**A REPORT ON**

E-Banking Automation Using Selenium and Java

**PROJECT REPORT**

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

**BACHELOR OF TECHNOLOGY**

(Computer Science & Engineering)

**SUBMITTED BY**

**Pranav Duggal 1702765 CSE – 8(Y2)**

**UNDER THE GUIDANCE OF**

**Ms. Amandeep Kaur**

Class Councilor



**Department of Computer Science & Engineering**

**Chandigarh Engineering College**  
Kharar-Banur Highway, Sector-112 Greater Mohali,

Punjab – 140307 (INDIA)

(Approved by AICTE, New Delhi and Affiliated to IKGPTU, Jalandhar)

May 2021

## ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to *Ms. Amandeep Kaur* Dept. of Computer Science & Engineering, CEC-CGC, landran for her generous guidance, help and useful suggestions.

I express my sincere gratitude to Dr. Sukhpreet Kaur, HoD in Department of CSE, CEC-CGC, Landran for her stimulating guidance, continuous encouragement.

I also wish to extend my thanks to my instructors for their insightful comments and constructive suggestions to improve the quality of this research work.

#### **Pranav Duggal**

**Roll No:-1702765**

**Section:-CSE -8(Y2)**

**ABSTRACT**

Automation testing is an emerging field that draws maximum benefits with minimum efforts. It is to increase quality and reliability of software. Selenium is a set of software tools that supports test automation. Selenium is a well-established testing framework and it is used with a large number of browsers. As Selenium supports a variety of programming languages, it becomes easier for the tester to write scripts in his preferred programming language. But the professional must have adequate expertise in the specific language to write test script and for each web page its own test case script needs to be written

Hence, I have designed a generic framework which can be used for automation testing of web elements on the web pages. In our framework we have created a utility file that will contain most of the generic functions needed by the user to operate on the webpage. These functions are reusable and can be used multiple times for to check for the web page functionality.

Here any software user can import this generic framework and perform their own tests. This framework is designed using selenium WebDriver Automation Tool, Java platform on Eclipse editor and Chrome browser. This makes the framework user friendly interface for creating and executing test suites. It is portable and platform independent. It also enables to perform different types of testing. JUnit is used for report generation which gives the output of the test i.e. it tells whether the test has passed or failed.

**Introduction**

According to Wikipedia, **Selenium** is a portable framework for testing web applications. Selenium provides a playback tool for authoring functional tests without the need to learn a test scripting language.

It also provides a test domain-specific language (Selenese) to write tests in several popular programming languages, including C#, Groovy, Java, Perl, PHP, Ruby, and Scala.

**Selenium is an open-source, test automation tool** that has become an important automation tool in the software quality assurance world. This selenium testing tool consists of a different set of tools which include Selenium WebDriver**,**Selenium RC, Selenium IDE, and Selenium Grid, all of which have different features.

Selenium testing tool is a lightweight tool and is developer-friendly, commonly used for automating web applications.

**Test automation** using selenium webdriver with java, automation testing can be used in any operating system environment such as Windows, Linux, and OS X and has been first developed by Jason Huggins in the year 2004.

Cross-browser testing in selenium is an effective selenium testing method used by testers and this tool is also used for [web application testing](https://www.testingxperts.com/blog/5-best-practices-for-web-application-testing). This selenium testing tool is composed of several components that provide different features.

**Project Objective**

In this Project I have used a demo e banking website(demo.guru99.com) This application is based on e banking application that folows the client- server architechure, in which the authenticate and authorised user" can create add the new customers and do the various operations like Create new Bank Account , Depositing Money into the account, Withdrawing money,

Sending Money to another payee account. We can also check the Account balance and check mini statements to check the Amount debited and Amount Credited in Account. This is a Demo Application and is going through the testing phase. I have made some Test Cases

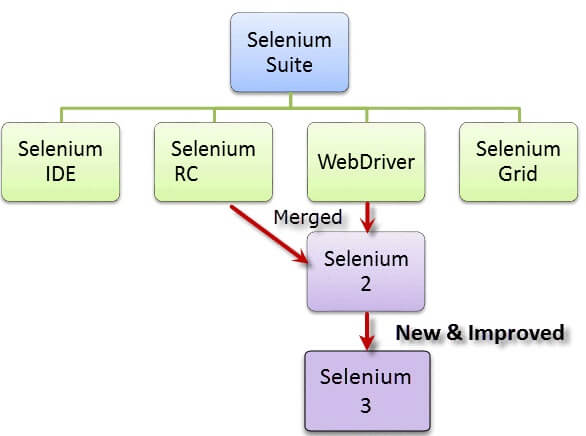
That checks the functionalities of the links and buttons, checks whether the buttons are clickable (Clickablity).I have used Selenium for conducting the testing. Why Selenium?  Selenium has the capability to test dynamic websites, i.e. websites where the content in a page changes dynamically on click of a button or other user functionalities. It works by mimicking a real user who is interacting with a webpage. Moreover, the API(Application Programming Interface) of Web driver is user-friendly which can be understood easily. It can also be used with other testing frameworks like [JUnit to ease automated testing with Selenium](https://www.lambdatest.com/blog/automated-testing-with-junit-and-selenium-for-browser-compatibility/). User friendliness of Selenium WebDriver is one of the widely acclaimed benefits of Selenium WebDriver for automation testing. The tool being open source, it allows users to script their personalized extensions which allows them to develop actions that are customized and can be manipulated once the user reaches an advanced level.

**Tools Used in this Project**

**Selenium:- Selenium** is a free (open-source) automated testing framework used to validate web applications across different browsers and platforms. You can use multiple programming languages like Java, C#, Python etc to create Selenium Test Scripts. Testing done using the Selenium testing tool is usually referred to as Selenium Testing.

Selenium Software is not just a single tool but a suite of software, each piece catering to different Selenium QA testing needs of an organization. Here is the list of tools

* Selenium Integrated Development Environment (IDE)
* Selenium Remote Control (RC)
* WebDriver
* Selenium Grid



**Apache-Maven:- Maven** is a [build automation](https://en.wikipedia.org/wiki/Build_automation) tool used primarily for [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) projects. Maven can also be used to build and manage projects written in [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)), and other languages. The Maven project is hosted by the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation), where it was formerly part of the [Jakarta Project](https://en.wikipedia.org/wiki/Jakarta_Project).

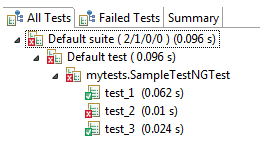
Maven addresses two aspects of building software: how software is [built](https://en.wikipedia.org/wiki/Software_build), and its dependencies. Unlike earlier tools like [Apache Ant](https://en.wikipedia.org/wiki/Apache_Ant), it uses conventions for the build procedure, and only exceptions need to be written down. An [XML](https://en.wikipedia.org/wiki/XML) file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required [plug-ins](https://en.wikipedia.org/wiki/Plug-in_(computing)). It comes with pre-defined targets for performing certain well-defined tasks such as compilation of code and its packaging. Maven dynamically downloads [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache.[[2]](https://en.wikipedia.org/wiki/Apache_Maven#cite_note-maven2repo-2) This local cache of downloaded [artifacts](https://en.wikipedia.org/wiki/Artifact_(software_development)) can also be updated with artifacts created by local projects. Public repositories can also be updated.

**TestNg:- TestNG**is an automation testing framework in which NG stands for "Next Generation". TestNG is inspired from[JUnit](https://www.guru99.com/junit-tutorial.html)which uses the annotations (@). TestNG overcomes the disadvantages of JUnit and is designed to make [end-to-end testing](https://www.guru99.com/end-to-end-testing.html) easy.

