**TASK 5:**

**Aim:** To perform parameterized testing using Junit

**Procedure:**

**Steps:**

1. Creating Java Project
   * Click on File and select New project
   * Enter project name as com.vogella.Junit.Prime
   * Click on Next and then on Finish
2. Creating java Test project
   * Right click on com.vogella.Junit.Prime
   * Click on properties and select tab build path
   * Click on source and click on Create New Folder
   * Give the folder name as Test and click on next
   * Click on Finish and then on OK
3. Creating java class
   * Right click on com.vogella.Junit.Prime and click on New
   * Click on class and give class name as Prime
   * Click on Finish
   * Type the following code

**Prime.java**

**package** com.vogella.JUnit.Prime;

**public** **class** Prime

{

**public** **int** isPrime(**int** n)

{

**int** c=0;

**for**(**int** i=1;i<n;i=i+1)

{

**if**(n%i==0)

c=c+1;

**if**(c>1)

**return** 0;

}

**if**(c==1)

**return** 1;

**return** 0;

}

}

4.Create java test class

* Right click on com.vogella.JUnit.Prime and then click on new.
* Click on JUnit test case
* Change the name of folder src to test in source folder tab
* Click on browse and select Prime class and click on Next
* Click on Finish and then on OK
* Add the following code

**Prime1.java**

package com.vogella.JUnit.Prime;

import static org.junit.Assert.\*;

import java.lang.reflect.Array;

import java.util.Arrays;

import java.util.Collection;

import org.junit.Test;

import org.junit.runner.RunWith;

import org.junit.runners.Parameterized;

import org.junit.runners.Parameterized.Parameters;

@RunWith(Parameterized.class)

public class Prime1 {

private int expected;

private int first;

public Prime1(int expectedResult,int first)

{

this.first=first;

this.expected=expectedResult;

}

@Parameters

public static Collection<Integer[]>PrimeNumber(){

return Arrays.asList(new Integer[][]{{1,7},{1,4},{1,3}});

}

@Test

public void testIsPrime() {

Prime pr=new Prime();

System.out.println("Prime with parameters:"+first);

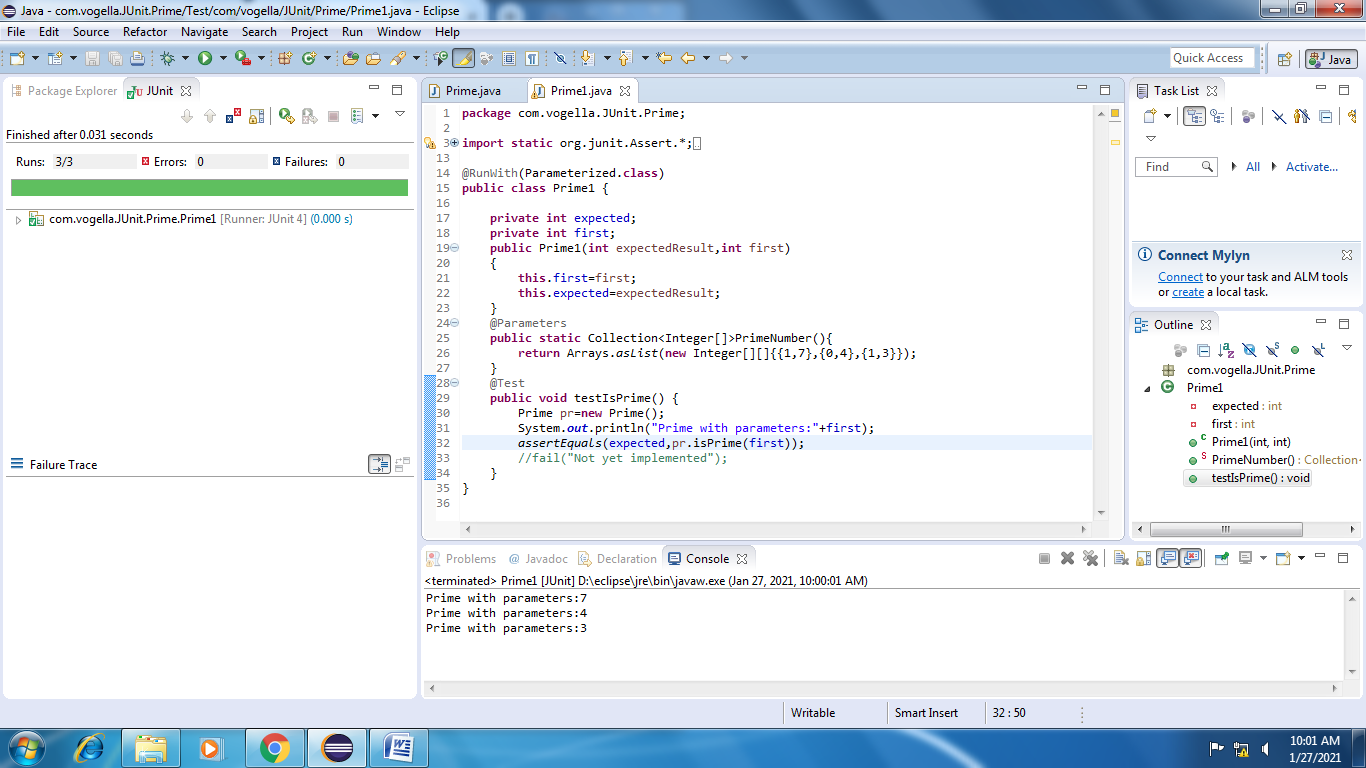
assertEquals(expected,pr.isPrime(first));

}

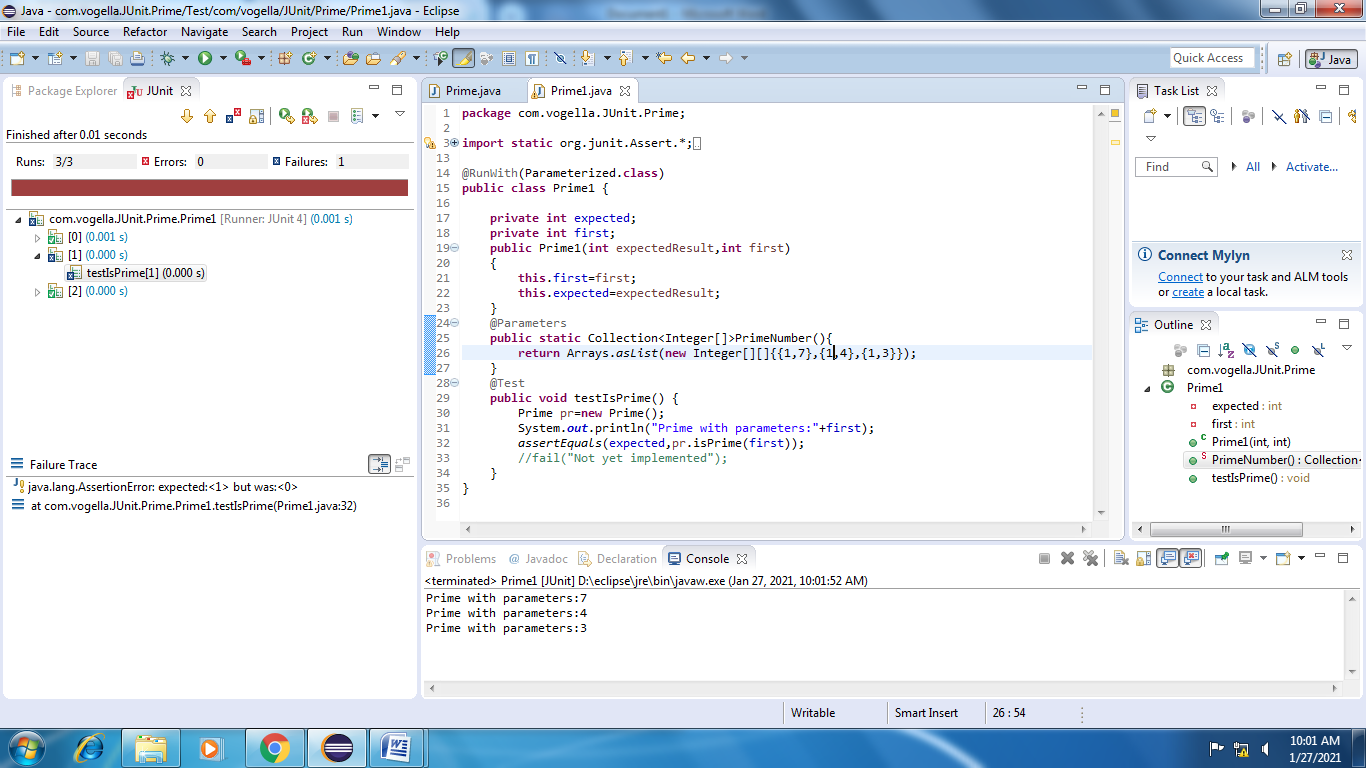
}

**Output:**

Pass case for Prime



Fail case for Prime



**Test suite design:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Name**: Parameterized Testing | | | | | | |
| **Test case id**: ID\_5 **Test Designed by:** Kundana  **Test Priority**: low **Test Designed Date**:  **Module Name**: Junit Testing **Test Executed by**: Kundana  **Test Title**: Blackbox testing  **Test Executed Date**:  **Description:** Test case for problem using Parameterized Testing | | | | | | |
| **Pre-Condition**: User should give one input number and one expected output | | | | | | |
| **Stage** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **status (Pass/Fail)** | **Remarks** |
| 1 | One  valid  integer  value | 6 | 0 | 0 | Pass | Nil |
| 2 | 13 | 1 | 1 | Pass | Nil |
| 3 | 29 | 1 | 1 | Pass | Nil |
| 4 | 15 | 0 | 0 | Pass | Nil |
| 5 | 9 | 0 | 1 | Fail | Nil |
| **Post condition**: Expected result should match with value returned by function | | | | | | |

