RAJA SHREE SHIVRAY PRATISHTHAN’S  
MAHARASHTRA COLLEGE OF SCIENCE AND COMMERCE  
KOTHRUD, PUNE-38

**CERTIFICATE**

This is to certify that,

Mr. **AKSHAY PRADIP SHETE**

Has successfully completed the project titled

“**TESTING FUNDAMENTALS**”

As a part of SAVITRIBAI PHULE PUNE UNIVERSITY

Course as per following details.

M.Sc.(Computer Science) – IV

**Project Guide Principal**

**Internal Examiner Industrial Expert External Examiner**

***ACKNOWLEDGEMENT***

I would like to express my heart full gratitude towards those who guided us and played most important role in the successful development of this project called **“TESTING FUNDAMENTALS”** and without this, project would have never been successful.

My sincere thanks to our project guide **“Prof. Mrs. Vidya Dole”** who with almost patience and perseverance guided us willingly towards the project.

We are also thankful to all the teaching and nonteaching staff of **“Computer Science Department”** for guiding us at every step to complete our project successfully.

**THANK YOU VERY MUCH…**

***SCOPE OF THE PROJECT***

Testing Fundamentals covers all the topics related to testing so the end user or document reader can get an easy idea about Testing Concepts and its fundamentals.

Our aim is to give an detailed idea and information about each and every part of testing from basics to advance. So anyone become expert in Testing Environment.

It covers every topic, definition of testing and its types. As well as it contains information about Test Data, Test Cases, Test Scripts etc.

It has wide scope and best for a new comer.

***TESTING CONCEPTS***

**What Is Testing** :-

Software Testing is a program conducted to provide information to the stakeholders, clients and the people associated with it with the information about the quality of software product or service under test known as Application Under Test (AUT).

It mainly include various parts of testing an application through functionally and non-functionally. Testing help us to meet following criteria under AUT.

* It should meets the requirement that guided its design and development.
* Responds correctly to all kind of inputs as well as performs its function within an acceptable time.
* It is sufficiently usable and must achieve the general result its stakeholders and clients desire.
* Testing also help us to check the quality of software developed.

**Is Testing Required** :-

It is most important question for the people “Is Testing Required ?”. Now software is a very huge part of our day to day life. Our life and every day to day activity is linked or we can say that it is associated with many of the softwares.

So for developer it is most mandatory task to make a software of best quality and purpose and which become more easy for end users to operate. But, if we think with developer’s perspective it is not easy task for him that developed an software with all business requirements from clients, stakeholders and so on with simultaneously testing it with user’s perspective.

It is not possible to the same person or same team to develop an application and check whether it meets quality standards or not with testing it. It is difficult to test the entire application functionally and non-functionally to them as lot of factors would come while doing this task such as Cost, Time, Pressure etc.

So to check proper behaviour, quality standard and other functional and non-functional issue we requires Testing of an application. Here testing come into picture and it helps us a lot as we said earlier with benefits.

**Software Requirement :**-

Software requirement state the various requirements of software from the stakeholders and client perspective. It should have following characteristics which would enable a developer or tester to understand the client’s requirement with more clarity.

CHARACTERISTICS OF SOFTWARE REQUIREMENT

Complete

Non-Redundant

Unambiguous

Testable

1. **Unambiguous** :-

A good requirement should be stated in such a way that multiple must interpret the same meaning out of it. The requirement must be expressed in simple, straight forward way with clarity.

Example:-

**Given Requirement** :- The system shall not accept passwords longer than 25 characters.

This is considered as an ambiguous requirement since it does not clearly state the behavior of the system in case the input value of password violates the length constraint. Multiple readers might arrive at interpretations like :-

1. The system shall not let the user enter more than 25 characters.
2. The system shall truncate the entered string to 25 characters.
3. The system shall display an error message if the user enters more than 25 characters.
4. **Complete** :-

Software requirements must contain all the significant needs and conditions expected from the system relating to the functionality, performance etc. it must specify the responses of the system to all varieties of input data.

**Given Requirement :-** On loss of power the battery backup must support normal operations.

The requirement is incomplete since it does not specify for how long the battery backup must support in normal operations. The developer and the tester will not be able to process the requirement without the critical piece of information.

The requirement can be reconstructed as :-

On loss of power the battery backup must support normal operations for 1 Hr.

1. **Testable** :-

Software requirement is said to be testable if there exists some finite cost-effective process that enables a tester to check if this requirement is met in the software.

Example :-

**Given Requirement** :- The system should be user friendly.

User-friendliness is not a computable measure. A tester cannot determine whether software meets this condition unless it is made calculate in terms of measures the GUI components, quality etc.

The requirement can be reconstructed as :-

The user interface should be menu driven. It shall provide dialog bexoes, help screens, radio buttons, drop down lists and input boxes.

1. **Non-Redundant :-**

Software requirements must be stated in such a way that it clearly points to one behaviour or functionality of a system. If more than one requirement points to the same behaviour or functionality, then these requirements become redundant.

Example :-

**Given Requirement 1** :- The system shall not accept only “@” and “.” as special characters for the email id filed in the registration page. If any other special characters used then the system must display an appropriate error message.

**Given Requirement 2** :- All the email id fields in the system must accept only special characters “@” and “.”.

Here both the requirement points to the same functionality so it creates redundancy.

**Testing Techniques :-**

Software quality can be assessed on the basis of six quality factors : Functionality, Reliability, Usability, Maintainability and Portability. These are various techniques available to assess these factors.

1. **Static Testing** :-

Static testing is a testing method that involves analysis of code and documentation without executing the program. Defects identified with static testing are less expensive to fix as compared to defects identified later in development and testing cycle.

The testing involves both developers and testers.

LOW

INFORMAL

LEVEL OF FORMALITY

WALKTHROUGH

HIGH

INSPECTION

PEER REVIEW

**Informal Reviews** :- This technique involves reviewing the document and providing informal comments.

**Technical Reviews** :- This involves reviewing the technical specification documents, standards of the software such as Test Startegy, Test Plan and Requirement Specification documents.

**Walkthrough** :- This involves explanation of product by the developer to the whole team. Review comments are noted down.

**Inspection** :- This is a formal review to find the defects. Reviewers find out the defects and inform the development team to rectify the issue.

1. **Dynamic Testing :-**

Dynamic testing is a testing method involving execution of software and validating the output with the expected result. Black and White box testing are techniques of dynamic testing. It can be performed at all stages of the life cycle.

|  |  |
| --- | --- |
| Static Testing | Dynamic Testing |
| Testing done without executing the program. | Testing done by executing the program. |
| Testing does verification process. | Dynamic testing does validation process. |
| Static testing is about prevention of defects. | Dynamic testing is about finding and fixing the defects. |
| This testing can be performed before compilation. | It is performed after compilation. |
| Static testing involves checklist. | It involves test cases for execution. |
| Cost of finding defects and fixing defect is less. | Cost of finding and fixing the defect is high. |
| More reviews comments are highly recommended for good quality. | More defects are highly recommended for good quality. |

1. **Black Box Testing** :-

Black Box Testing is a testing method which examines the application functionality. It ignores the internal mechanism of the system and concentrates on the generated output. The tester is not aware of the system architecture and has no access to the source code.

The tester is conscious about “**What software is supposed to do but not aware of how the software does it.**”

For example, the tester is aware of what output is generated on giving a particular input, but is not aware about how the software is generating the output.

|  |  |
| --- | --- |
| ADVANTAGES | DISADVANTAGES |
| Well suited and efficient for large code segments. | Limited Coverage. |
| Code access not required. | Inefficient testing. |
| Clearly separates user’s perspective from the developer’s perspective through visibility defined roles. | Bind Coverage. |
| Moderately skilled testers required. | The test cases are difficult to design. |

1. **White Box Testing:-**

White Box Testing is a testing method which analyses the internal structure and working of a program/system. The tester is required to have programming skills and knowledge of internal structure of the program code.

|  |  |
| --- | --- |
| ADVANTAGES | DISADVANTAGES |
| It helps in optimising the code. | Sometime it is not possible to look into every nook and corner to find out errors. |
| Extra lines of codes can be removed. | It is difficult to maintain white box testing as code analysers and debuggers tools are required which is available at high cost. |
| As the tester has knowledge of the source code it become very easy to find out which type of test data required and suite the application. | Due to the fact that a skilled tester is needed to perform white box testing the cost is increased. |

**Types of Testing :-**

1. **Unit Testing :-**

Unit testing is done by developers. It comes under white box testing. It involves development of stubs and drivers. The purpose of unit testing is to separate each part of the program and verify proper functionality of individual modules based on requirements.