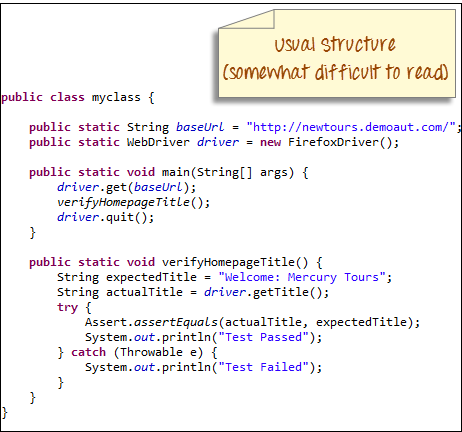
Using TestNG, you can generate a proper report, and you can easily come to know how many test cases are passed, failed, and skipped. You can execute the failed test cases separately.

For example:

* Suppose, you have five test cases, one method is written for each test case (Assume that the program is written using the main method without using testNG). When you run this program first, three methods are executed successfully, and the fourth method is failed. Then correct the errors present in the fourth method, now you want to run only fourth method because first three methods are anyway executed successfully. This is not possible without using TestNG.
* The TestNG in Selenium provides an option, i.e., testng-failed.xml file in test-output folder. If you want to run only failed test cases means you run this XML file. It will execute only failed test cases.
* Generate the report in a proper format including a number of test cases runs, the number of test cases passed, the number of test cases failed, and the number of test cases skipped.
* Multiple test cases can be grouped more easily by converting them into testng.xml file. In which you can make priorities which test case should be executed first.
* The same test case can be executed multiple times without loops just by using keyword called 'invocation count.'
* Using testng, you can execute multiple test cases on multiple browsers, i.e., cross [browser testing](https://www.guru99.com/top-10-cross-browser-testing-tools.html).
* The TestNG framework can be easily integrated with tools like TestNG Maven, Jenkins, etc.
* Annotations used in the testing are very easy to understand ex: @BeforeMethod, @AfterMethod, @BeforeTest, @AfterTest
* WebDriver has no native mechanism for generating reports. TestNG can generate the report in a readable format like the one shown below.

[](https://www.guru99.com/images/report-0057.png)

* TestNG simplifies the way the tests are coded. There is no more need for a static main method in our tests. The sequence of actions is regulated by easy-to-understand annotations that do not require methods to be static.

[](https://www.guru99.com/images/usual_structure-0059.png)

[](https://www.guru99.com/images/testng_structure-0058.png)

* Uncaught exceptions are automatically handled by TestNG without terminating the test prematurely. These exceptions are reported as failed steps in the report.

**Git:-** Git is a [free and open source](http://www.git-scm.com/about/free-and-open-source) distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git relies on the basis of distributed development of a software where more than one developer may have access to the source code of a specific application and can modify changes to it which may be seen by other developers..

Initially designed and developed by [Linus Torvalds](https://en.wikipedia.org/wiki/Linus_Torvalds) for [Linux kernel](https://www.kernel.org/) development in 2005.

Every git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server.

Git allows a team of people to work together, all using the same files. And it helps the team cope up with the confusion that tends to happen when multiple people are editing the same files.

**GitHub:-** Git Hub is a Collaboration platform. It is built on top of git. It allows you to keep both local and remote copies of your project. A project which you can publish it among your team members as they can use it and update it from there itself.

**Advantages of Using Git Hub For Selenium.**

* When multiple people work on the same project they can update project details and inform other team members simultaneously.
* Jenkins can help us to regularly build the project from the remote repository this helps us to keep track of failed builds.

**Project Phases**

Validating every module of software or application is a must to ensure product precision and accuracy. Since software testing itself is an elaborate process, testers carry it out in phases. Complexities can pop up if testing lacks organization. The complexities may include unresolved bugs, undetected regression bugs, or in the worst case, a module that skipped testing because the deadline got closer.

Each phase of this project has a specific goal and deliverables. It involves the initiation, execution, and termination of the testing process.

Let’s take a look at different phases of the software testing life cycle in detail.

1. Requirement Analysis

[Testers](https://www.testim.io/blog/software-tester-qualities/) have to view, study, and analyze the available specifications and requirements. Certain requirements produce outcomes by feeding them with input data. These requirements are testable requirements. Testers study both[functional](https://www.testim.io/blog/automated-functional-testing/) and non-functional requirements. After that, they have to pick out testable requirements.

Activities in this phase include brainstorming for requirement analysis and identifying and prioritizing test requirements. They also include picking out requirements for both[automated](https://www.testim.io/blog/what-is-test-automation/) and manual testing.

There are a few things you have test even if not explicitly mentioned. A click on an active button should do something, a text field for phone number shouldn’t accept alphabets submitted. These things are universal and should always be tested. But in the requirement analysis phase it about knowing more specific details about the product. You need to learn how the product should be in its ideal state.

To sum it up:

* Understand the expected output from the product.
* Identify any loopholes in the specifications.
* Collect priorities.
* Perform automation feasibility checks.

2. Test Planning

The second step is test planning, and the QA team creates this plan after analyzing all the necessary testing requirements. They outline the scope and objectives after understanding the product domain. The team then analyzes the risks involved and defines time schedules and testing environments to create a strategy.

After that, management finalizes the tools and assigns roles and responsibilities to individuals. An approximate timeline is also defined by which the testing of each module should be completed.

To sum it up:

* Prepare test plan documentation.
* Estimate time and efforts.
* Finalize on tools and platform.
* Assign tasks to teams and individuals.
* Identify training requirements

3. Test Case Designing and Development

After development and planning, it’s time to let the creative juices flow! Based on the test plan, testers design and develop test cases. Test cases should be extensive and should cover almost all the possible cases. All the applicable permutations and combinations should be gathered. You can prioritize these test cases by researching which of them are most common or which of them would affect the product the most.

Next comes the verification and validation of specified requirements in the documentation stage. Also, the reviewing, updating, and approval of automation scripts and test cases are essential processes of this stage. This phase also includes defining different test conditions with input data and expected outcomes.

To sum it up:

* Research and gather possible actions on the product.
* Create test cases.
* Prioritize test cases.
* Prepare automated scripts for test cases.

4. Test Environment Setup

Testing activities need certain environmental factors—such as servers, frameworks, hardware, and software—for executing developed test cases. Software and hardware configuration, along with test data setup, are the main components of this phase. And it’s mandatory to smoke test and to equip your testers with bug reporting tools.

In the developer community, it’s common to hear “it ran on my system, but it’s not running on yours”. Hence it is important that your test environment covers all the environments that the user would use.

For example, some feature that works in Google Chrome doesn’t work in Internet Explorer. The working of features also differ based on software and hardware requirements. A feature might work smoothly on 4 GB RAM but might create issues with 1 GB RAM. Research on environments used by end-users would help you prioritize your test environments.

It’s the job of the QA manager supervising the team to take care of setting up the test environment.

To sum it up:

* Understand minimum requirements
* List down software and hardware required for different levels of performance.
* Prioritize test environments
* Setup test environments
* [Smoke test](https://www.guru99.com/smoke-testing.html) the built environments

5. Test Execution

An application is ready for testing once the team is done with all the previous phases. According to the test plan, the testers execute test cases. They also identify, detect, and log the defects, thus reporting the bugs. The team is also responsible for comparing expected results with the real outcome. If any bugs are found, they need to be documented to pass it on to the development team for a fix.

Once the development team removes a bug,[regression testing](https://www.testim.io/blog/automated-regression-testing/) begins. Regression testing is to ensure that the software or application works even after deploying a change. When testing after a bug fix, test the complete product again. Because a fix for a bug could create a bug on some other part of the product. And because the same tests need to be run again and again after every fix and deployment, it’s recommended to use scripts or [automated testing tools](https://www.testim.io/).

To sum it up:

* Run test cases.
* Identify deviation from expected behavior of the product.
* Log failed cases with details
* Test again after bug fixes.

6. Test Closure

And that brings us to the last stage of the STLC: test closure.

The end of test execution and delivery of the end product marks the onset of the test closure phase. The QA team checks the test results and discusses them with other team members. Some other factors they consider are product quality, test coverage, and project cost. If there’s a deviation from estimated values, further analyzes can be done to identify what didn’t go as expected.