**Result:** Performing parameterized testing using Junit has been done successfully.

**TASK 6:**

**Aim**: To perform testing visual c# using NUnit.

**Procedure**:

**Steps**:

1. Creating Class Library.

* Click on File and select New project
* Click on Visual C# and rename it as Leela64
* Click on Finish

1. Add the following code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Leela64

{

public class Addition

{

public int add(int x, int y)

{

return (x + y);

}

}

public class Subtraction

{

public int sub(int x, int y)

{

return (x - y);

}

}

public class Division

{

public int div(int x, int y)

{

return (x / y);

}

}

public class Multiplication

{

public int mul(int x, int y)

{

return (x \* y);

}

}

}

1. Add new class Library

* Right click on Project in Solution explorer.
* Click on Add and New Project(Class Library)
* Type the following code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using NUnit.Framework;

using Leela64;

namespace Test

{

[TestFixture]

public class Class1

{

[Test]

public void prs()

{

Multiplication m1 = new Multiplication();

Addition a1 = new Addition();

Subtraction s1 = new Subtraction();

Division d1 = new Division();

Assert.AreEqual(20, m1.mul(5, 4));

Assert.AreEqual(1, s1.sub(8, 7));

Assert.AreEqual(7, a1.add(3, 4));

Assert.AreEqual(1, d1.div(5, 5));

}

[Test]

public void prs1()

{

Multiplication m1 = new Multiplication();

Addition a1 = new Addition();

Subtraction s1 = new Subtraction();

Division d1 = new Division();

Assert.AreEqual(10, m1.mul(2, 4));

Assert.AreEqual(1, s1.sub(8, 7));

Assert.AreEqual(7, a1.add(3, 4));

Assert.AreEqual(1, d1.div(5, 5));

}

}

}

1. Add references

* Right Click on Test and click on Add references
* Add the Existing Project and also nunit.framework.dll
* Click on OK

1. Build Solution

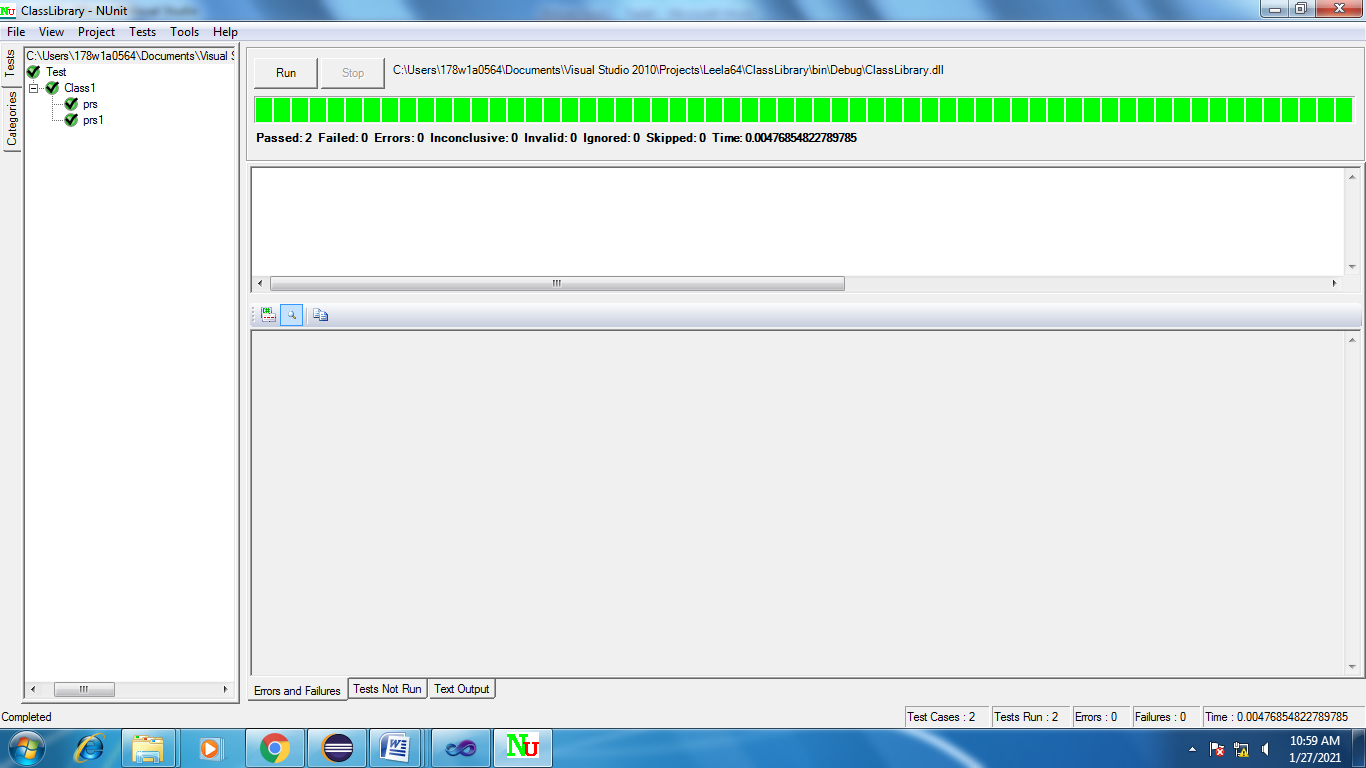
* Click on Build
* Now click on Build Solution

1. Testing using NUnit

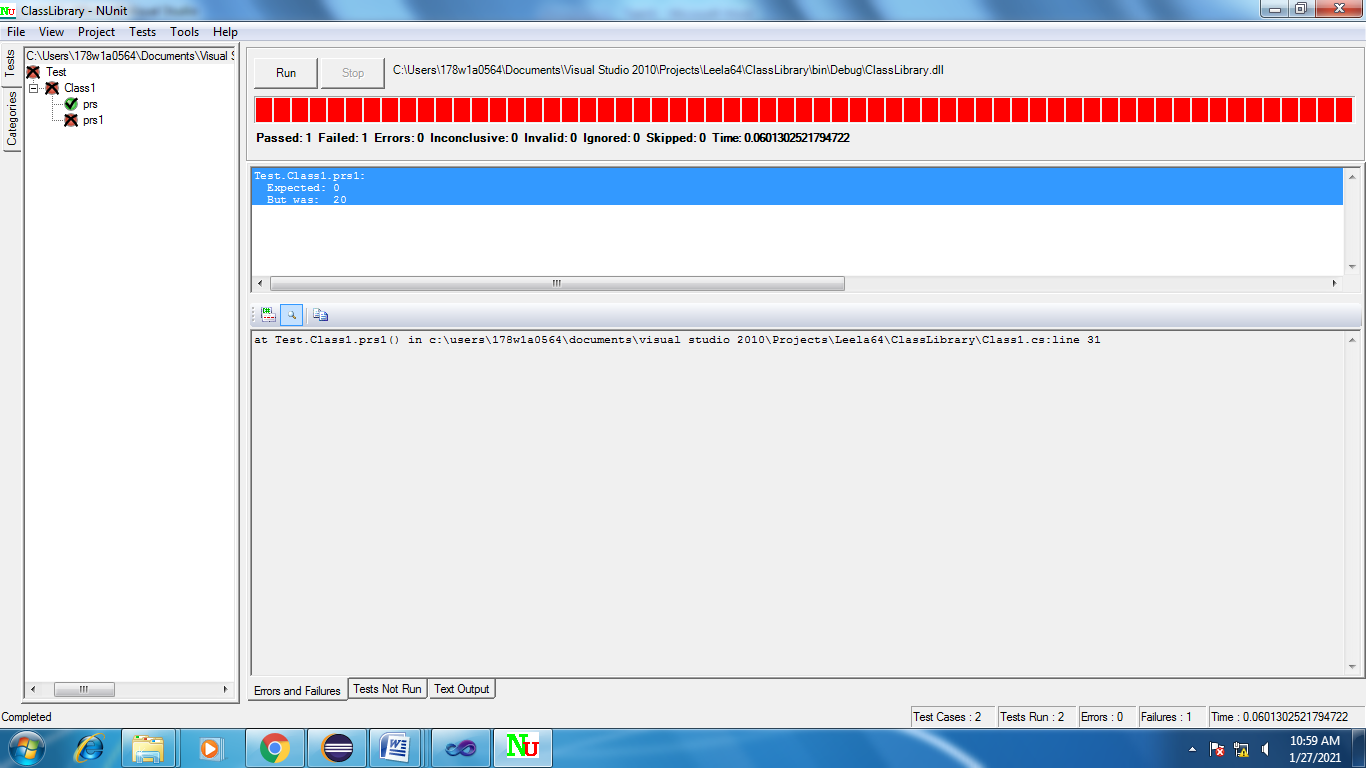
* Open NUnit and click on File
* Click on Open project and select the Visual Project
* Select the test code project.
* Click on bin and then on debug and then on Test.nunit.dll
* Click on Run