1. **Integration Testing :-**

Integration testing tests the functionality of various modules of an application combined together. It is especially applicable to client/server and distributed systems. It also test the interaction between software and hardware. It comes under both white box testing and black box testing. Integration testing helps to expose defects in the interfaces.

1. **Functional Testing :-**

Functional testing is performed on a complete and integrated system to ensure that the specified functionality required in the system works. It comes under black box testing and focuses on testing the application against use cases, design documents and SRS. It test the functionality of each and every section of an application.

1. **System Testing :-**

System testing is performed on a completed integrated system. It comes under black box testing. It is implemented by the dedicated testing team which ensures that the application meets the functional and technical specification along with quality standards.The test environment is quite similar to production environment where the application is supposed to be deployed.

1. **Smoke Testing :-**

Smoke testing is performed on a initial build of a software to make sure that the crucial functionalities of the software are working fine. The main objective is to ensure that major functionalities are working and the build is stable enough to continue with through functional testing.

1. **Sanity Testing :-**

Sanity testing is performed to ensure that the defects have been fixed and no further defects have been introduced due to minor changes in the code or functionality. It is a subset of regression testing. Both smoke and sanity testing help to detect the defects at an early stage to save time and cost involves in rigorous testing.

1. **Regression Testing :**-

Regression testing mainly focuses on detecting such defects in an application and ensures that enhancement in application is functioning properly. It provides consistent and repetitive validation of every new product release. Methods include re-running previously executed tests. Automated execution of tests is preferred in regression testing.

1. **Acceptance Testing :**-

Acceptance testing is performed to ensure the software has met the requirements and performs the way customer expects in production environment. It helps to detecting spelling mistakes, cosmetic errors and severe defects that may lead to major errors in an application. It should agree to legal and contractual requirements. Alpha Testing and Beta Testing are the sub types of Acceptance Testing.

1. **Performance Testing :-**

Performance testing is performed to determine whether the non-functional requirement like response time, throughput and many more in accordance with service level agreement for the project.

Load Testing, Stress Testing, Endurance Testing, Availability Testing, Volume Testing, Scalability Testing etc.

* **Functional Requirements :-**

These describe the services and functionalities a system or a system component should provide. It also describes the expected behaviour of the system to the various input given at different situations.

Considering the simple example of “XYZ Bank” the example of functional requirements are :-

1. When an invalid input is given for a field UserId, the system should display the error message “Please enter valid UserId” to the user in a popup window.
2. The system should accept only alphabets for the UserId filed. It must not accept more than 25 characters.
3. On entering a valid “UserId” and “Password” and on clicking on the “Login” button the user should navigate to the “My Dashboard” page.

* **Non - Functional Requirements :-**

Some examples of Non-Functional Requirements are given in below Table.

|  |  |
| --- | --- |
| Non-Functional Requirement | Description |
| Performance Requirement | Response Time, Transaction Rates, Throughput, Reliability etc. |
| Operating Constraints | System resources – hardware and software, human resources. |
| Platform Constraints | Target platform. |
| Modifiability | Efforts to make changes in software. |
| Security | System and data protection. |
| Portability | Effort to move software to different platform. |
| Usability | Ease to operate the system developed. |
| Legal | Intellectual property rights, privacy policies etc. |

* **Methods Of Testing :-**

To test an application perfectly and clearly we have various ways available in market. We can select any method as per our business needs and requirement. Testing method can be select by client, stakeholders or sometime BA team also. But it should be useful and efficient to test an entire application.

Methods Of Testing

Automation Testing

AI Testing

Machine Testing

Manual Testing

In this session we have covered only Manual Testing and Automation Testing. We tried to provide full and detailed information about this type of testing techniques as follows :-

1. **Manual Testing :-**

Manual Testing is a process of finding out the defects or bugs in a software program. In this method tester plays an important role of end user and verifies that all the features of the application are working correctly. **The tester manually executes test cases without using any automation tools.** The tester prepares a test plan document which describes the detailed and systematic approach to testing of software applications. Test cases are planned to cover almost 100% of the software application. As manual testing involves complete test cases it is a time consuming test.

**Life Cycle Of Manual Testing :-**

Every testing technique has few steps which are useful while testing an application. Manual testing also have its own life cycle as follows :-

**Test Closure**

**Test Cycle Closure**

**Test Case Execution**

**Test Planning and Design**

**Requirement Capture and Analysis**

**Phase 1: Requirement Capture & Analysis :-**

Requirement analysis is the most significant stage in STLC. During this phase the Quality Assurance (QA) team tries to understand the requirements from testing point of view and identify the testable requirement.

The QA team interacts with various stakeholders like Business Analyst, Development Team, Client, Technical Leads, System Architects, etc. to understand the requirements of the system in detail.

The QA team also analyses the feasibility for test automation in this phase.

**Phase 2: Test Planning and Design :-**

In this phase the QA Manager makes the strategy for the complete testing process. The cost and effort required are estimated for the project. The Test Plan is designed which specifies the scope, approach, resources and schedule of all testing activities. The plan includes the features to be tested the various types of testing to be performed, the roles and responsibilities for personnel involved, the resources and schedule required to complete testing.

In Test Design phase test cases and test scripts are designed, reviewed and reworked upon. Identification of test data and preparation of RTM is also done in this phase. Test environment setup is the process of identifying the software and hardware conditions under which a product would be tested.

Test Plan, Effort Estimation, Test Cases and Scripts, RTM are the important aspects of test design.

**Phase 3: Test Execution :-**

During this phase the test team tests readiness of environment which was set up in the test design phase. The testers also execute the test cases as per the test plan. The defects are reported back to the development team. Once the defects are fixed, the system is tested again to ensure that the software is defect free and ready to be deployed at the client side.

Test Execution Reports, Defect Report are the important aspects of Test Execution.

**Phase 4: Test Cycle Closure :-**

The testing team meets and evaluates the cycle completion criteria based on time, cost and test coverage.

Cycle Completion Report is the aspect of Test Cycle Closure.

**Phase 5: Test Closure :-**

This is the last phase of testing cycle. In this phase the testing process used for the project is evaluated and lessons learnt from the process are documented. The main focus here is to identify the best process can be implemented in the future.

Project Learning Documents, Case Study, Metrics Report are the most important aspects of Test Closure.

**Functional Techniques :-**

1. **Equivalence Partitioning :-**

The whole range of input is split into set of equivalence classes, such that a single value acts as a sample for each equivalence class. Exhaustive testing is not required in this case.

Example 1:-

A program needs to be tested with input range between 1000 and 2000. This can be divided into three equivalence classes as follows :-

* < 1000
* Between 1000 and 2000
* 2000

Any value can be picked up in each of the equivalence class and can be used for testing.

1. **Boundary Value Analysis :-**

This technique consists of developing test cases and data that focus on the input and output boundaries of a given function as these are more prone to errors.

Example 1 :-

A program needs to be tested with input range between 1000 and 2000. Boundary analysis will have the following boundary values to be tested.

* Lower Boundary (999 and 1001)
* Upper Boundary (1999 and 2001)
* On the Boundary (1000 and 2000)

1. **Orthogonal Array Testing Strategy (OATS) :-**

OATS is a systematic statistical approach to test pair-wise interactions. The testing technique provides uniform coverage of test that cover all variable pair combinations thereby reducing the number of testes that are likely to find defects. It saves testing effort and particularly useful at integration test level.

**Structural Techniques :-**

Structural technique is mostly used for White Box Testing. Since white box testing mainly deals with program code, testing is focused on coverage of code. The below testing techniques would help in having more code coverage.

1. **Statement Coverage :-**

Test Cases are designed in such a way that every statement of the code is executed at least once.

1. **Branch Coverage:-**

In Branch testing, test cases are designed to exercise branches or decision points in a unit. Given a structural specification for a unit specifying the control flow within the unit, test cases can be designed to exercise branches.

1. **Path Coverage :-**

The test cases are executed in such a way that every path is executed at least once.

