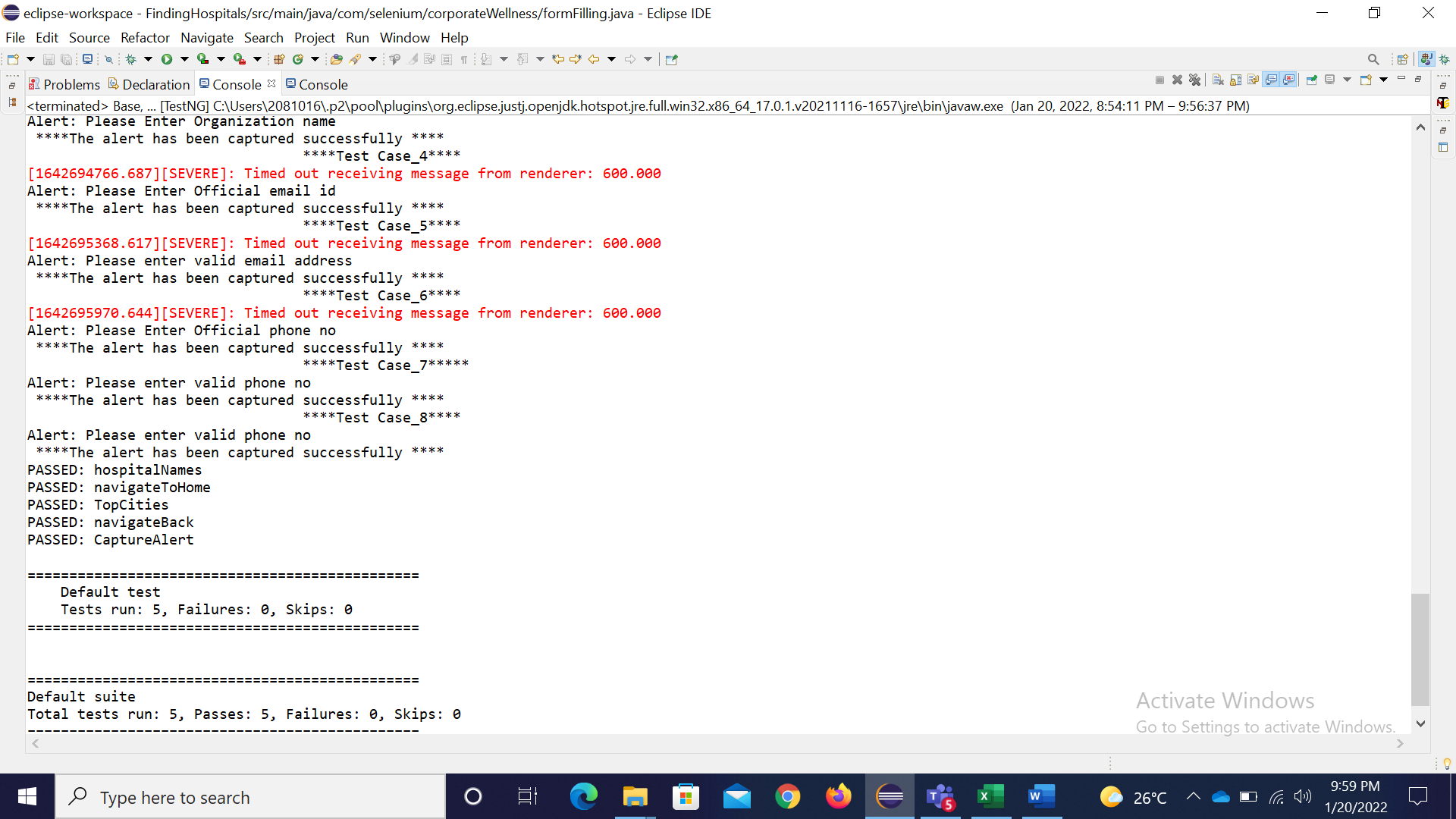
**CONSOLE REPORT:** 











The results of all the test cases is displayed in the console. The Extent report of the project is generated once the automation gets completed and is seen in the Report folder . All the screenshots will be available in the “Screenshot” folder.