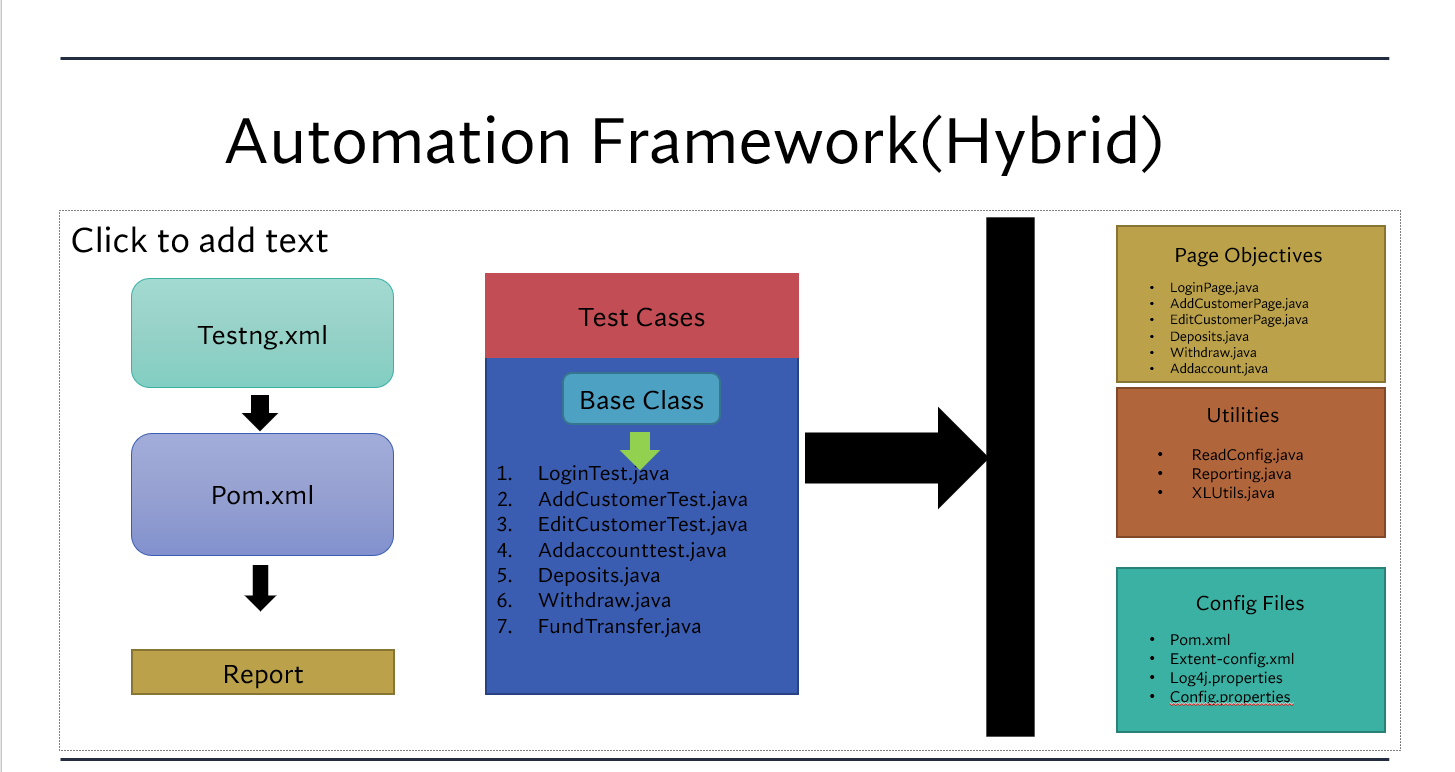
It’s an essential practice for testers to come together and discuss the conclusion after testing. Any issues faced during testing, flaws in strategies can be discussed here. You can also work on coming up with a better approach for testing based on the learnings during testing. If you follow [DevOps](https://devops.com/) or [canary release](https://www.bmc.com/blogs/canary-deployment-release/) practice, testing is frequent. You can decide on how often to send reports and what details to mention while sending reports to different stakeholders.

Apart from that, the team also considers test metrics, the fulfillment of goals, and their adherence to deadlines. Once they have a total grasp on what happened, they can evaluate the entire testing strategy and process.

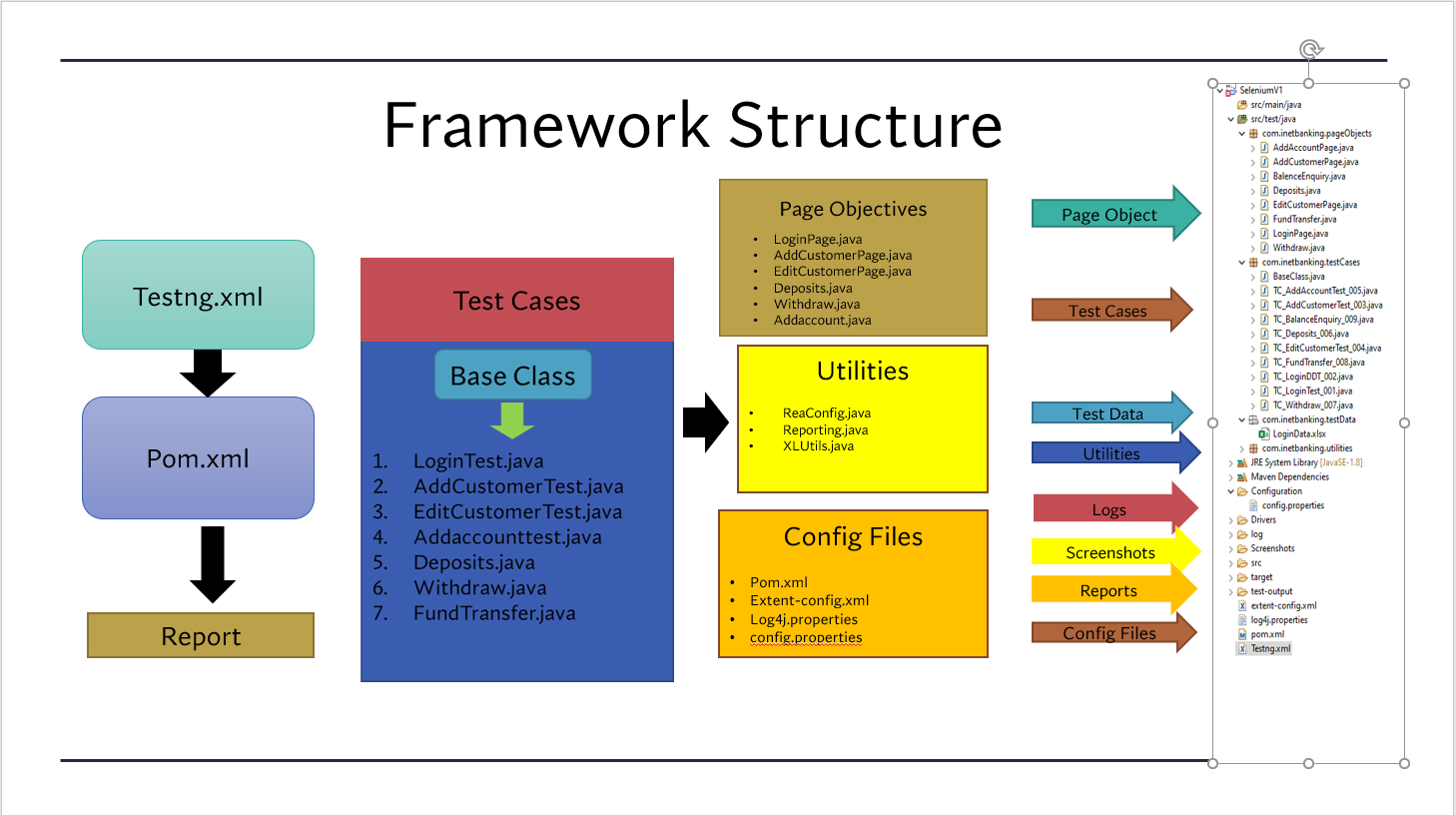
To sum it up:

* Verify that all tests are completed.
* Evaluate factors such as quality, test coverage, timeline, and cost.
* Document the conclusion.
* Discuss the learning and find out if the testing process can be improved.
* Prepare test closure report.