**Output**:

**Pass Case**

****

**Fail Case**

****

**Test suite design:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Name**: NUnit Testing | | | | | | |
| **Test case id**: ID\_6 **Test Designed by:** Kundana  **Test Priority**: low **Test Designed Date**:  **Module Name**: Nunit Testing **Test Executed by**: Kundana  **Test Title**: Blackbox testing  **Test Executed Date**:  **Description:** Test case for problem using NUnit Testing | | | | | | |
| **Pre-Condition**: User should give two input numbers and one expected output | | | | | | |
| **Stage** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **status (Pass/Fail)** | **Remarks** |
| 1 | Addition | 6,4 | 10 | 10 | Pass | Nil |
| 2 | Subtraction | 7,3 | 4 | 4 | Pass | Nil |
| 3 | Multiplication | 2,4 | 8 | 8 | Pass | Nil |
| 4 | Division | 6,3 | 2 | 3 | Fail | Nil |
| **Post condition**: Expected result should match with value returned by function | | | | | | |

**Result:** Performing testing visual c# using NUnit has been implemented successfully.

**TASK 7:**

**Aim**: To perform testing visual c# using NUnit.

**Procedure**:

**Steps**:

1. Creating Class Library.

* Click on File and select New project
* Click on Visual C# and rename it as Triangle64
* Click on Finish

1. Add the following code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Triangle64

{

public class Triangle

{

public int ischeck(int x, int y,int z)

{

if(x==y && y==z && z==x)

return 1;

if(x==y || y==z || z==x)

return 2;

else

return 3;

}

}

}

1. Add new class Library

* Right click on Project in Solution explorer.
* Click on Add and New Project(Class Library)
* Type the following code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using NUnit.Framework;

using Triangle64;

namespace Test

{

[TestFixture]

public class Class1

{

[Test]

public void prs()

{

Triangle d1=new Triangle();

Assert.AreEqual(1, d1.ischeck(5,5,5));

}

}

}

1. Add references

* Right Click on Test and click on Add references
* Add the Existing Project and also nunit.framework.dll
* Click on OK

1. Build Solution

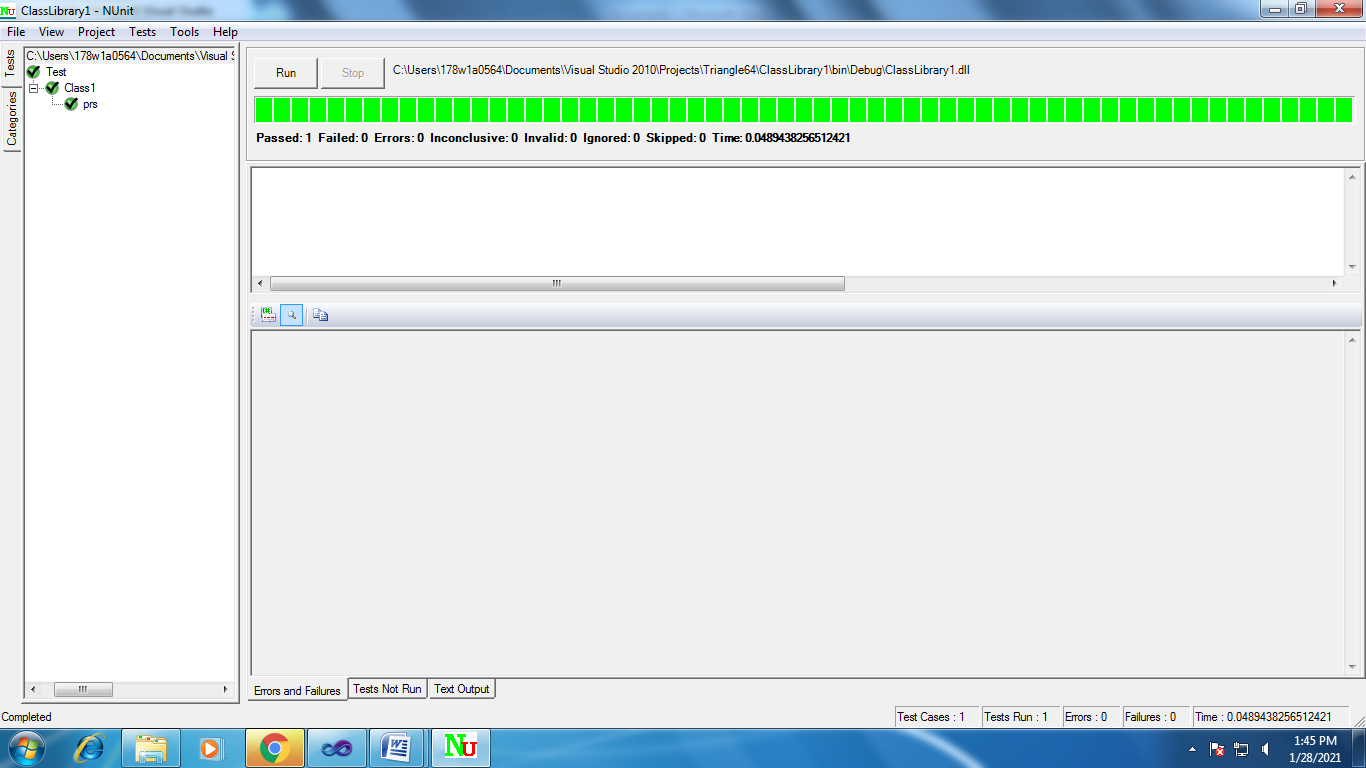
* Click on Build
* Now click on Build Solution

1. Testing using NUnit

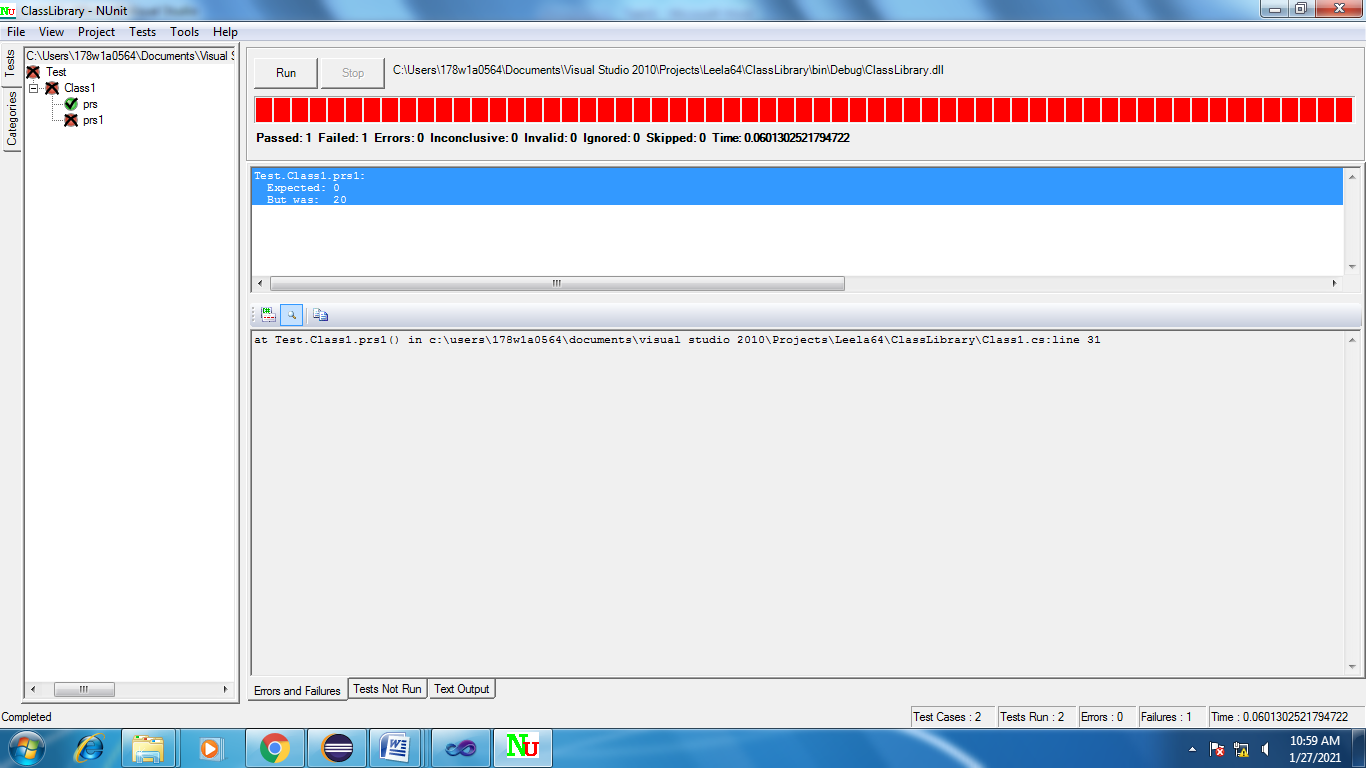
* Open NUnit and click on File
* Click on Open project and select the Visual Project
* Select the test code project.
* Click on bin and then on debug and then on Test.nunit.dll
* Click on Run