**Test Case Preparation :-**

Test cases are the scenarios which are used to test functionality of an application or a software step by step. It describes the information about scenario which is helpful to test an application.

There are various ways to write test cases. Test cases should be unambiguous, complete, non-redundant so the tester should be easily understand and execute the test case.

Below is the snippet of the test case :-

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BRD No | TC Name | Test Case Description | Precondition | Step 1 | Expected Result | Step 2 | Expected Result | Step 3 | Expected Result |
| CR-745 | CR745\_TC\_01 | This test case is use to check Login-functionality of an application. | User should have access to an application. | Enter valid Username and Password. Click on Submit. | User should be login succesfully and navigate to the main page. | User should be able to see Last Login Time. | Last Login time should be display automatically on screen. | Click on Logout Button. | User should be logout succesfully from an application. |
| CR-745 | CR745-TC-02 | New Customer Id should be created after entering valid customer details. | i) User should have application access.  ii) Valid customer details available. | Click on New Customer Button and enter valid details of customer. | Details should be entered succesfully. | Click on submit button and then Customer ID populated | Valid Customer ID should be populated which is able to use for further process | Click on Logout Button. | User should be Logout from an application succesfully. |

**Test Data :-**

Test data is specifically identified data to perform testing of a software application. Test data generation is limited by factors like cost, time and quality.

Some of the key points about Test Data are :-

**Usage:-**

To verify if a particular set of input data into the application produces an expected result. To challenge the capability of the program to respond to exceptional, extreme, unusual or unexpected input.

**How To Create :-**

Produced manually by the tester or produced automatically using a program.

**How To Use :-**

Entered manually by testers from the excel sheet or read automatically from files (Database, XML, Flat Files) using automation tools.

**Type of Test Data :-**

**Different Type of Test Data**

Based on the business transactions like AUT.

Ex:- Order ID, Payment reference etc.

**Transactional**

These are data that are rarely updated but often read.

Ex:- Customer information like Name, Email, Mobile No etc.

Parameters that drive the application or the system.

Ex:- Licence details, Server Details, User Credentials.

Static Data is a data which never changes throughout the application. It remains constant.

Ex :- Gender of a person.

**Master**

**Configurable**

**Static**

**Software Defect :-**

A software defect is an error or fault in the software that leads to an unexpected behaviour of the system. The defect could be caused by reasons like a miscommunication of requirement, poor coding skills, time pressure etc.

The software defects are categorized based on :-

* Severity :- It is the extent to which a defect can affect the software.
* Priority :- The urgency with which the defect should be addressed.

**Priority**

**Low**

**Urgent**

Feature that is rarely used does not work.

**Critical**

Key feature does not work.

**Severity**

The caption on an image is written in the wrong font.

Company logo is in wrong color.

**Non- Critical**

**Defect Status :-**

|  |  |
| --- | --- |
| Status | Description |
| New | A defect has been logged for the first time. |
| Assigned | The defect has been assigned to the development team for fix. |
| Open | The developer has started analysing and working on the defect. |
| Fixed | The developer makes code changes in order to rectify the defect. |
| Retest | The tester rechecks the code and the system to check if the defect has been removed. |
| Deferred | When the defect is expected to be fixed in the upcoming releases then the status is changed to postponed or deferred. Some of the other factors are low priority of the defect or lack of time for the release or low severity. |
| Duplicate | If two defects refer to the same concept, then it is considered as duplicate. |
| Rejected | If the defect posted by the tester is not genuine the developer rejects the defect. |
| Reopen | If the defect is not fixed after the code change then the tester posts the defect again. |
| Closed | If the defect does not exist in the software anymore then the tester closes the defect. |

**How To Raise Software Defect :-**

We are working on a bank module where we have to create Customer ID after filling all customer details. We have to perform following test case.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BRD No | TC Name | Test Case Description | Precondition | Step 1 | Expected Result | Step 2 | Expected Result | Step 3 | Expected Result |
| CR-745 | CR745-TC-02 | New Customer Id should be created after entering valid customer details. | i) User should have application access.  ii) Valid customer details available. | Click on New Customer Button and enter valid details of customer. | Details should be entered succesfully. | Click on submit button and then Customer ID populated | Valid Customer ID should be populated which is able to use for further process | Click on Logout Button. | User should be Logout from an application succesfully. |

But while creating new customer, Application not able to populate new Customer ID for that customer. So we raised the defect as follows :-

**Defect Description :-**

CR 745 | Customer Module | ST | Customer ID not populated after entering valid customer details.

**Steps To Reproduce :-**

1. Login into application.
2. Click on New Customer button.
3. Enter all the required customer details and click on submit button.

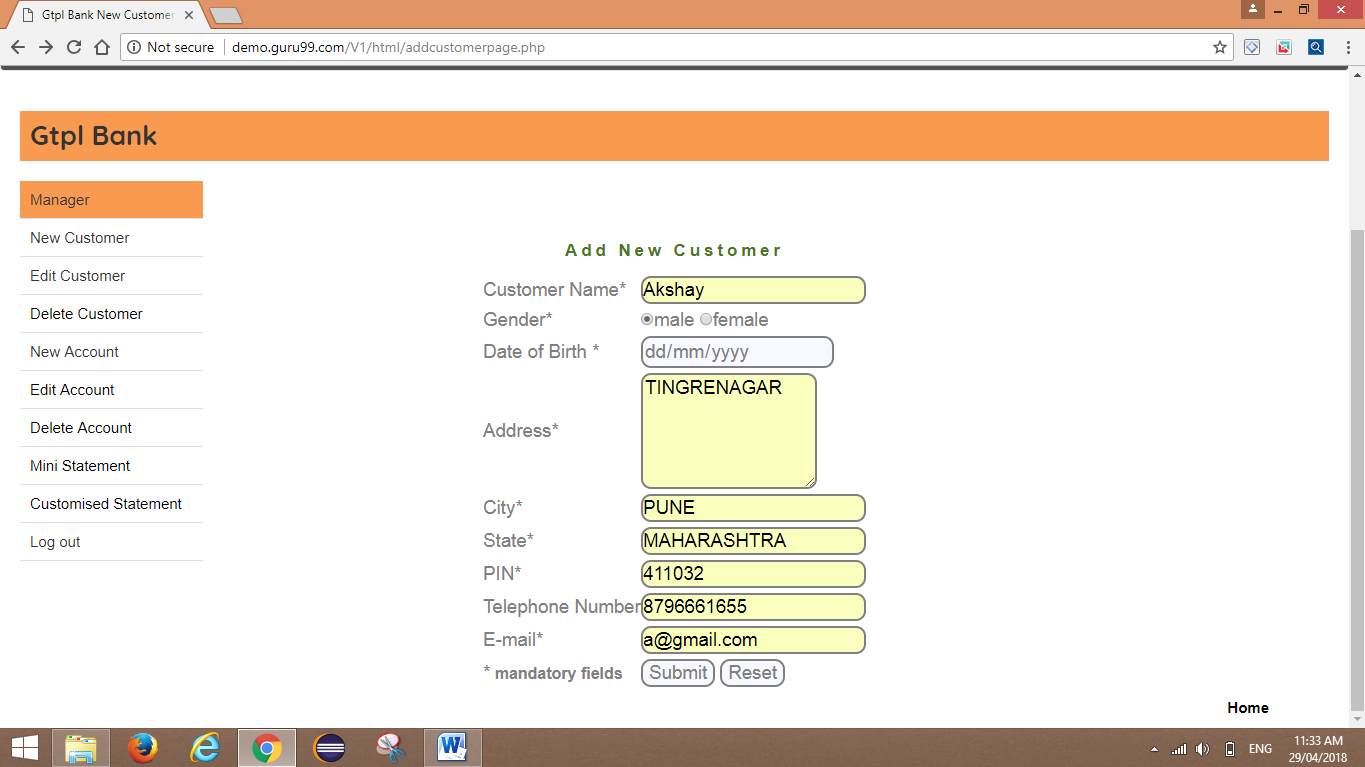
**Expected Result :-**

Customer ID should be automatically populated for a new customer.