**Automation Framework (Hybrid)**



**Framework Structure**

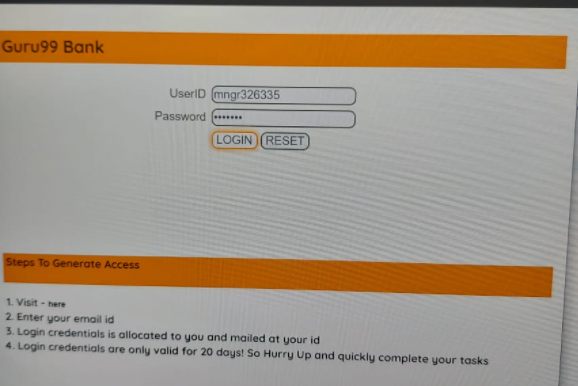


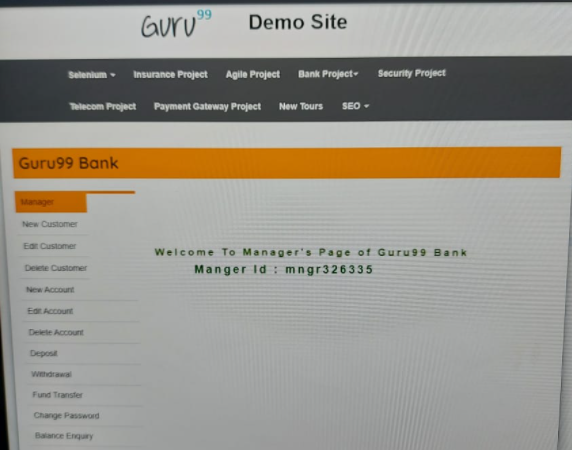
**Source Code and Output of the Projects**