**Output**:

**Pass Case**

****

**Fail Case**

****

**Test suite design:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Name**: Parameterized Testing | | | | | | |
| **Test case id**: ID\_7 **Test Designed by:** Kundana  **Test Priority**: low **Test Designed Date**:  **Module Name**: Nunit Testing **Test Executed by**: Kundana  **Test Title**: Blackbox testing  **Test Executed Date**:  **Description:** Test case for problem using Parameterized Testing | | | | | | |
| **Pre-Condition**: User should give one three input numbers and one expected output | | | | | | |
| **Stage** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **status (Pass/Fail)** | **Remarks** |
| 1 | Three  valid  integer  value | 1,1,1 | 1 | 1 | Pass | Nil |
| 2 | 1,2,3 | 3 | 3 | Pass | Nil |
| 3 | 2,9,9 | 2 | 2 | Pass | Nil |
| 4 | 1,1,5 | 2 | 2 | Pass | Nil |
| 5 | 9,9,9 | 1 | 3 | Fail | Nil |
| **Post condition**: Expected result should match with value returned by function | | | | | | |

**Result:** Performing testing visual c# using NUnit has been implemented successfully.

**TASK 8:**

**Aim**: To perform testing visual c# using NUnit.

**Procedure**:

**Steps**:

1. Creating Class Library.

* Click on File and select New project
* Click on Visual C# and rename it as Prime64
* Click on Finish

1. Add the following code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Prime64

{

public class Prime

{

**public** **int** isPrime(**int** n)

{

**int** c=0;

**for**(**int** i=1;i<n;i=i+1)

{

**if**(n%i==0)

c=c+1;

**if**(c>1)

**return** 0;

}

**if**(c==1)

**return** 1;

**return** 0;

}

}

}

1. Add new class Library

* Right click on Project in Solution explorer.
* Click on Add and New Project(Class Library)
* Type the following code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using NUnit.Framework;

using Prime64;

namespace Test

{

[TestFixture]

public class Class1

{

[Test]

public void prs()

{

Prime p=new Prime();

Assert.AreEqual(1, p.isPrime(5));

}

}

}

1. Add references

* Right Click on Test and click on Add references
* Add the Existing Project and also nunit.framework.dll
* Click on OK

1. Build Solution

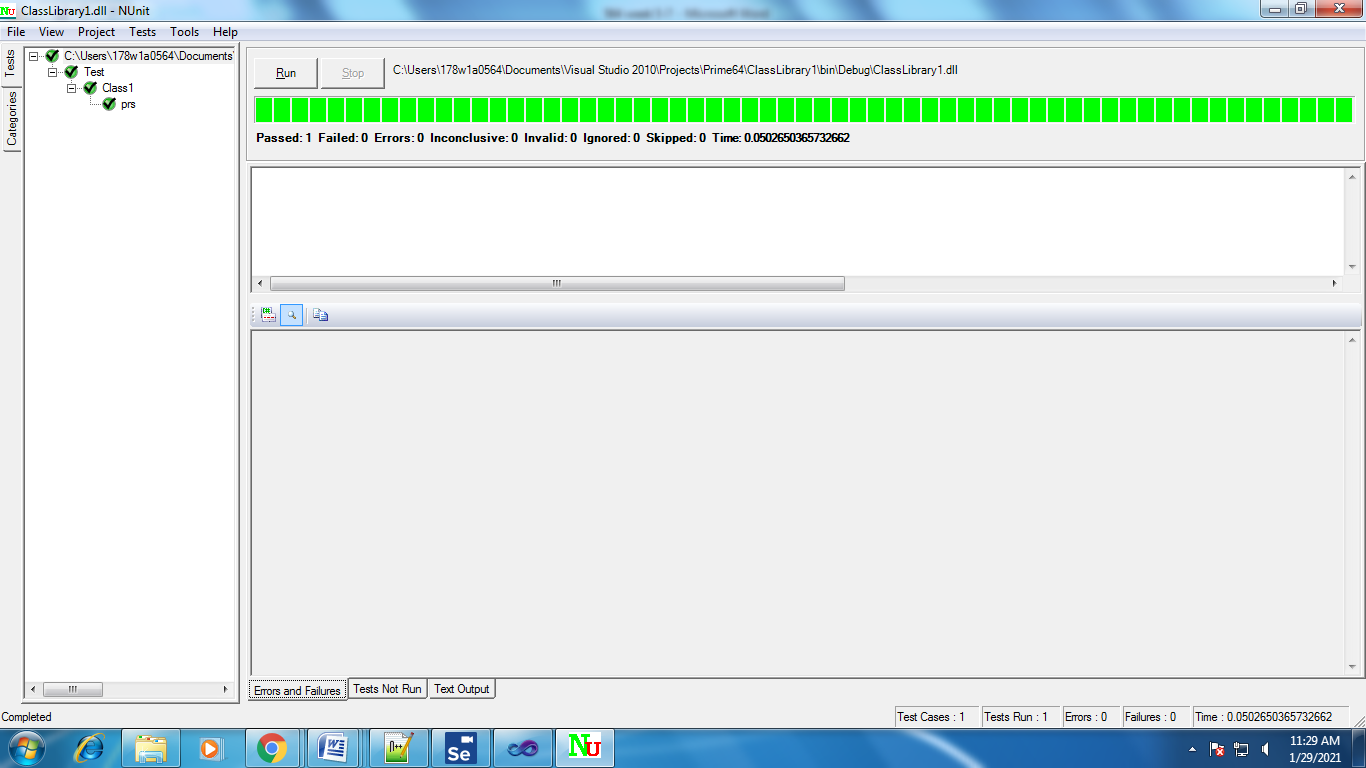
* Click on Build
* Now click on Build Solution

1. Testing using NUnit

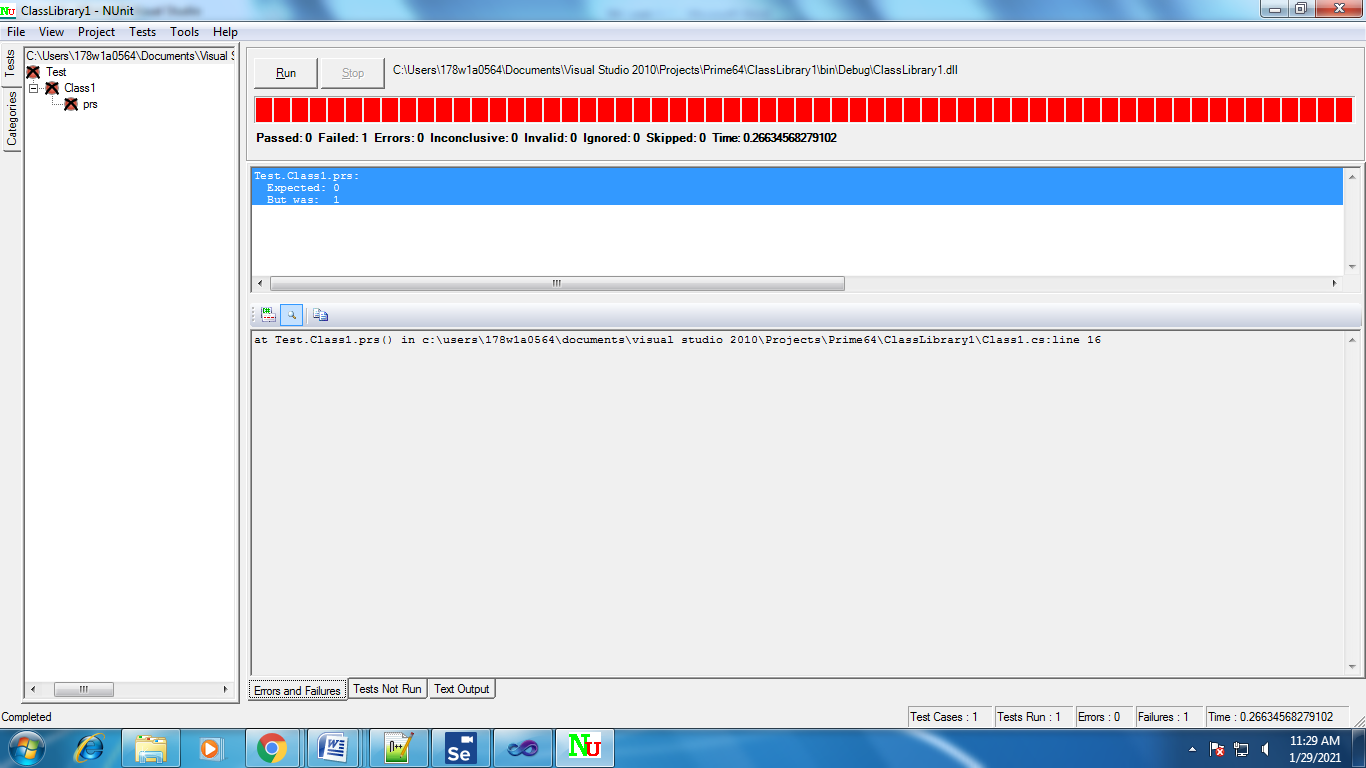
* Open NUnit and click on File
* Click on Open project and select the Visual Project
* Select the test code project.
* Click on bin and then on debug and then on Test.nunit.dll
* Click on Run

**Output**:

**Pass Case**

****

**Fail case**

****

**Test suite design:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Name**: Parameterized Testing | | | | | | |
| **Test case id**: ID\_8 **Test Designed by:** Kundana  **Test Priority**: low **Test Designed Date**:  **Module Name**: Nunit Testing **Test Executed by**: Kundana  **Test Title**: Blackbox testing  **Test Executed Date**:  **Description:** Test case for problem using Parameterized Testing | | | | | | |
| **Pre-Condition**: User should give one input number and one expected output | | | | | | |
| **Stage** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **status (Pass/Fail)** | **Remarks** |
| 1 | One  valid  integer  value | 6 | 0 | 0 | Pass | Nil |
| 2 | 13 | 1 | 1 | Pass | Nil |
| 3 | 29 | 1 | 1 | Pass | Nil |
| 4 | 15 | 0 | 0 | Pass | Nil |
| 5 | 9 | 0 | 1 | Fail | Nil |
| **Post condition**: Expected result should match with value returned by function | | | | | | |

**Result:** Performing testing visual c# using NUnit has been implemented successfully.

**TASK 9:**

**Aim:** Toperform record play back using Selenium IDE Script.

**Description:**

The entire script creation process can be classified into 3 chunks:

**Process 1: Recording** – Selenium IDE aids the user to record user interactions with the browser and thus the recorded actions as a whole are termed as Selenium IDE script.