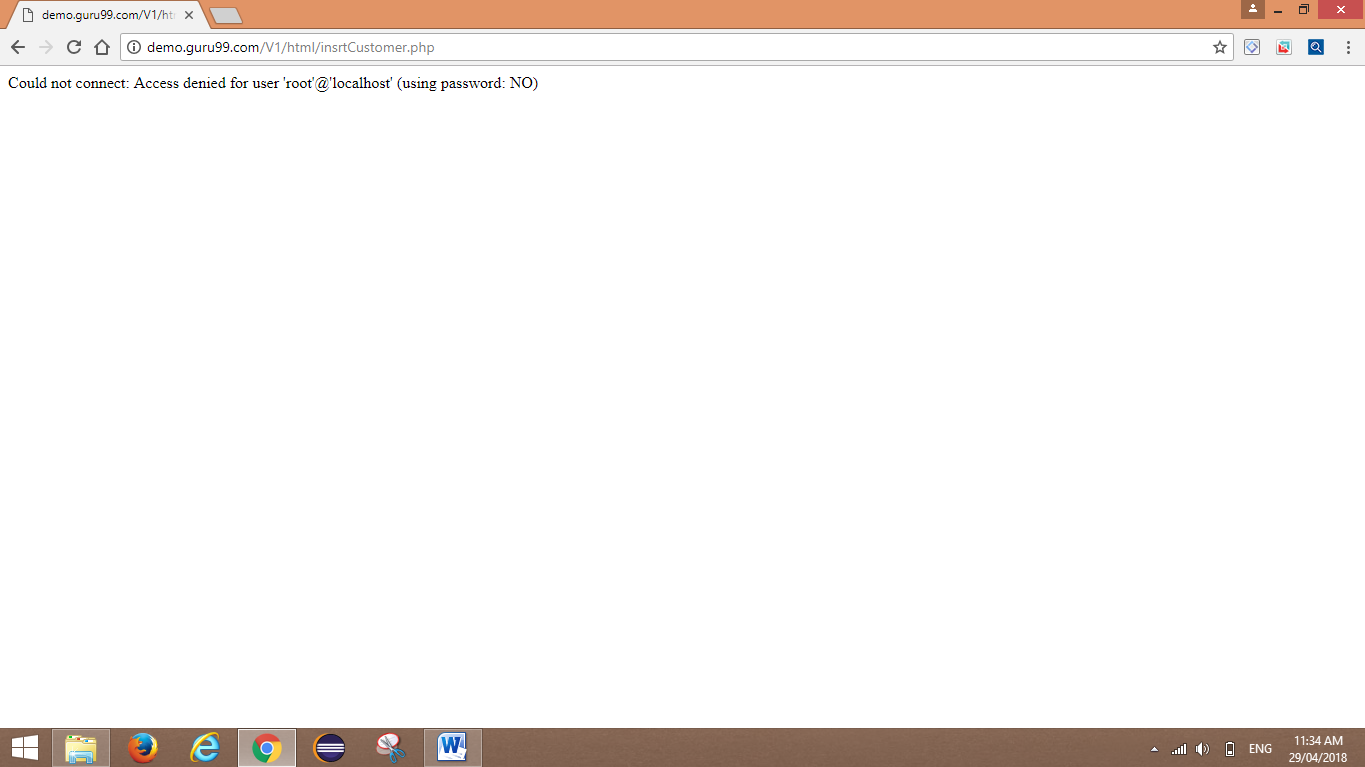
**Actual Result :-**

Customer ID is not populating automatically for a new customer.

**Defect Evidences :-**

****

After clicking on submit button Customer ID is not populated automatically.



Defect ID should be automatically created after raising a defect. For Ex:- Defect 1234.

**Defect Cycle :-**

**ASSIGNED  
(**Assign to the development team.**)**

**OPEN**(Defect is open and assigned for further investigation.)

**NEW**Defect 1234

**REOPEN  
(**If retest failed Defect 1234 is reopen.)

**CLOSED  
(**If retest passed Defect 1234 is closed.)

**RETEST**(Defect retest need to be done.)

**FIXED**

**(**Code fixed received.**)**

**Advantages Of Manual Testing :-**

* Manual testing is eye ball testing.
* Applications with short life cycles.
* Applications that have constantly changes.
* It requires less time and expense to begin productive manual testing.
* Automation cannot replace human intuition, inference, and inductive reasoning.
* Automation testing cannot change course in the middle of a test run to examine something that had not been previously considered.
* Manual QA testing can be used in both small and big projects.
* Easily we can update our test case according to project movement.
* It is covered in limited cost.
* Easy to learn for new people who are entered into testing.

**Disadvantages Of Manual Testing :-**

* GUI objects size difference and colour combination etc. is not easy to find out in manual testing.
* Load testing and performance testing is not possible in manual testing.
* Running test manually is very time consuming job.
* Regression Test cases are time consuming if it is manual testing.

**Automation Testing :-**

Automation testing is a technique uses an application to implement entire life cycle of the software in less time and provides efficiency and effectiveness to the testing software.

Automation testing is an Automatic technique where the tester writes scripts by own and uses suitable software to test the software. It is basically an automation process of a manual process. Like regression testing, Automation testing also used to test the application from load, performance and stress point of view.

In other word, Automation testing uses automation tools to write and execute test cases, no manual involvement is required while executing an automated test suite. Usually, testers write test scripts and test cases using the automation tool and then group into test suites.

The main goal of Automation testing is to increase the test efficiency and develop software value.

**Automation Testing Beats Manual Testing :-**

There are various points where automation testing beats manual testing. These points are very important and strong where automation is feasible than manual.

* Faster Execution
* More Accurate
* Lesser Investment In Human Resource
* Supports Regression Testing
* Frequent Executions
* Supports Light Out Execution

**Selenium As An Automation Testing Tool :-**

Selenium is a suite of software tools to automate web browsers.

It is open source and mainly used for functional testing and regression testing.



* Supports different PL -> Java, Python, C#, PHP, Ruby, Pearl, JavaScript.
* Supports different OS -> Windows, Mac, Linux, iOs, Android.
* Supports different Browsers -> IE, Firefox, Chrome, Safari, Opera.

**Selenium Vs Other Tools :-**

|  |  |  |  |
| --- | --- | --- | --- |
| Features | UFT | IBM RIT | Selenium |
| Licence | Required | Required | Open Source |
| Cost | High | High | Less |
| Customer Support | Dedicated HP Support | Dedicated IBM Support | Open Source Community |
| Hardware Consumption during script execution | High | High | Low |
| Coding Experience | Not much | Required | Should be very good along with technical capabilities of integrating framework. |
| Environment Support | Only for windows | Only for windows | Any OS |
| Language Support | VB Script | Java and C# | Any PL |

**Selenium Suite Of Tools:-**

IDE

RC

Selenium

WEBDRIVER

GRID

**What Is Selenium WebDriver:-**

* Selenium WebDriver is a programming interface to create and execute test cases.
* Test cases are created and executed using Element Locators/ Object Locators/ WebDriver methods.
* Selenium WebDriver has only a programming interface; not IDE
* Fast as it interact with directly browser.
* Each browser has its own driver on which the application runs. Selenium webdriver makes direct calls to the browser.





Automation Script Browser Web Server

**WebDriver Features and Drawbacks :-**

**Selenium WebDriver Features :-**

* Supports Most Programing languages, Browsers & Operating System.
* Overcome limitations of Selenium 1 like file upload, download, pop-ups & dialog barrier.
* WebDriver API is simpler than RC’s API. It does not contain redundant & confusing commands.
* Supports Batch Testing, Cross Browser Testing etc.

**Selenium WebDriver Drawbacks :-**

* Detailed test report cannot be generated.
* It dosen’t have IDE so it is difficult to create test cases.











Automates

Invokes





**Cucumber Framework :-**

Cucumber is a testing tool that supports “**Behavior Driven Development (BDD)**” framework. It defines application behavior using simple English text, defined by a language called Gherkin.

Cucumber allows automation functional validation that is easily read and understood. Cucumber was initially implemented in Ruby and then extended to Java framework. Both the tools support native JUnit.

This tutorial is fairly comprehensive and covers all the necessary aspects on Cucumber using examples for easy understanding.

**Advantages Of Cucumber Framework :-**

* Cucumber supports different languages like Java.net and Ruby.
* It acts as a bridge between the business and technical language. We can accomplish this by creating a test case in plain English text.
* It allows the test script to be written without knowledge of any code, it allows the involvement of non-programmers as well.
* It serves the purpose of end-to-end test framework unlike other tools.
* Due to simple test script architecture, Cucumber provides code reusability.