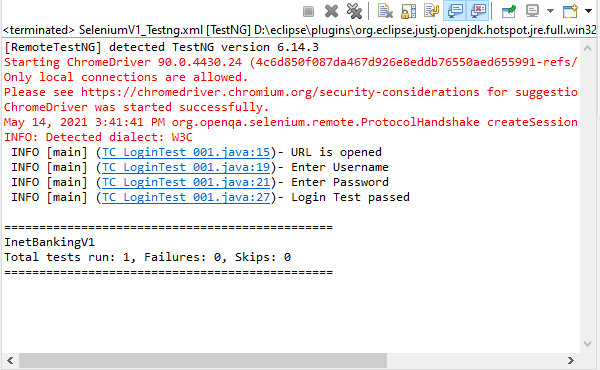
**(With Pictures)**

**Login TestCase:**

**Output:**

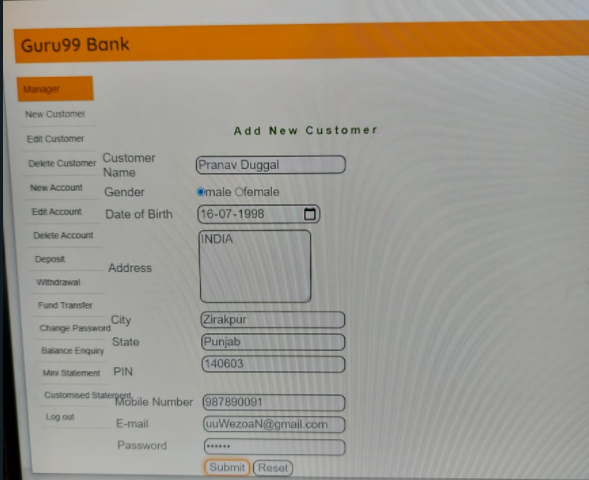


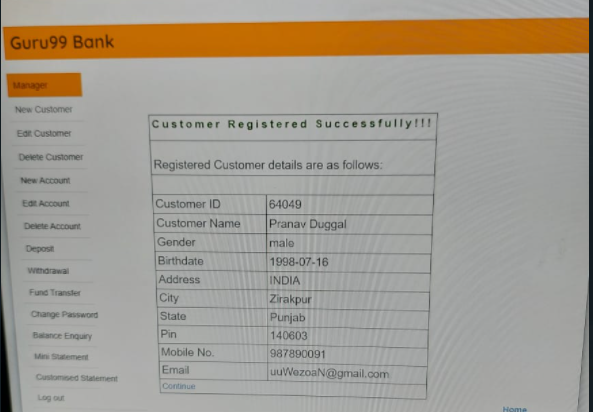




**Adding Customer Information:**

**Output:**

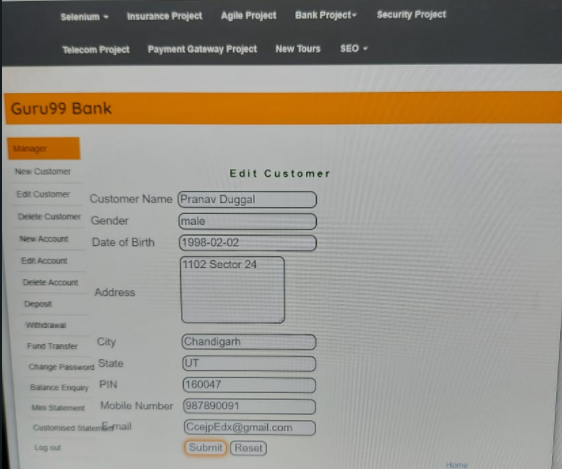


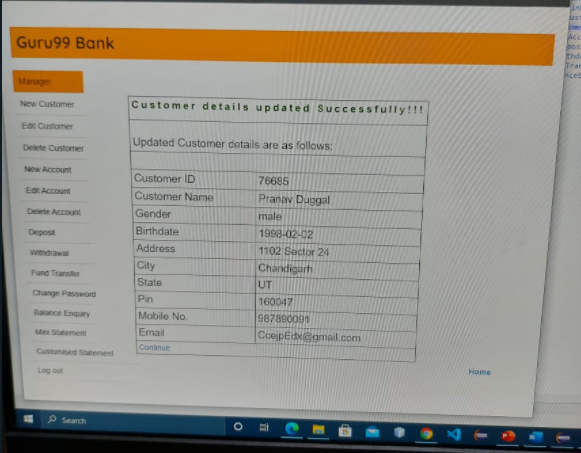


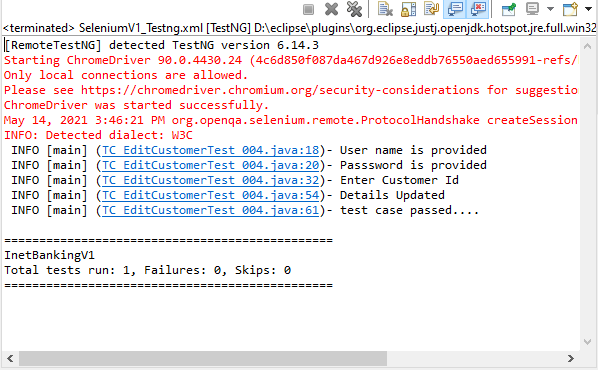


**Edit Customer Information:**

**Output:**

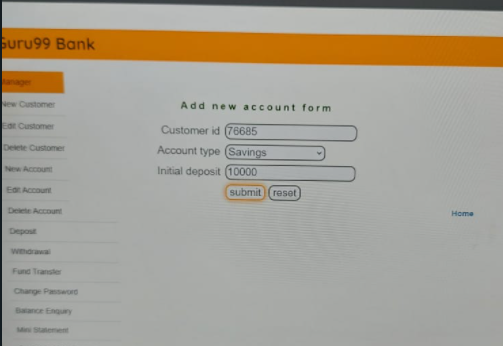


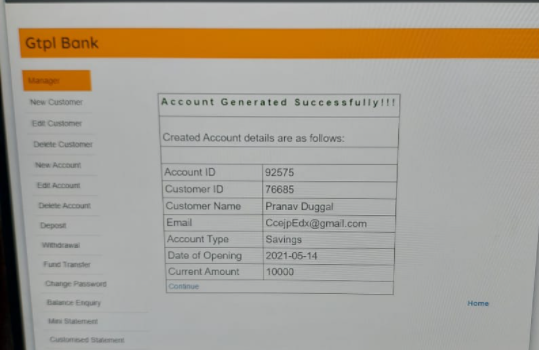


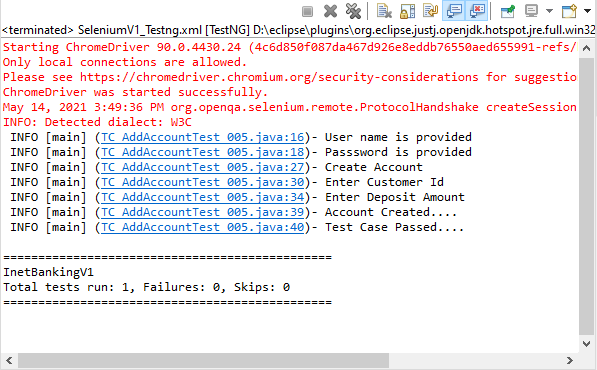


**Creating Customer Bank Account:**

**Output:**

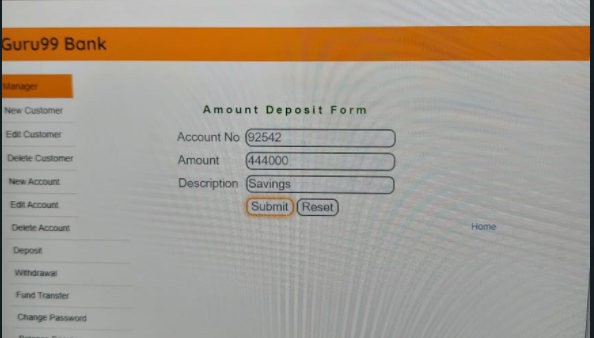


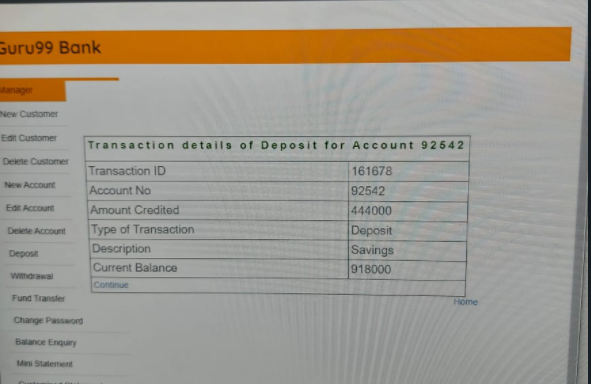


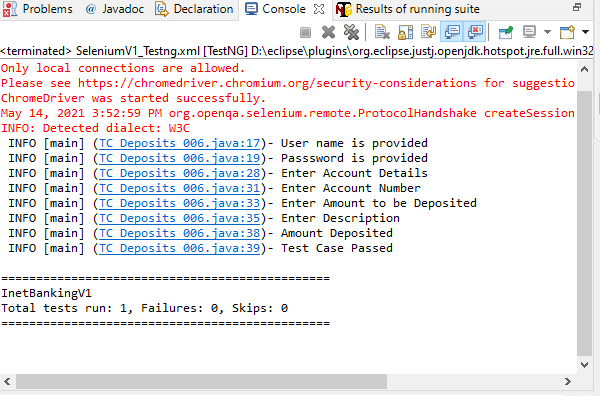


**Depositing Amount in the Account:**

**Output:**

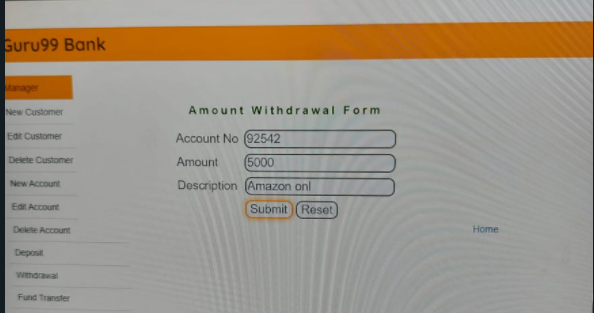


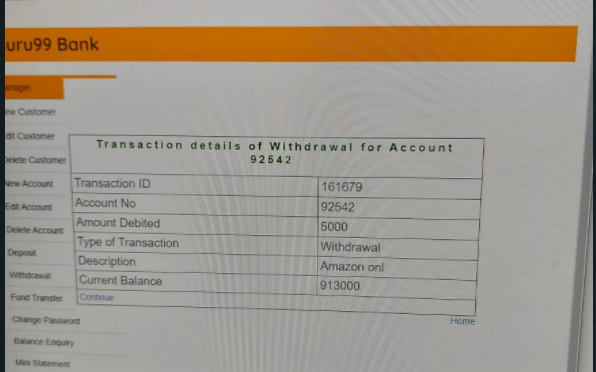


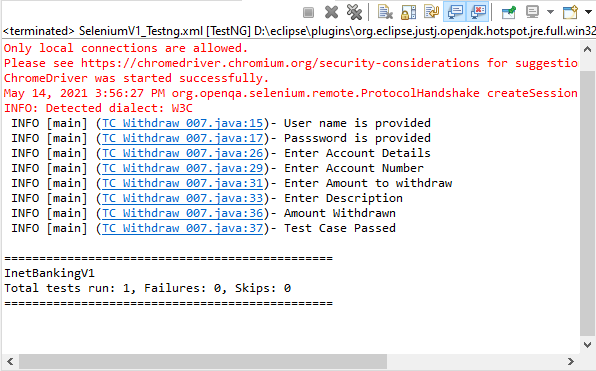


**Withdrawing Amount from Account:**

**Output:**

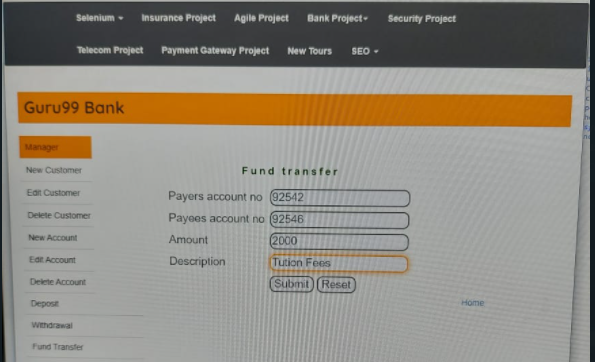




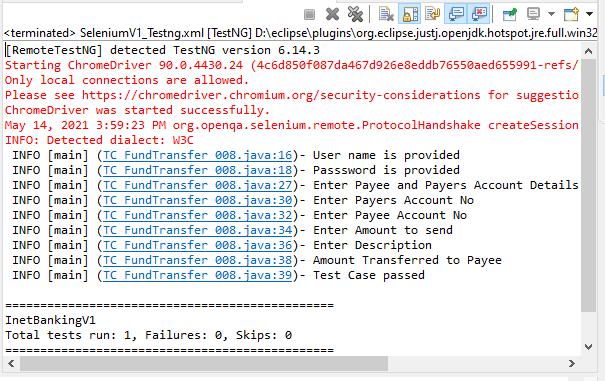


**Transferring From One Account to Another:**

**Output:**

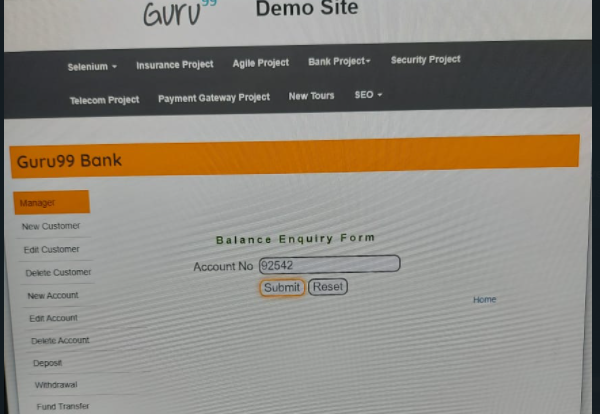


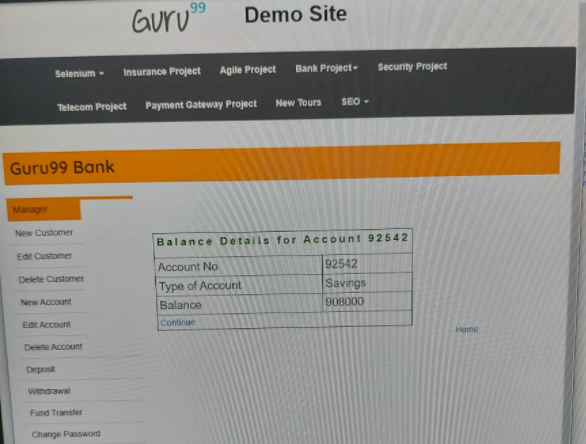


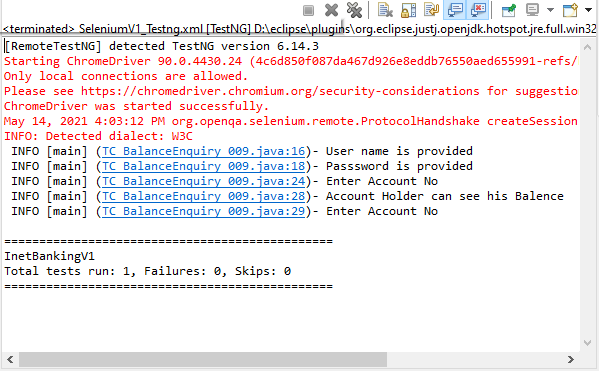


**Checking Balance in Account:**

**Output:**





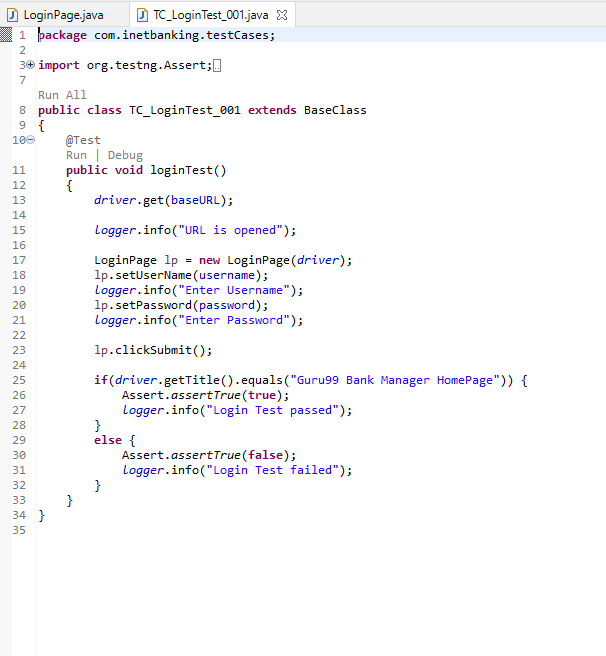


**Source Code**

**(Here TC stands for Test Cases other codes are Page Objects)**

**Login TestCase:**

**Tc\_LoginTest\_001.java**



**LoginPage.java**



**Adding Customer:**

**AddCustomerPage.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class AddCustomerPage {

WebDriver ldriver;

public AddCustomerPage(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[2]/a")

@CacheLookup

WebElement lnkAddNewCustomer;

@FindBy(how = How.NAME, using = "name")

@CacheLookup

WebElement txtCustomerName;

@FindBy(how = How.NAME, using = "rad1")

@CacheLookup

WebElement rdGender;

@CacheLookup

@FindBy(how = How.ID\_OR\_NAME, using = "dob")

WebElement txtdob;

@CacheLookup

@FindBy(how = How.NAME, using = "addr")

WebElement txtaddress;

@CacheLookup

@FindBy(how = How.NAME, using = "city")

WebElement txtcity;

@CacheLookup

@FindBy(how = How.NAME, using = "state")

WebElement txtstate;

@CacheLookup

@FindBy(how = How.NAME, using = "pinno")

WebElement txtpinno;

@CacheLookup

@FindBy(how = How.NAME, using = "telephoneno")

WebElement txttelephoneno;

@CacheLookup

@FindBy(how = How.NAME, using = "emailid")

WebElement txtemailid;

@CacheLookup

@FindBy(how = How.NAME, using = "password")

WebElement txtpassword;

@CacheLookup

@FindBy(how = How.NAME, using = "sub")

WebElement btnSubmit;

public void clickAddNewCustomer() {

lnkAddNewCustomer.click();

}

public void custName(String cname) {

txtCustomerName.sendKeys(cname);