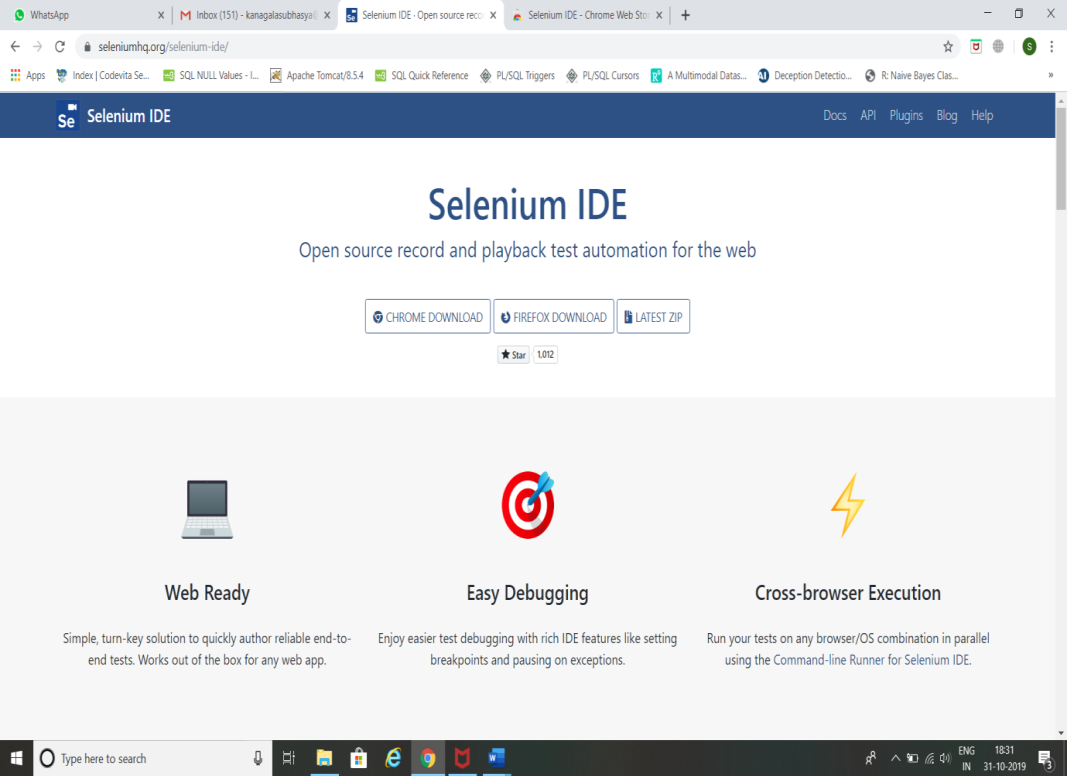
**Process 2: Playing back**– In this section, we execute the recorded script so as to verify and monitor its stability and success rate.

**Process 3: Saving** – Once we have recorded a stable script, we may want to save it for future runs and regressions.

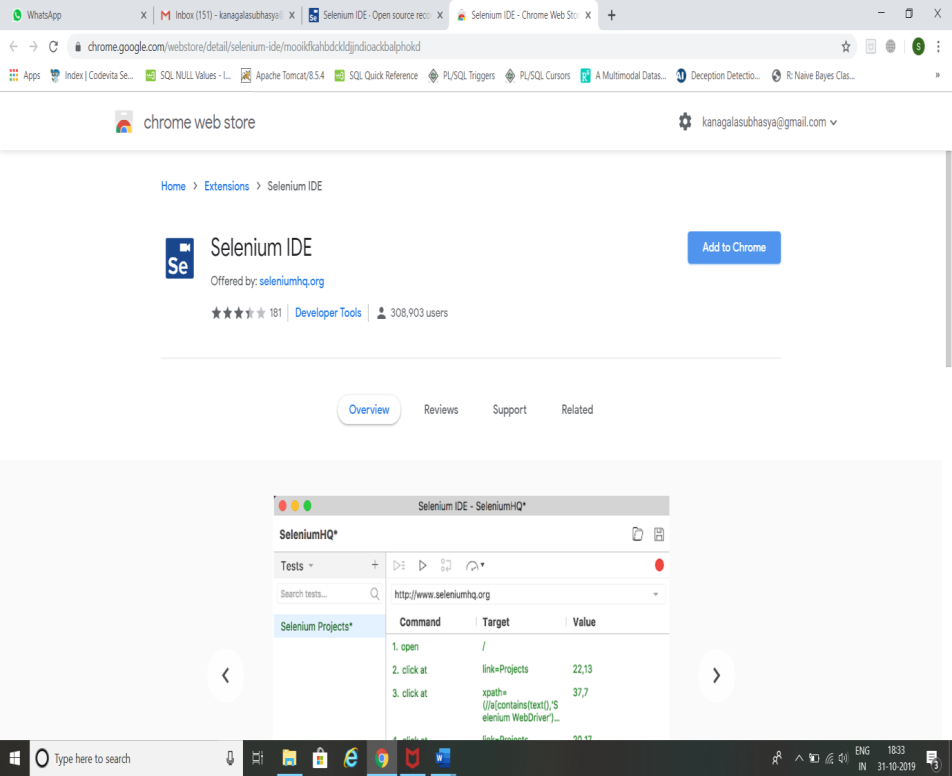
**Procedure:**

**Steps:**

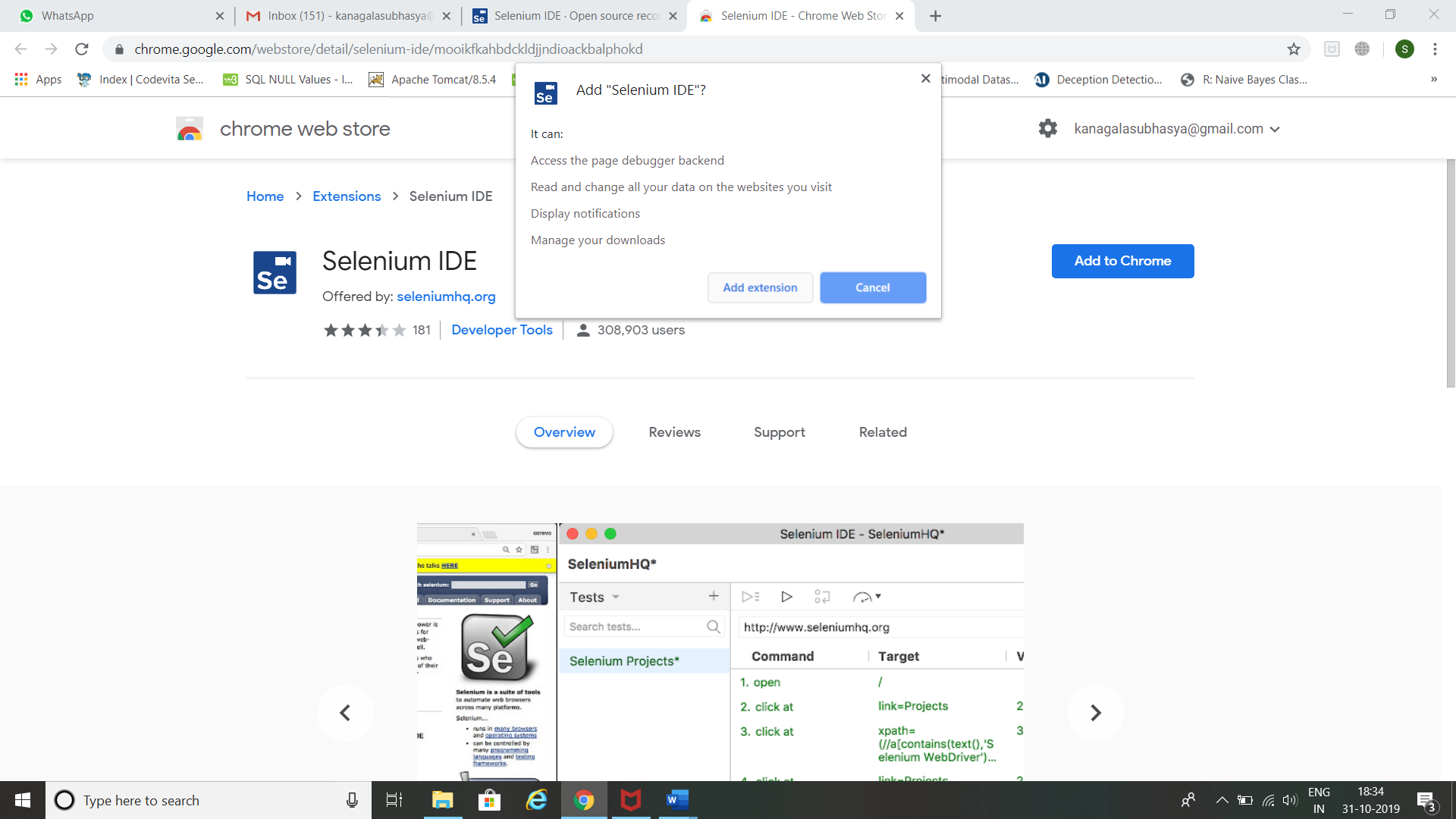
* Open Google Chrome 🡪 search for Selenium IDE.