**Feature:** Bank Retail/Corporate Customer Records

As A System Administrator.

I want to handle the customer details properly.

So That the customer details modified or retrieved anytime as per requirement.

*@CR745\_TC01*

**Scenario:** Enter new Retail/Corporate customer details.

*Given* Customer details are given in CDI file as follows.

| Customer Name | Gender | DOB | ADDRESS | CITY | PINCODE | STATE | CONTACT |

| AKSHAY SHETE | M | 17/07/1996 | TINGRENAGAR | PUNE | 411032 | MAHARASHTRA | 8796661555 |

| PRIYANKA APSHINGE | F | 09/10/1995 | TINGRENAGAR | PUNE | 411032 | MAHARASHTRA | 9545722245 |

| SANDIP GHOLAP | M | 05/10/1995 | VIMANAGAR | PUNE | 411032 | MAHARASHTRA | 70301355696 |

*When* System Administrator enter all the details properly.

*Then* Details must be saved successfully.

*@CR745\_TC02*

**Scenario:** Create New Bank Account for a customer

*Given* For new customer create a new bank account as follows.

| Customer ID | Initial Amount | Type |

| CQO008696 | 1000 | SA |

| CQO008692 | 10000 | CU |

*When* Create new account with above details.

*And* Checked whether new account contains all the required information of the customer.

*Then* Account should be created successfully.

*And* Required Information must be present.

**JUnit Framework (Automation Tool) :-**

JUnit is a unit testing framework for Java programming language. It plays a crucial role test-driven development, and is a family of unit testing frameworks collectively known as JUnit.

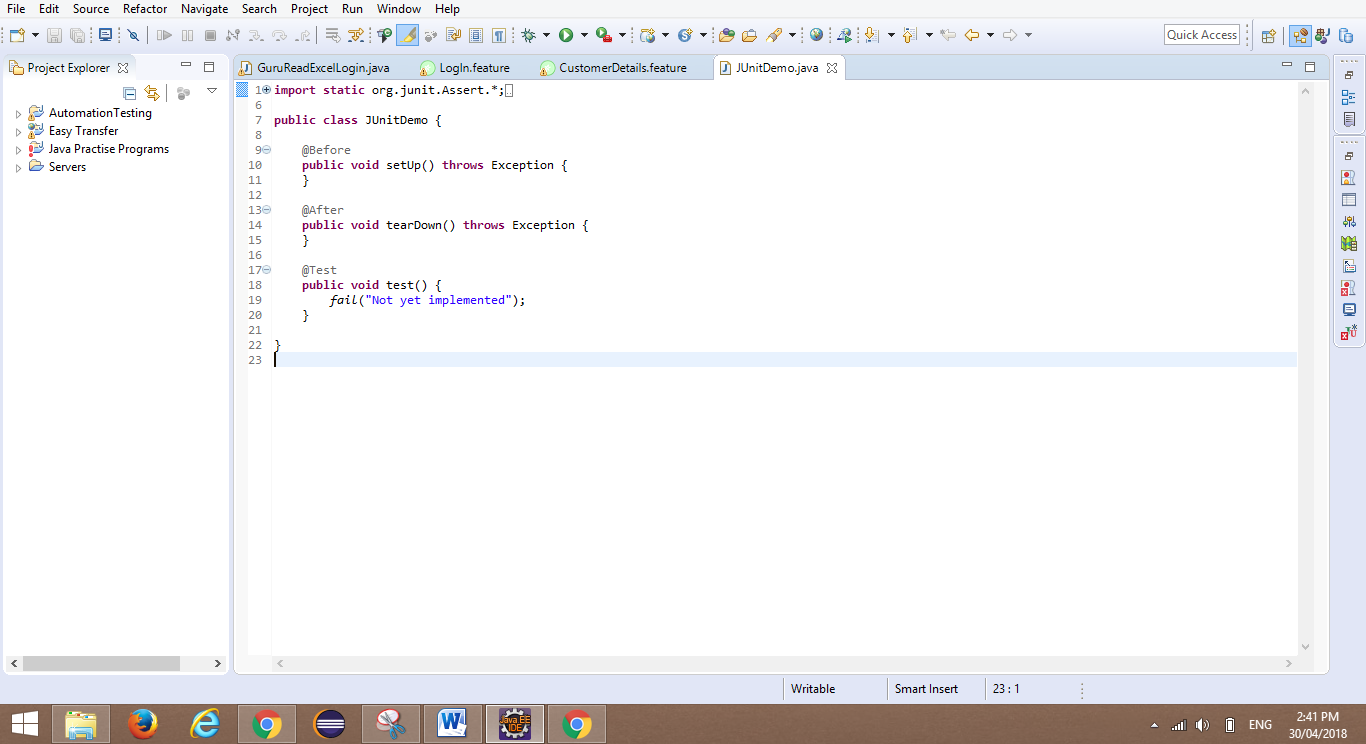
JUnit promotes the idea of "first testing then coding", which emphasizes on setting up the test data for a piece of code that can be tested first and then implemented. This approach is like "test a little, code a little, test a little, code a little." It increases the productivity of the programmer and the stability of program code, which in turn reduces the stress on the programmer and the time spent on debugging.

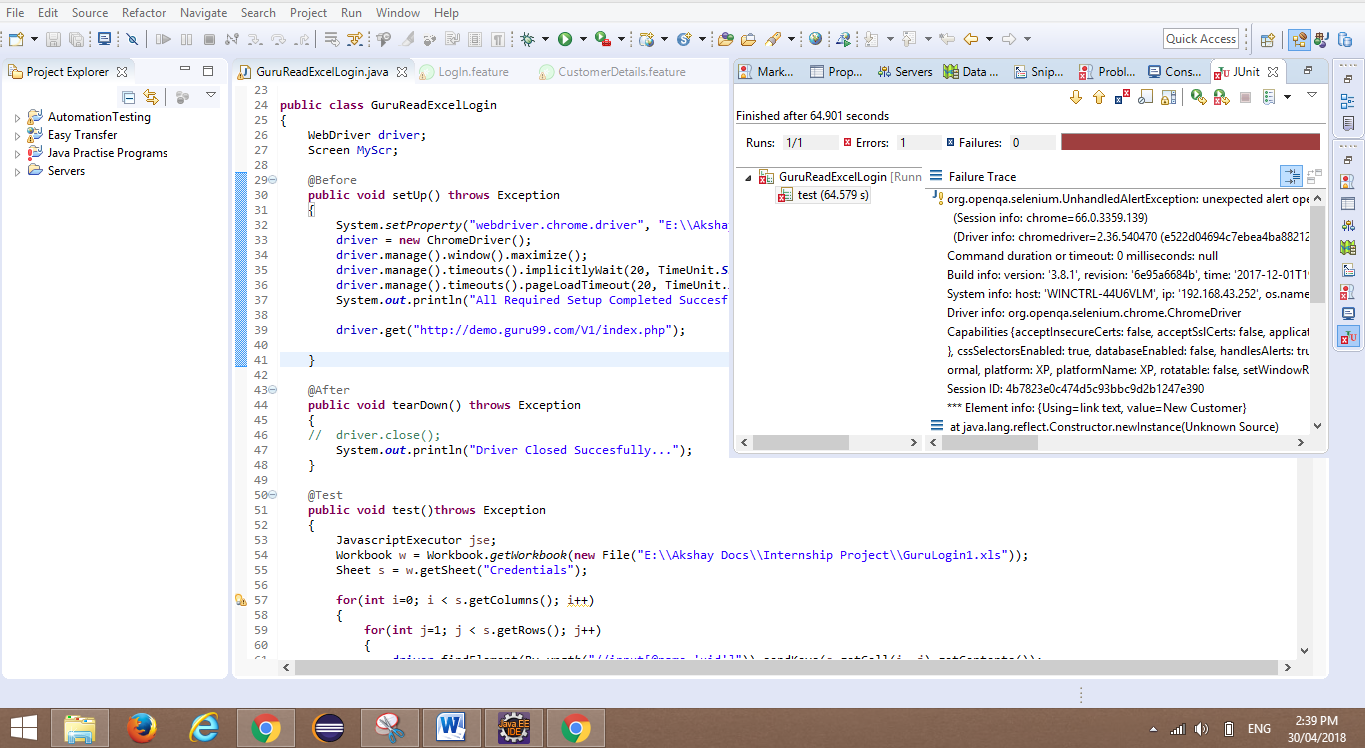
**Features of JUnit :-**

* JUnit is an open source framework, which is used for writing and running tests.
* Provides annotations to identify test methods.
* Provides assertions for testing expected results.
* Provides test runners for running tests.
* JUnit tests allow you to write codes faster, which increases quality.
* JUnit is elegantly simple. It is less complex and takes less time.
* JUnit tests can be run automatically and they check their own results and provide immediate feedback. There's no need to manually comb through a report of test results.
* JUnit tests can be organized into test suites containing test cases and even other test suites.
* JUnit shows test progress in a bar that is green if the test is running smoothly, and it turns red when a test fails.