}

public void custgender(String cgender) {

rdGender.click();

}

public void custdob(String mm,String dd,String yy) {

txtdob.sendKeys(mm);

txtdob.sendKeys(dd);

txtdob.sendKeys(yy);

}

public void custaddress(String caddress) {

txtaddress.sendKeys(caddress);

}

public void custcity(String ccity) {

txtcity.sendKeys(ccity);

}

public void custstate(String cstate) {

txtstate.sendKeys(cstate);

}

public void custpinno(String cpinno) {

txtpinno.sendKeys(String.valueOf(cpinno));

}

public void custtelephoneno(String ctelephoneno) {

txttelephoneno.sendKeys(ctelephoneno);

}

public void custemailid(String cemailid) {

txtemailid.sendKeys(cemailid);

}

public void custpassword(String cpassword) {

txtpassword.sendKeys(cpassword);

}

public void custsubmit() {

btnSubmit.click();

}

}

**TC\_AddCustomerTest\_003.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.testng.Assert;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.AddCustomerPage;

import com.inetbanking.pageObjects.LoginPage;

public class TC\_AddCustomerTest\_003 extends BaseClass

{

@Test

public void addNewCustomer() throws InterruptedException, IOException

{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

AddCustomerPage addcust=new AddCustomerPage(driver);

addcust.clickAddNewCustomer();

logger.info("providing customer details....");

addcust.custName("Pranav Duggal");

addcust.custgender("male");

addcust.custdob("16","07","1998");

Thread.sleep(3000);

addcust.custaddress("INDIA");

addcust.custcity("Zirakpur");

addcust.custstate("Punjab");

addcust.custpinno("140603");

addcust.custtelephoneno("987890091");

String email=randomestring()+"@gmail.com";

addcust.custemailid(email);

addcust.custpassword("abcdef");

addcust.custsubmit();

Thread.sleep(3000);

logger.info("validation started....");

boolean res=driver.getPageSource().contains("Customer Registered Successfully!!!");

if(res==true)

{

Assert.assertTrue(true);

logger.info("test case passed....");

}

else

{

logger.info("test case failed....");

captureScreen(driver,"addNewCustomer");

Assert.assertTrue(false);

}

}

}

**Edit Customer:**

**TC\_EditCustomerTest\_004.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.openqa.selenium.By;

import org.testng.Assert;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.EditCustomerPage;

import com.inetbanking.pageObjects.LoginPage;

public class TC\_EditCustomerTest\_004 extends BaseClass {

@Test

public void editCustomer() throws InterruptedException, IOException{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

EditCustomerPage editcust = new EditCustomerPage(driver);

editcust.clickEditCustomer();

editcust.custId(customerId);

logger.info("Enter Customer Id");

editcust.editsubmit();