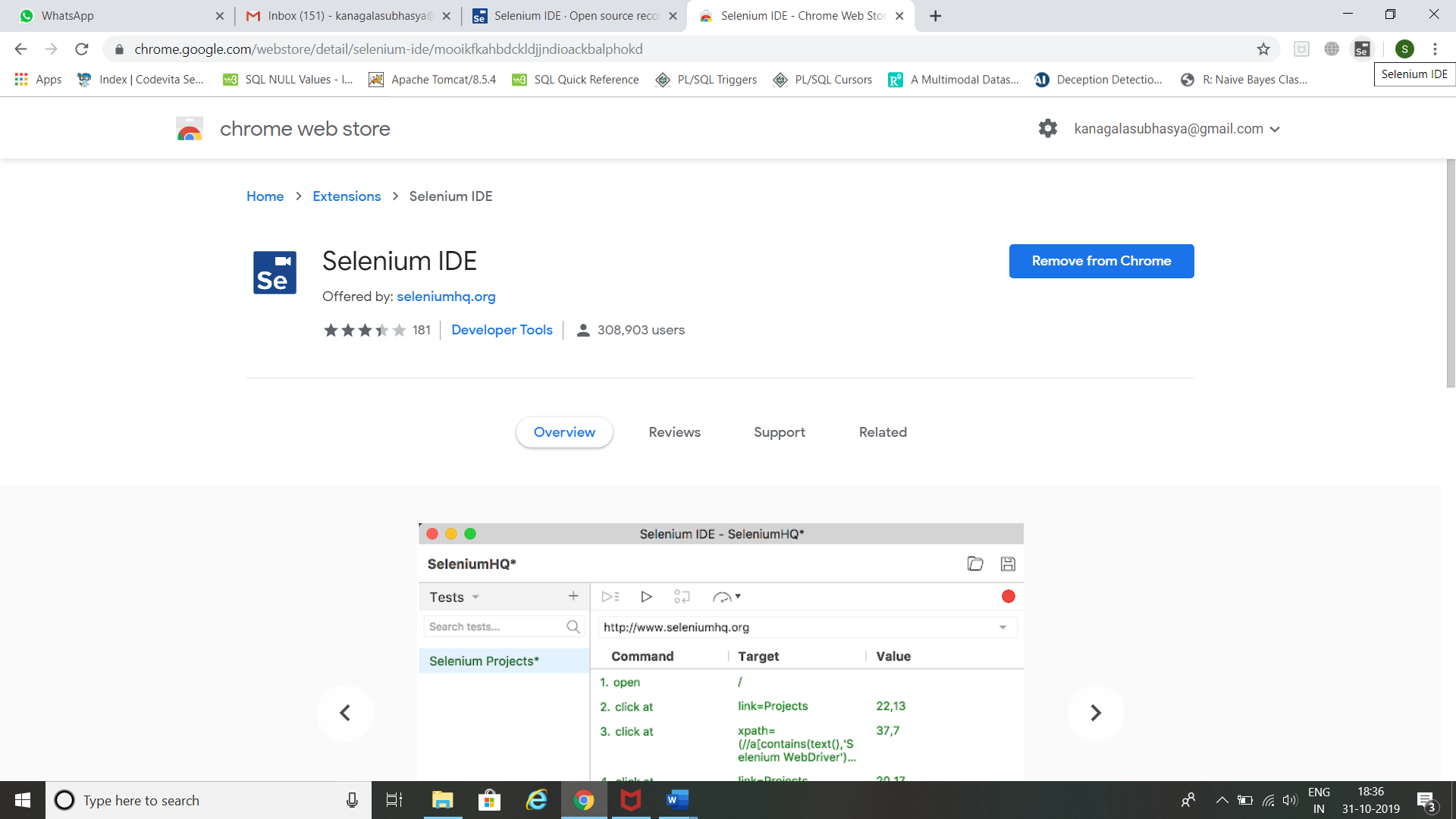
* Click on CHROME DOWNLOAD.
* Click on Add to Chrome.



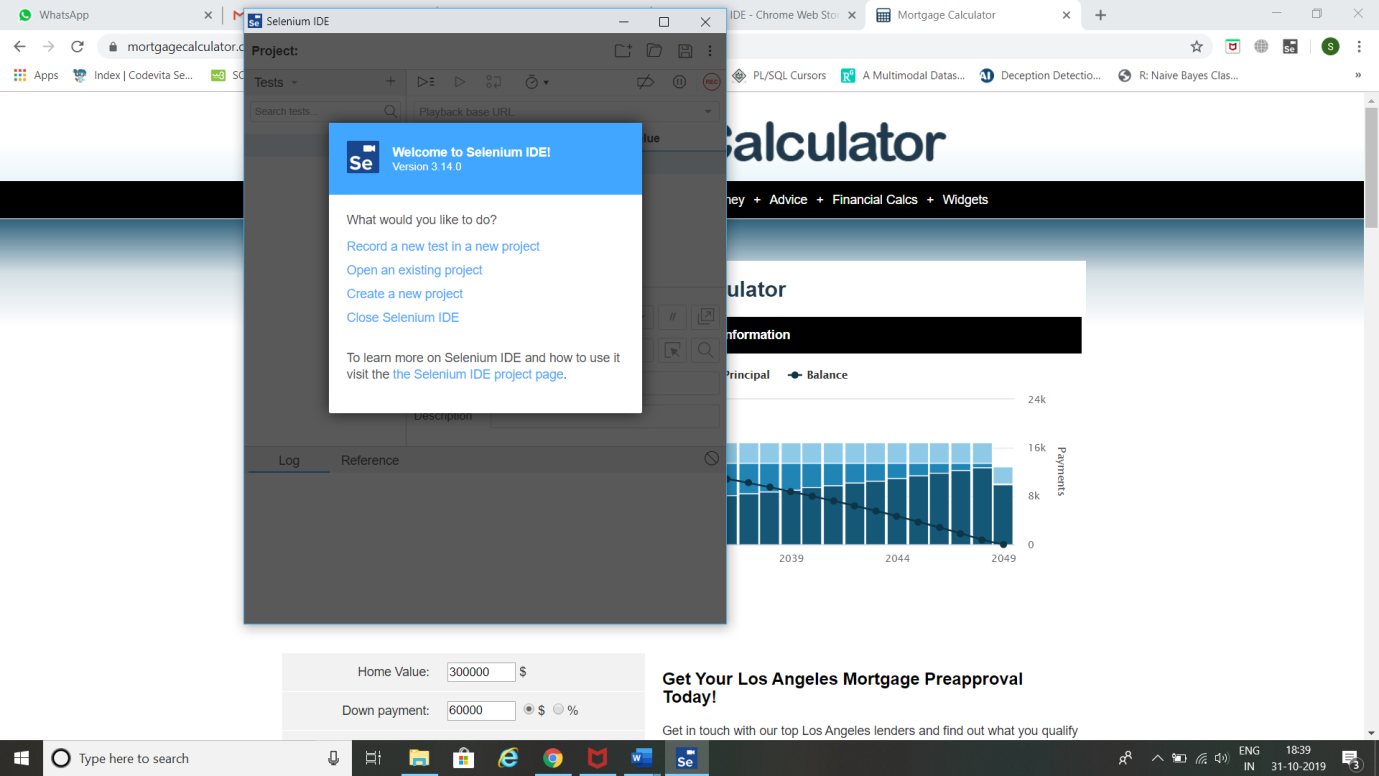
* Click on Add Extension.



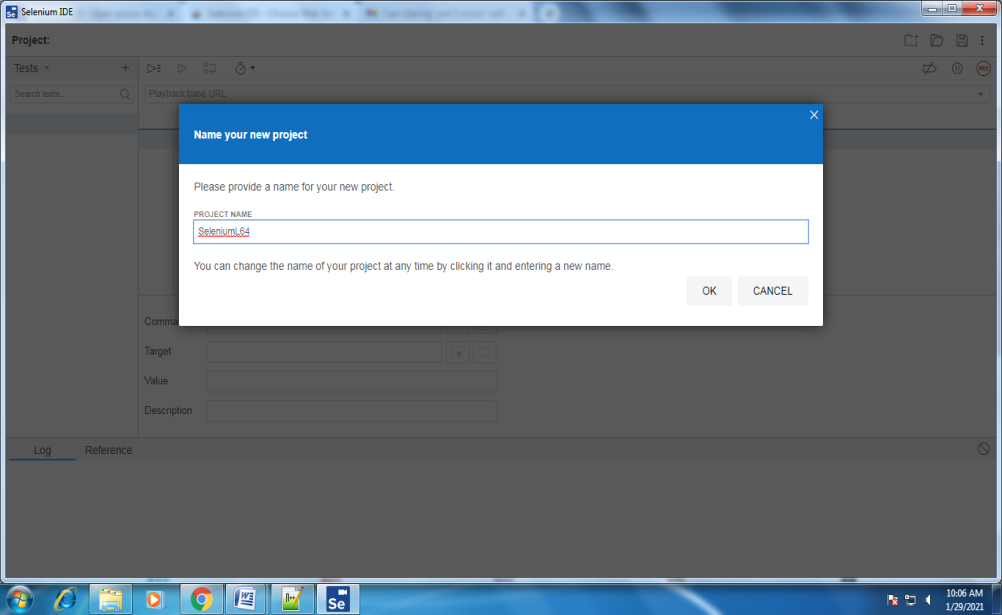
* Click on Selenium IDE Add-on.