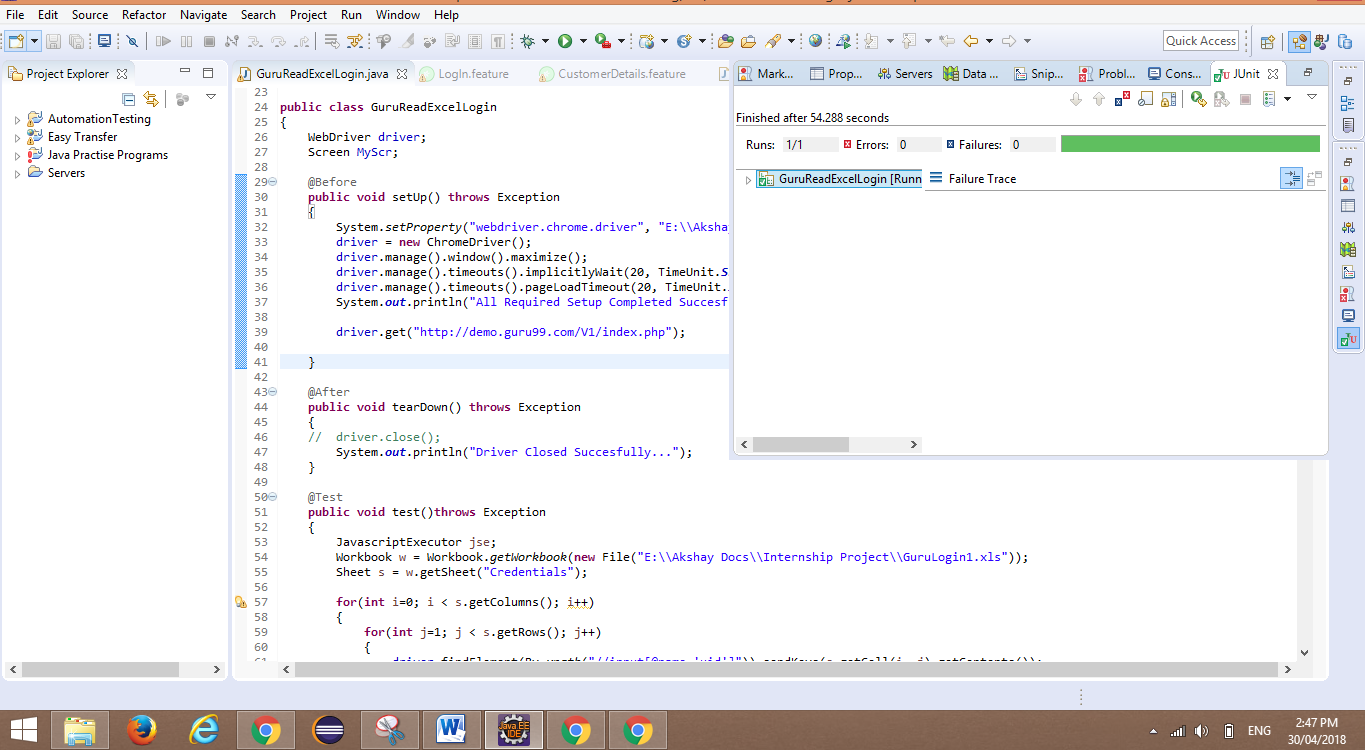
**JUnit Framework Screenshots :-**

Basic JUnit Framework is looked like below :-

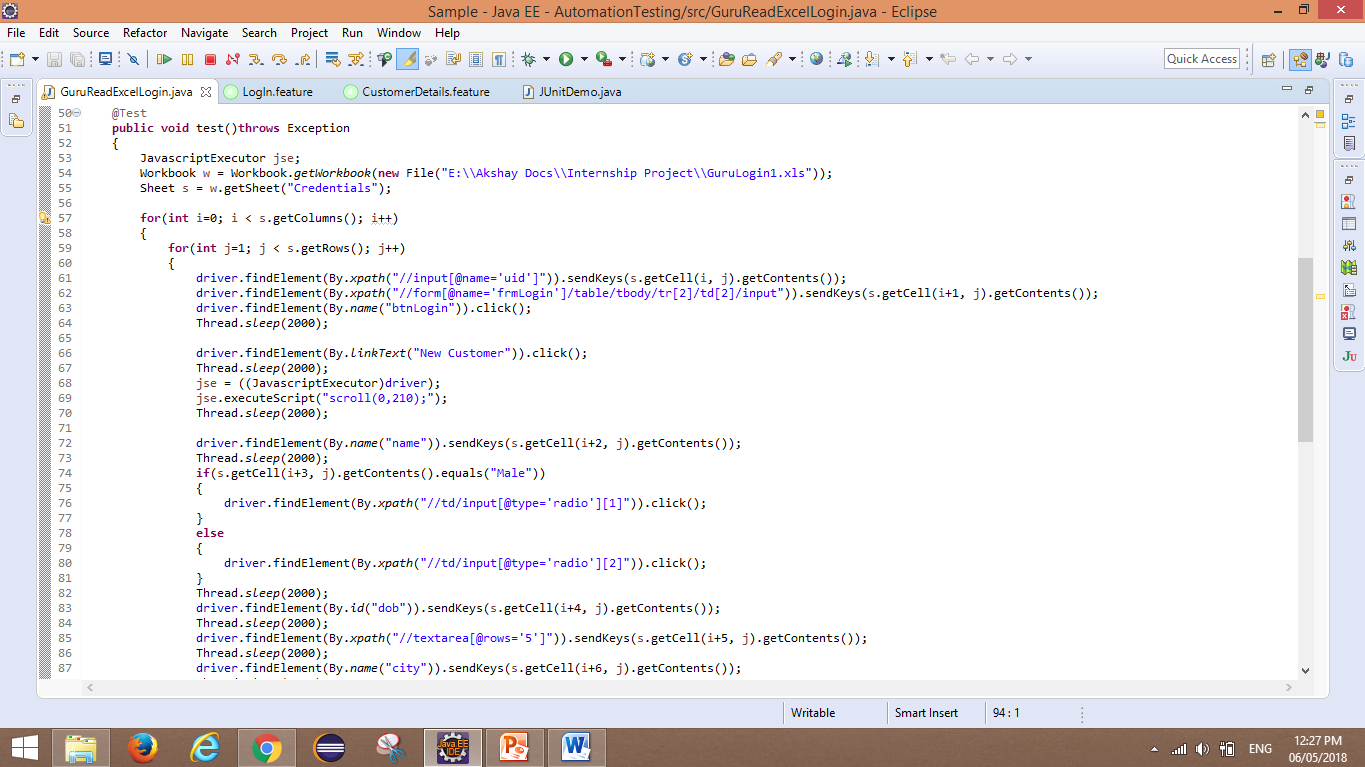


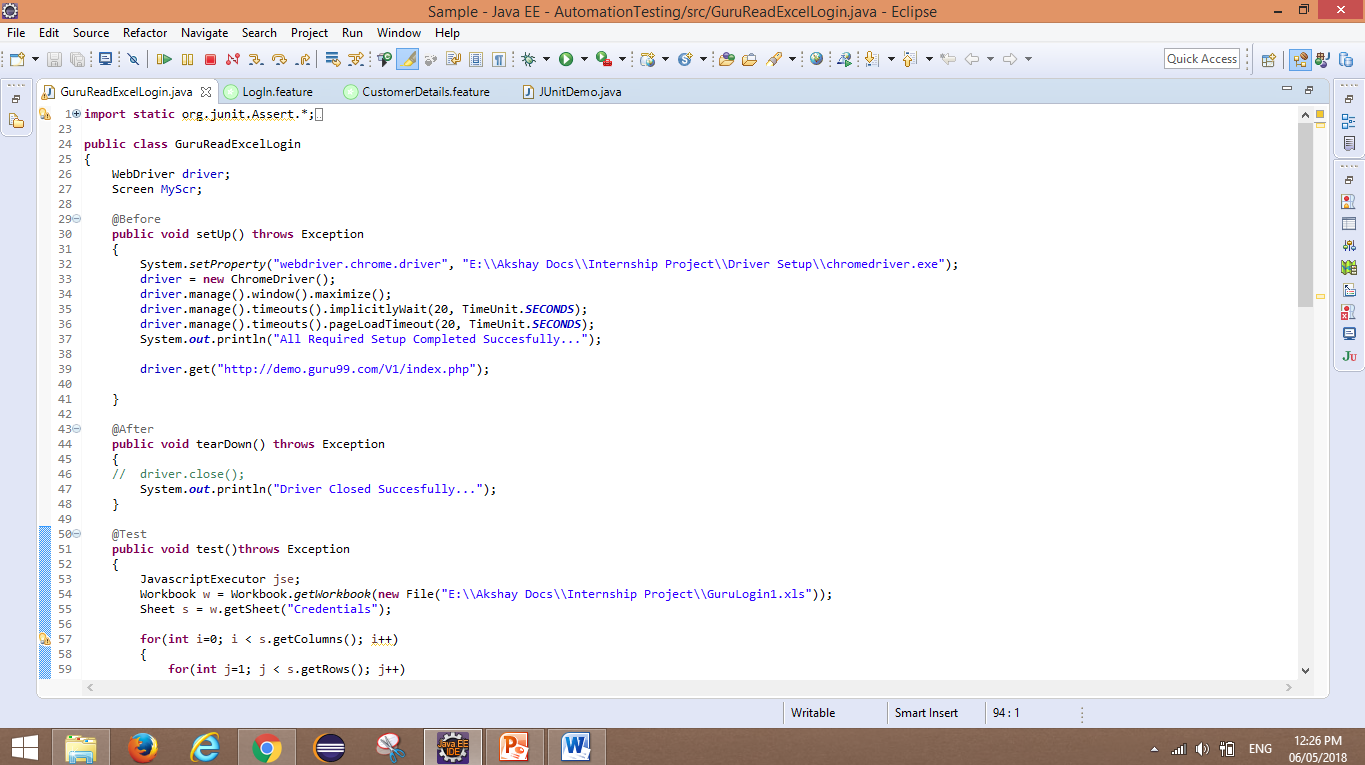