Thread.sleep(3000);

driver.findElement(By.xpath("/html/body/table/tbody/tr/td/table/tbody/tr[7]/td[2]/textarea")).clear();

editcust.custaddress("1102 Sector 24");

driver.findElement(By.xpath("/html/body/table/tbody/tr/td/table/tbody/tr[8]/td[2]/input")).clear();

editcust.custcity("Chandigarh");

driver.findElement(By.xpath("/html/body/table/tbody/tr/td/table/tbody/tr[9]/td[2]/input")).clear();

editcust.custstate("UT");

driver.findElement(By.xpath("/html/body/table/tbody/tr/td/table/tbody/tr[10]/td[2]/input")).clear();

editcust.custpinno("160047");

driver.findElement(By.xpath("/html/body/table/tbody/tr/td/table/tbody/tr[11]/td[2]/input")).clear();

editcust.custtelephoneno("987890091");

driver.findElement(By.xpath("/html/body/table/tbody/tr/td/table/tbody/tr[12]/td[2]/input")).clear();

String email=randomestring()+"@gmail.com";

editcust.custemailid(email);

editcust.custsubmit();

Thread.sleep(3000);

logger.info("Details Updated");

boolean res=driver.getPageSource().contains("Customer details updated Successfully!!!");

if(res==true)

{

Assert.assertTrue(true);

logger.info("test case passed....");

}

else

{

logger.info("test case failed....");

captureScreen(driver,"editCustomer");

Assert.assertTrue(false);

}

}

}

**EditCustomerPage.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class EditCustomerPage {

WebDriver ldriver;

public EditCustomerPage(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[3]/a")

@CacheLookup

WebElement lnkEditCustomer;

@FindBy(name="cusid")

@CacheLookup

WebElement txtCustomerId;

@CacheLookup

@FindBy(name="AccSubmit")

WebElement btn1Submit;

@CacheLookup

@FindBy(how = How.NAME, using = "addr")

WebElement txtaddress;

@CacheLookup

@FindBy(how = How.NAME, using = "city")

WebElement txtcity;

@CacheLookup

@FindBy(how = How.NAME, using = "state")

WebElement txtstate;

@CacheLookup

@FindBy(how = How.NAME, using = "pinno")

WebElement txtpinno;

@CacheLookup

@FindBy(how = How.NAME, using = "telephoneno")

WebElement txttelephoneno;

@CacheLookup

@FindBy(how = How.NAME, using = "emailid")

WebElement txtemailid;

@CacheLookup

@FindBy(how = How.NAME, using = "password")

WebElement txtpassword;

@CacheLookup

@FindBy(how = How.NAME, using = "sub")

WebElement btnSubmit;

public void clickEditCustomer() {

lnkEditCustomer.click();

}

public void custId(String id) {

txtCustomerId.sendKeys(id);

}

public void editsubmit() {

btn1Submit.click();

}

public void custaddress(String caddress) {

txtaddress.sendKeys(caddress);

}

public void custcity(String ccity) {

txtcity.sendKeys(ccity);

}

public void custstate(String cstate) {

txtstate.sendKeys(cstate);

}

public void custpinno(String cpinno) {

txtpinno.sendKeys(String.valueOf(cpinno));

}

public void custtelephoneno(String ctelephoneno) {

txttelephoneno.sendKeys(ctelephoneno);

}

public void custemailid(String cemailid) {

txtemailid.sendKeys(cemailid);

}

public void custsubmit() {

btnSubmit.click();

}

}

**Creating Bank Account:**

**TC\_AddAccountTest\_005.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.AddAccountPage;

import com.inetbanking.pageObjects.LoginPage;

public class TC\_AddAccountTest\_005 extends BaseClass{

@Test

public void addAccount() throws InterruptedException, IOException{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

AddAccountPage addacc = new AddAccountPage(driver);

addacc.clickAddNewAccount();

logger.info("Create Account");

addacc.custId(customerId);

logger.info("Enter Customer Id");

addacc.accounttype("Savings");

addacc.initialdeposits("10000");

logger.info("Enter Deposit Amount");

addacc.custsubmit();

Thread.sleep(3000);

logger.info("Account Created....");

logger.info("Test Case Passed....");

Thread.sleep(3000);

}

}

**AddAccountPage.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class AddAccountPage {

WebDriver ldriver;

public AddAccountPage(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[5]/a")

@CacheLookup

WebElement lnkAddAccount;

@FindBy(name="cusid")

@CacheLookup

WebElement txtCustomerId;

@CacheLookup

@FindBy(how = How.ID\_OR\_NAME, using = "selaccount")

WebElement selectaccount;

@CacheLookup

@FindBy(how = How.NAME, using = "inideposit")

WebElement initialdeposit;

@CacheLookup

@FindBy(how = How.NAME, using = "button2")

WebElement btnSubmit;

public void clickAddNewAccount() {

lnkAddAccount.click();

}

public void custId(String id) {

txtCustomerId.sendKeys(id);

}

public void accounttype(String atype) {

selectaccount.click();

}

public void initialdeposits(String adeposit) {

initialdeposit.sendKeys(adeposit);

}

public void custsubmit() {

btnSubmit.click();

}

}

**Depositing Money:**

**TC\_Deposits\_006.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.Deposits;

import com.inetbanking.pageObjects.LoginPage;

public class TC\_Deposits\_006 extends BaseClass {

@Test

public void Amountdeposits() throws InterruptedException, IOException{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

Deposits dep = new Deposits(driver);

dep.clickDeposit();

logger.info("Enter Account Details");

dep.Accno("92542");

logger.info("Enter Account Number");

dep.Amount("4440000");

logger.info("Enter Amount to be Deposited");

dep.Desc("Savings");

logger.info("Enter Description");

dep.Accsubmit();

logger.info("Amount Deposited");

logger.info("Test Case Passed");

Thread.sleep(3000);

}

}

**Deposits.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class Deposits {

WebDriver ldriver;

public Deposits(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[8]/a")

@CacheLookup

WebElement lnkDeposits;

@CacheLookup

@FindBy(how = How.NAME, using = "accountno")

WebElement Accountnumber;

@CacheLookup

@FindBy(how = How.NAME, using = "ammount")

WebElement depositamount;

@CacheLookup

@FindBy(how = How.NAME, using = "desc")

WebElement Description;

@CacheLookup

@FindBy(how = How.NAME, using = "AccSubmit")

WebElement btnSubmit;

public void clickDeposit() {

lnkDeposits.click();

}

public void Accno(String ano) {

Accountnumber.sendKeys(ano);

}

public void Amount(String Depositamount) {

depositamount.sendKeys(Depositamount);

}

public void Desc(String des) {

Description.sendKeys(des);

}

public void Accsubmit() {

btnSubmit.click();

}

}

**Withdrawing Money:**

**TC\_Withdraw\_007.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.LoginPage;

import com.inetbanking.pageObjects.Withdraw;

public class TC\_Withdraw\_007 extends BaseClass{

@Test

public void Amountwithdraw() throws InterruptedException, IOException{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

Withdraw with = new Withdraw(driver);

with.clickwithdraw();

logger.info("Enter Account Details");

with.Accno("92542");

logger.info("Enter Account Number");

with.Amount("5000");

logger.info("Enter Amount to withdraw");

with.Desc("Amazon online shopping Fee");

logger.info("Enter Description");

with.Accsubmit();

logger.info("Amount Withdrawn");

logger.info("Test Case Passed");

Thread.sleep(3000);

}

}

**Withdraw.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class Withdraw {

WebDriver ldriver;

public Withdraw(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[9]/a")

@CacheLookup

WebElement lnkWithdraw;

@CacheLookup

@FindBy(how = How.NAME, using = "accountno")

WebElement Accountnumber;

@CacheLookup

@FindBy(how = How.NAME, using = "ammount")

WebElement withdrawamount;

@CacheLookup

@FindBy(how = How.NAME, using = "desc")

WebElement Description;

@CacheLookup

@FindBy(how = How.NAME, using = "AccSubmit")

WebElement btnSubmit;

public void clickwithdraw() {

lnkWithdraw.click();

}

public void Accno(String ano) {

Accountnumber.sendKeys(ano);

}

public void Amount(String Withdrawamount) {

withdrawamount.sendKeys(Withdrawamount);

}

public void Desc(String des) {

Description.sendKeys(des);