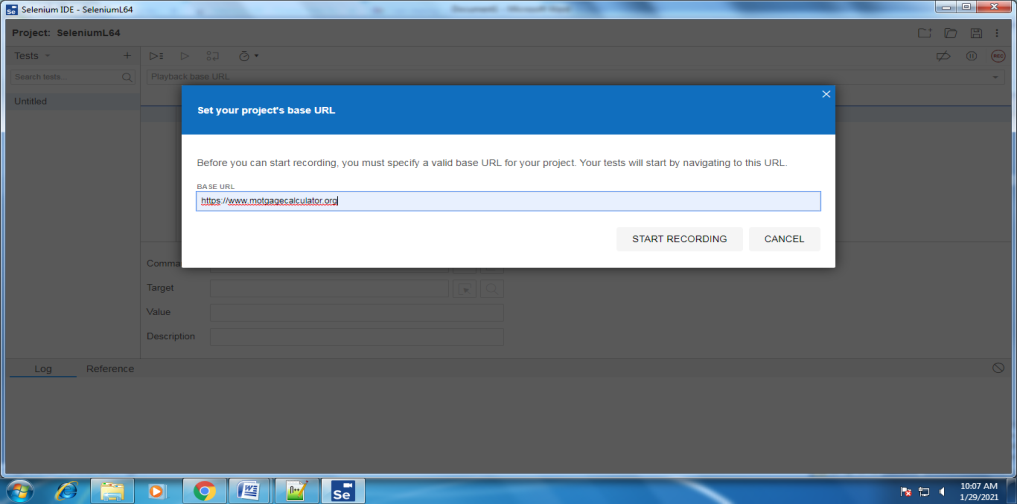
* Click Record a new test in a new project.



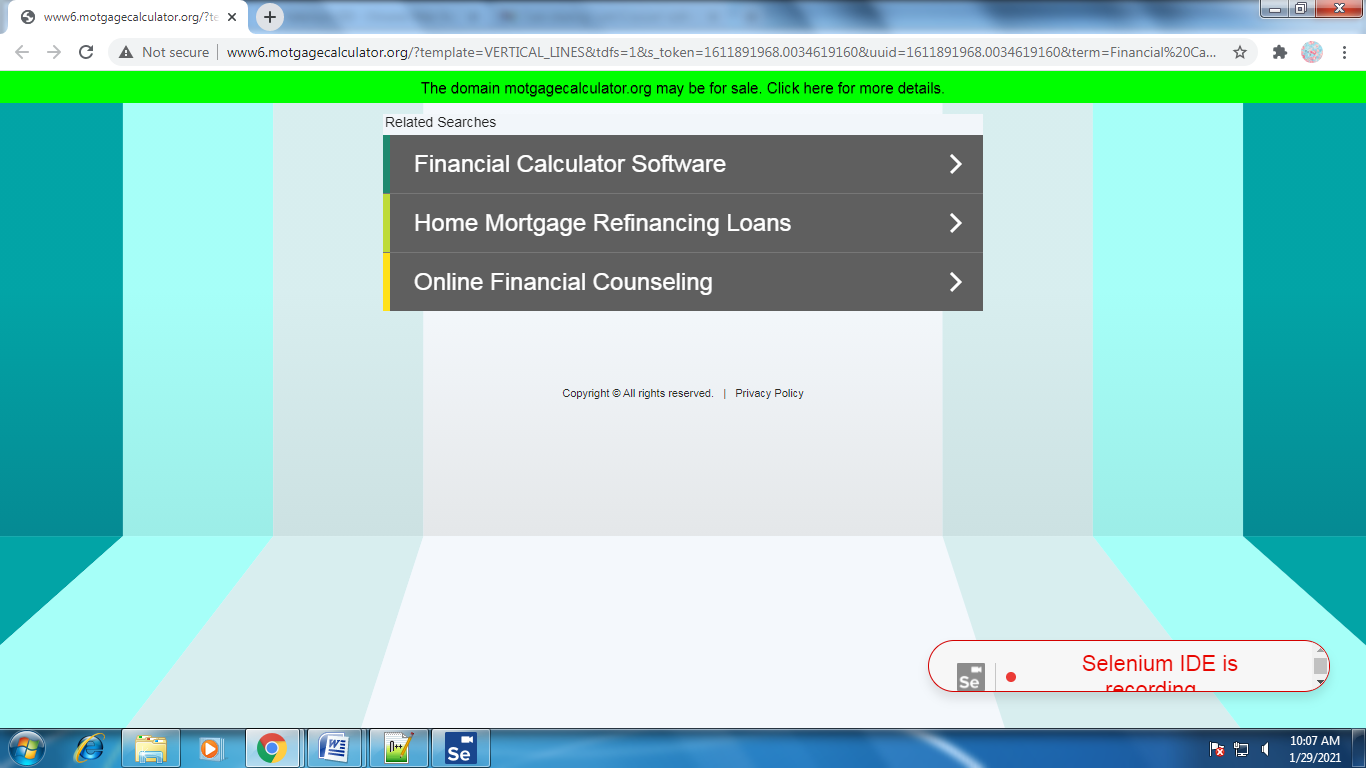
* Enter Project Name and click ok.



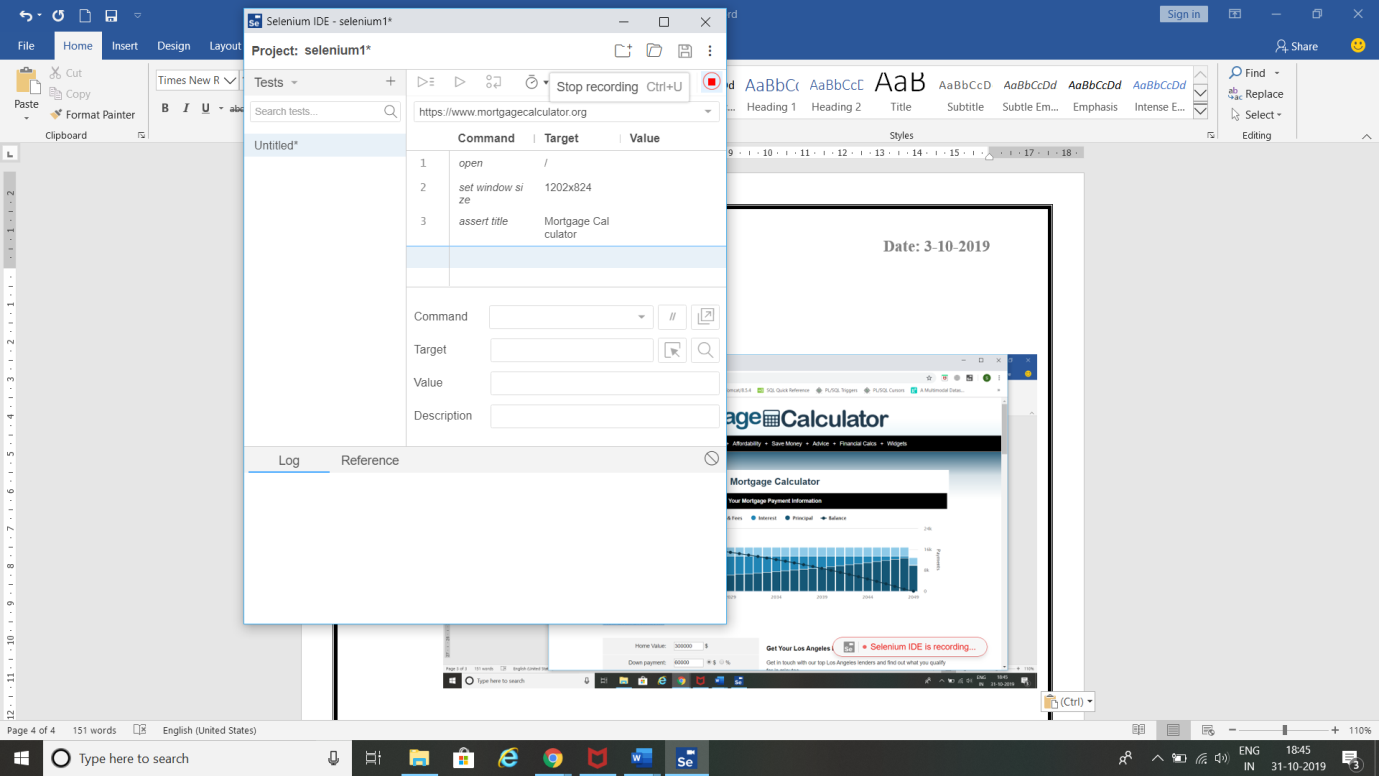
* Enter Base URL for Mortgage Calculator and click START RECORDING.