If our test script failed then JUnit Framework shows us respected error in detail with proper time script took for execution.

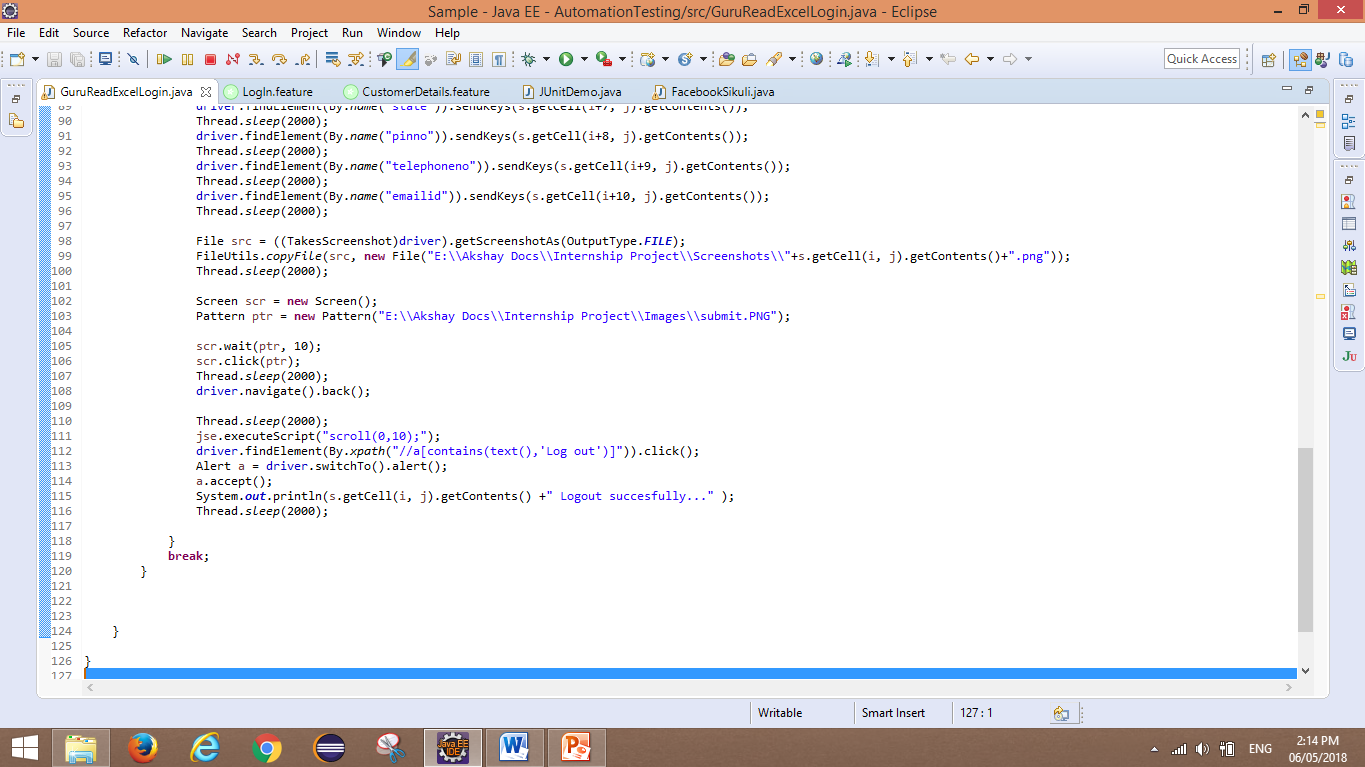
If our test script passed then JUnit Framework shows us only time taken by script for execution.