}

public void Accsubmit() {

btnSubmit.click();

}

}

**Fund Transfer:**

**TC\_FundTransfer\_008.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.FundTransfer;

import com.inetbanking.pageObjects.LoginPage;

public class TC\_FundTransfer\_008 extends BaseClass{

@Test

public void Fundtransfer() throws InterruptedException, IOException{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

FundTransfer ft = new FundTransfer(driver);

ft.clickfundtransfer();

logger.info("Enter Payee and Payers Account Details");

ft.Payersaccno("92542");

logger.info("Enter Payers Account No");

ft.Payeeaccno("92546");

logger.info("Enter Payee Account No");

ft.Transferamount("2000");

logger.info("Enter Amount to send");

ft.Description("Tution Fees");

logger.info("Enter Description");

logger.info("Amount Transferred to Payee");

logger.info("Test Case passed");

Thread.sleep(3000);

}

}

**FundTransfer.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class FundTransfer {

WebDriver ldriver;

public FundTransfer(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[10]/a")

@CacheLookup

WebElement lnkFundtransfer;

@CacheLookup

@FindBy(how = How.NAME, using = "payersaccount")

WebElement Payersaccountnumber;

@CacheLookup

@FindBy(how = How.NAME, using = "payeeaccount")

WebElement Payeeaccountnumber;

@CacheLookup

@FindBy(how = How.NAME, using = "ammount")

WebElement Amount;

@CacheLookup

@FindBy(how = How.NAME, using = "desc")

WebElement Description;

@CacheLookup

@FindBy(how = How.NAME, using = "AccSubmit")

WebElement btnSubmit;

public void clickfundtransfer() {

lnkFundtransfer.click();

}

public void Payersaccno(String pano) {

Payersaccountnumber.sendKeys(pano);

}

public void Payeeaccno(String Payeeano) {

Payeeaccountnumber.sendKeys(Payeeano);

}

public void Transferamount(String ta) {

Amount.sendKeys(ta);

}

public void Description(String des) {

Description.sendKeys(des);

}

public void Fundsubmit() {

btnSubmit.click();

}

}

**Balance Check:**

**TC\_BalanceEnquiry\_009.java**

package com.inetbanking.testCases;

import java.io.IOException;

import org.testng.annotations.Test;

import com.inetbanking.pageObjects.LoginPage;

import com.inetbanking.pageObjects.BalenceEnquiry;

public class TC\_BalanceEnquiry\_009 extends BaseClass{

@Test

public void Balenceenquiry() throws InterruptedException, IOException{

LoginPage lp=new LoginPage(driver);

lp.setUserName(username);

logger.info("User name is provided");

lp.setPassword(password);

logger.info("Passsword is provided");

lp.clickSubmit();

Thread.sleep(3000);

BalenceEnquiry be = new BalenceEnquiry(driver);

be.clickBalenceEnquiry();

logger.info("Enter Account No");

be.Custaccno("92542");

be.Balencesubmit();

logger.info("Account Holder can see his Balence");

logger.info("Enter Account No");

Thread.sleep(3000);

}

}

**BalanceEnquiry.java**

package com.inetbanking.pageObjects;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.CacheLookup;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.How;

import org.openqa.selenium.support.PageFactory;

public class BalenceEnquiry {

WebDriver ldriver;

public BalenceEnquiry(WebDriver rdriver) {

ldriver=rdriver;

PageFactory.initElements(rdriver, this);

}

@FindBy(how = How.XPATH, using ="/html/body/div[3]/div/ul/li[12]/a")

@CacheLookup

WebElement lnkBalenceEnquiry;

@CacheLookup

@FindBy(how = How.NAME, using = "accountno")

WebElement accountnumber;

@CacheLookup

@FindBy(how = How.NAME, using = "AccSubmit")

WebElement btnSubmit;

public void clickBalenceEnquiry() {

lnkBalenceEnquiry.click();

}

public void Custaccno(String ano) {

accountnumber.sendKeys(ano);

}

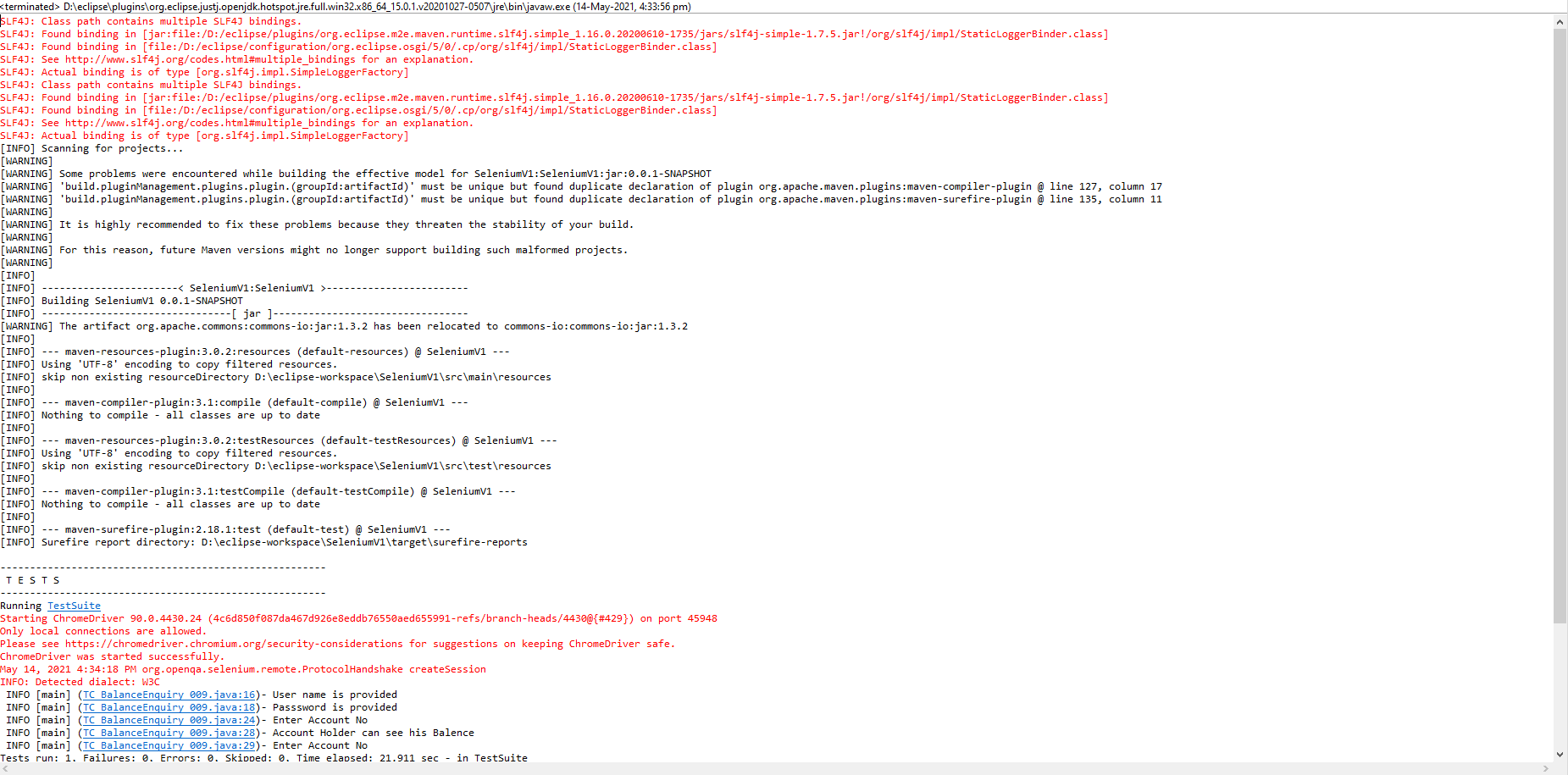
public void Balencesubmit() {

btnSubmit.click();

}}

**Executions**

**Running test cases with Maven pom.xml**



**Running test cases on Command Prompt**