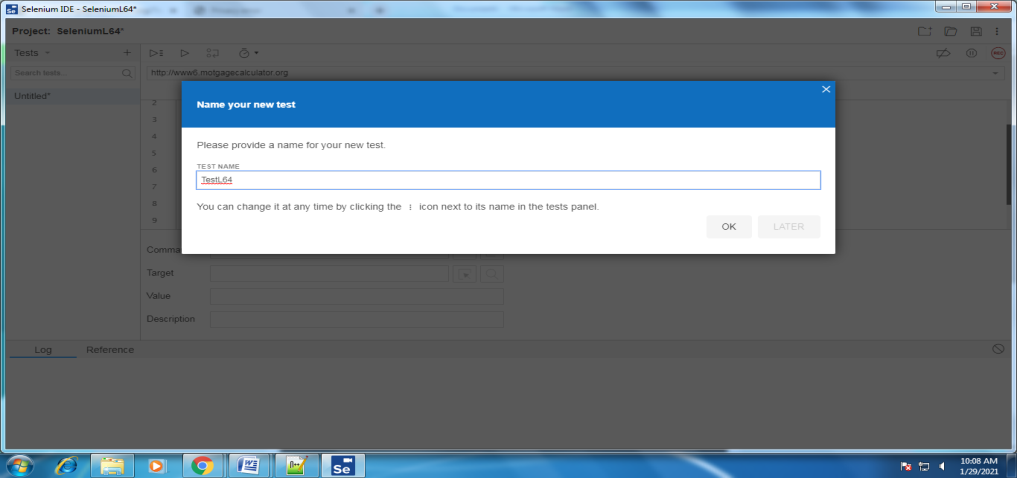
* It opens Mortgage Calculator home page. It will show the message Selenium IDE is Recording. Perform some action (like scrolling down the page, clicking on something etc) in that page for few seconds. This action will be recorded by Selenium IDE.



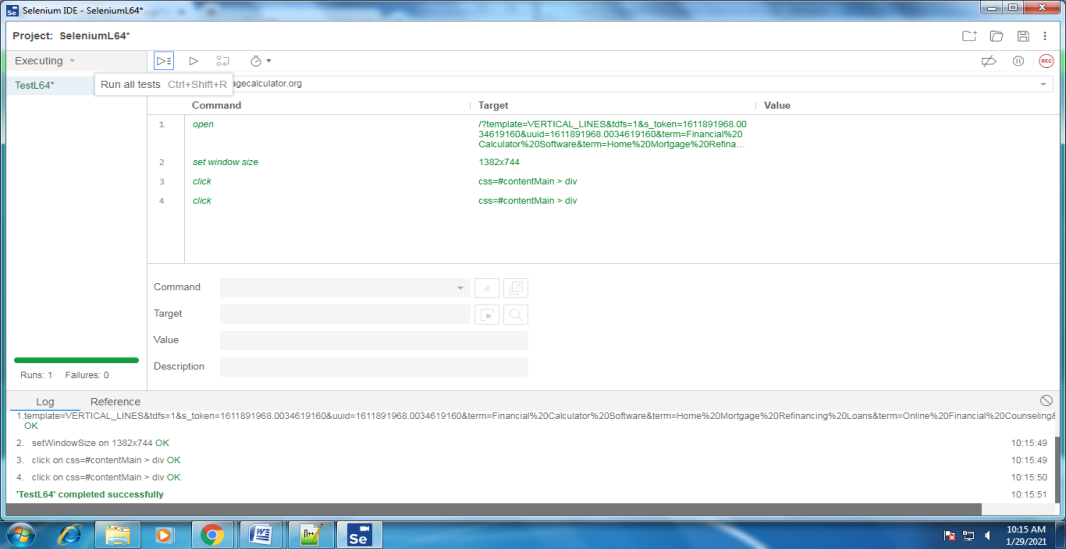
* Now pause the recording by clicking on the pause symbol present at right side of selenium IDE



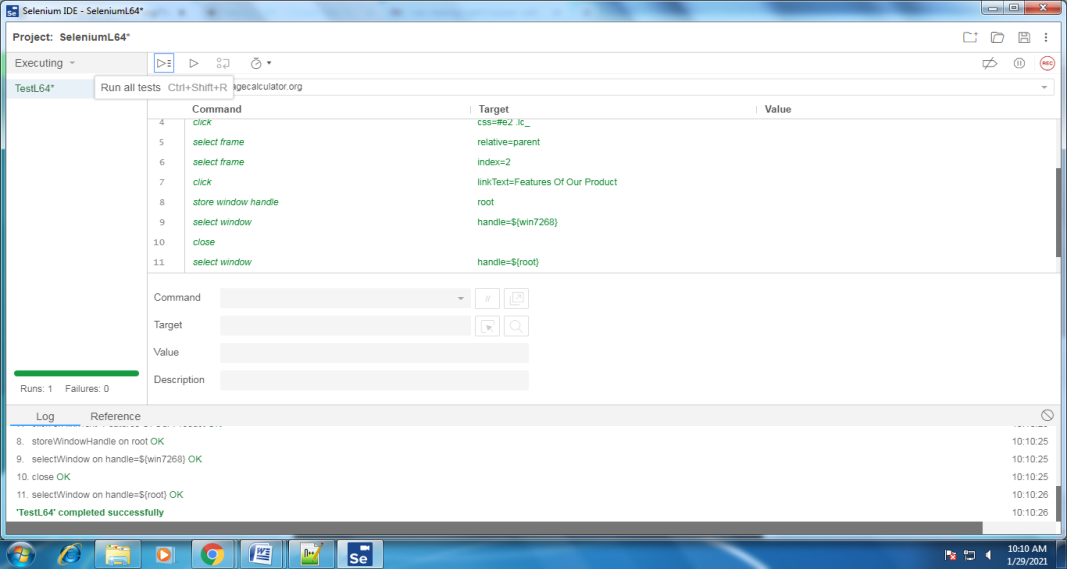
* Enter TEST NAME and click OK.



* Click on Run all Test Cases. It will open Mortgage Calculator and shows the actions performed by you previously.



* It shows message ‘TestL64’ completed successfully’.



* Click save icon 🡪 give name to the file 🡪 click save button 🡪It is saved with .side extension. Right click on that .side file 🡪click edit with notepad 🡪Copy the script present in it.This script shows what you have recorded. This script is shown as follows.

{  
  "id": "2964990a-31e9-4573-87a9-4f9a7f584840",  
  "version": "2.0",  
  "name": "Kundana",  
  "url": "[https://www.mortgagecalculator.org](https://www.mortgagecalculator.org/)",  
  "tests": [{  
    "id": "11ccb133-e943-44da-8ada-9c6b8dfc3f8d",  
    "name": "SeleniumTest",  
    "commands": [{  
      "id": "86ec6927-2570-43c4-aaa4-1f99c78c295d",  
      "comment": "",  
      "command": "open",  
      "target": "/",  
      "targets": [],  
      "value": ""  
    }, {  
      "id": "33440f8c-7057-4ed9-8ee6-f80f1a03ff79",  
      "comment": "",  
      "command": "setWindowSize",  
      "target": "683x728",  
      "targets": [],  
      "value": ""  
    }, {  
      "id": "54167e55-c018-4bb2-a98e-6c8c4d42ebad",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-group:nth-child(12)",  
      "targets": [  
        ["css=.highcharts-series-group:nth-child(12)", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "0ac15e25-b1ff-4830-943f-12cc28ff72c2",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-halo",  
      "targets": [  
        ["css=.highcharts-halo", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "2fef68cd-6f65-455c-b91d-0af8ff94cec4",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-1 > .highcharts-point-hover",  
      "targets": [  
        ["css=.highcharts-series-1 > .highcharts-point-hover", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "af804696-fa28-4481-a102-0223d64cfd89",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-2 > .highcharts-point-hover",  
      "targets": [  
        ["css=.highcharts-series-2 > .highcharts-point-hover", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "0533f86e-fd3f-462f-a2bb-691acd72ab34",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-2 > .highcharts-point-hover",  
      "targets": [  
        ["css=.highcharts-series-2 > .highcharts-point-hover", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "42723f55-6444-46de-9573-a2598028fc21",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-2 > .highcharts-point-hover",  
      "targets": [  
        ["css=.highcharts-series-2 > .highcharts-point-hover", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "2ed144bb-932f-4b69-8a33-0e1890990287",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-2 > .highcharts-point-hover",  
      "targets": [  
        ["css=.highcharts-series-2 > .highcharts-point-hover", "css:finder"]  
      ],  
      "value": ""  
    }, {  
      "id": "98710112-2579-4fb4-8e8b-5448ab80fbe0",  
      "comment": "",  
      "command": "click",  
      "target": "css=.highcharts-series-2 > .highcharts-point-hover",  
      "targets": [  
        ["css=.highcharts-series-2 > .highcharts-point-hover", "css:finder"]  
      ],  
      "value": ""  
    }]  
  }],  
  "suites": [{  
    "id": "4fe1d31c-6951-4fe7-9053-1610d6498806",  
    "name": "Default Suite",  
    "persistSession": false,  
    "parallel": false,  
    "timeout": 300,  
    "tests": ["11ccb133-e943-44da-8ada-9c6b8dfc3f8d"]  
  }],  
  "urls": ["<https://www.mortgagecalculator.org/>"],  
  "plugins": []  
}

**Result:** Thus,performing record play back using Selenium IDE Script has been performed successfully.