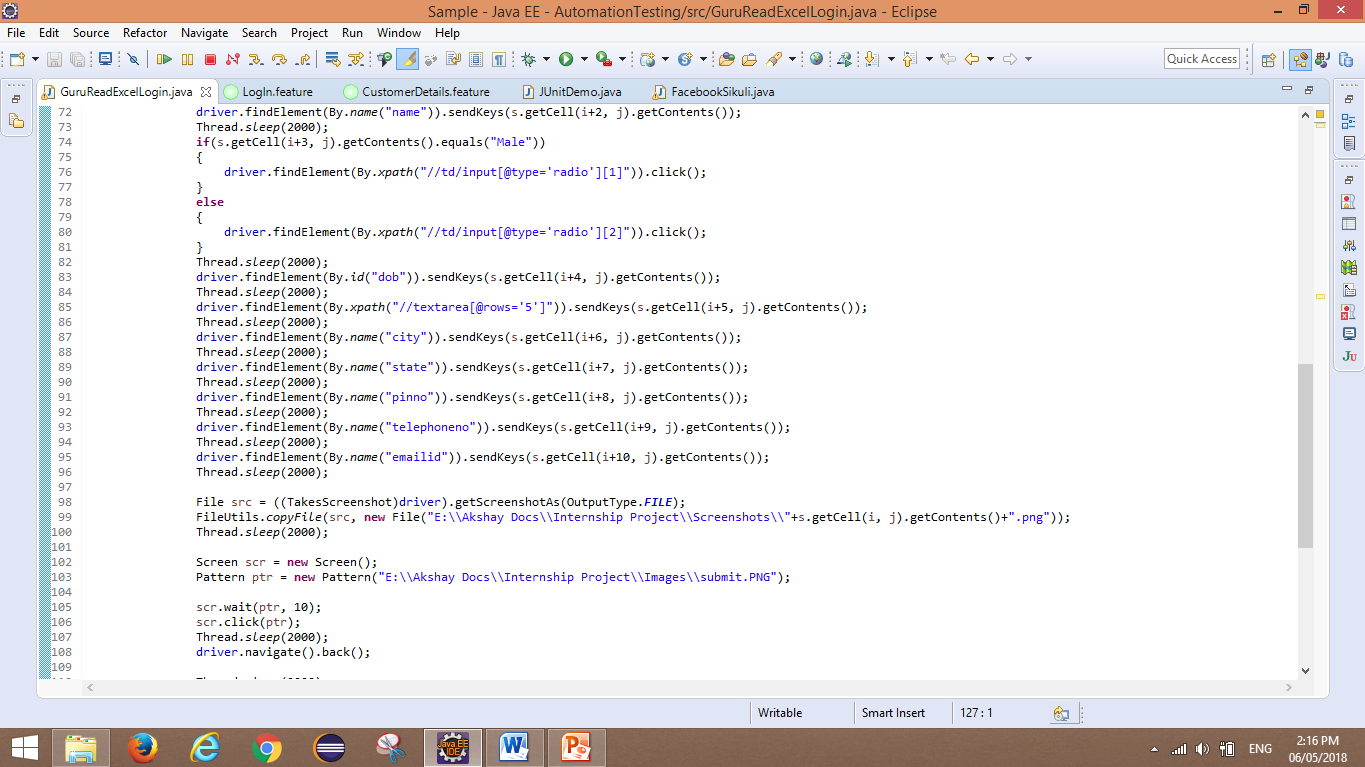


**Code Snippet Of Bank Product Testing :-**

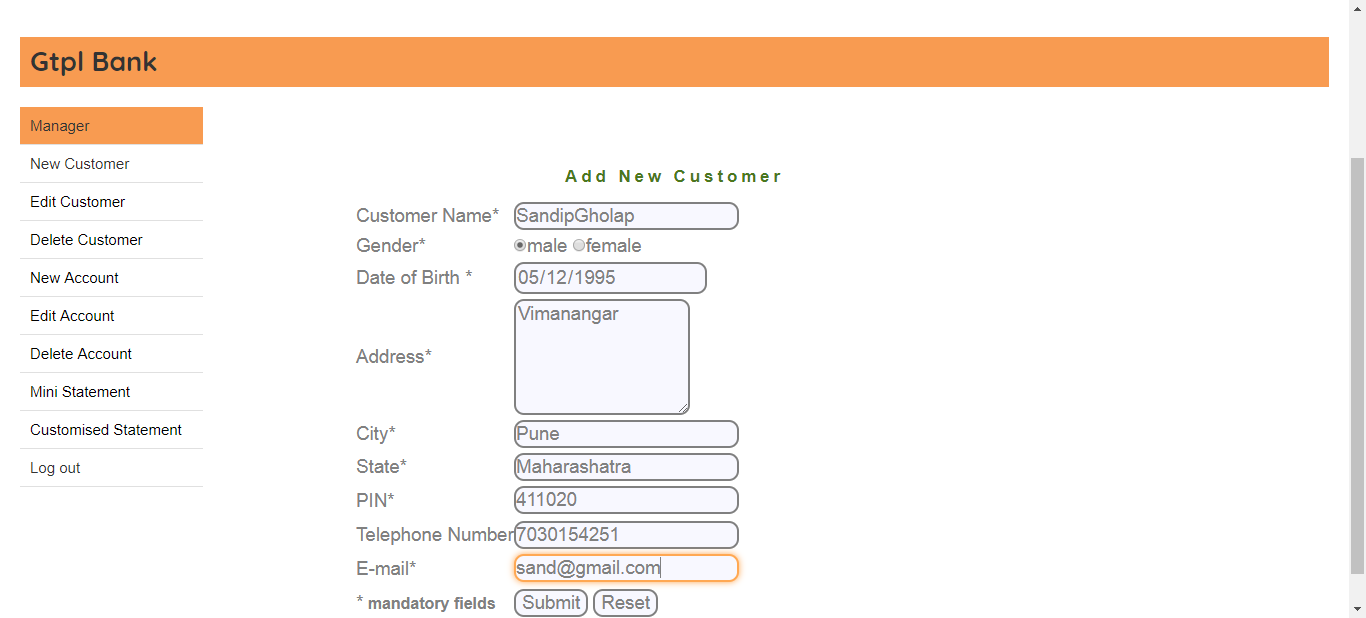
****

****



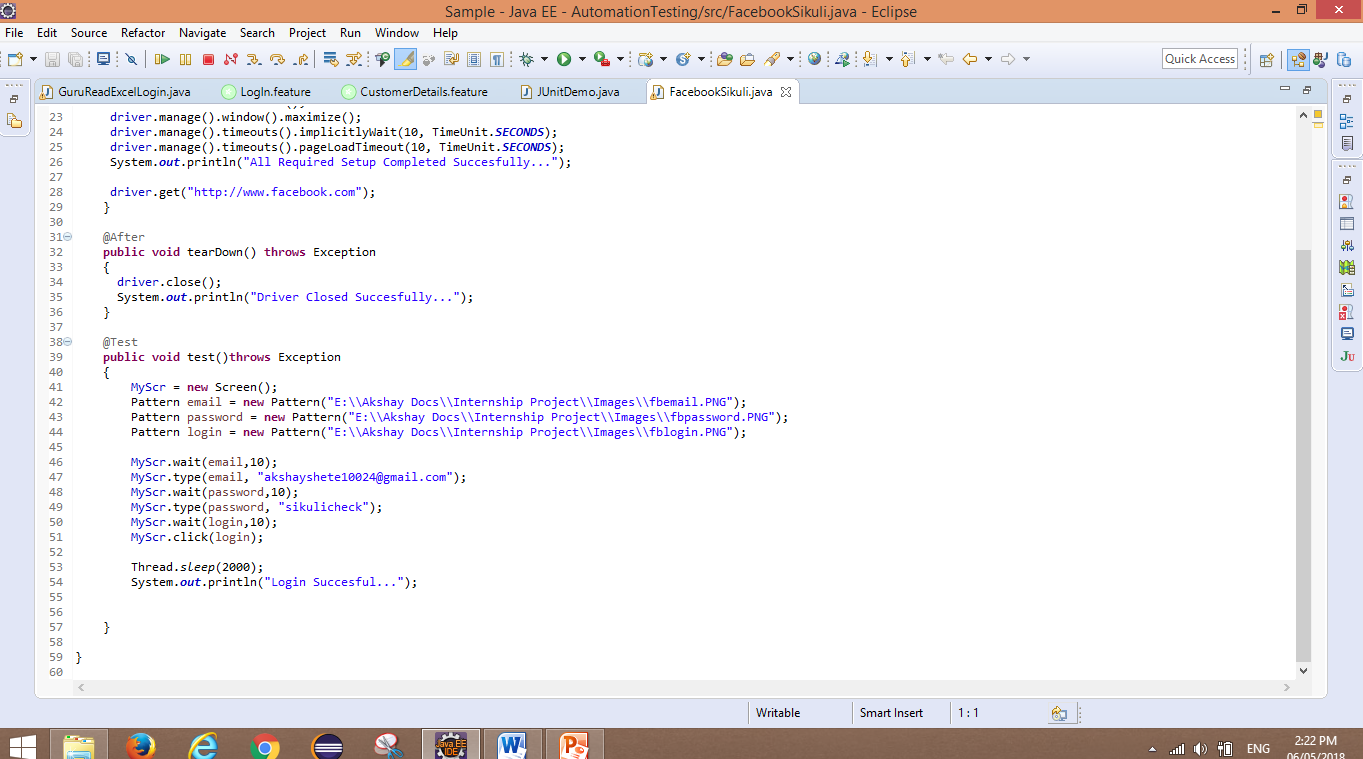








**Sikuli Images**

****

**Bibliography :-**

I found various information from popular web search Engine Google as well as my project mentor also guided me very well and they taught me each and every concept of testing clearly and effectively.

Google and Testing Fundamentals book helped me a lot throughout this project.

**Websites:-**

<http://www.apache.org>

<http://www.eclipse.org>

<http://en.wikipedia.org>

<http://www.techopedia.com>