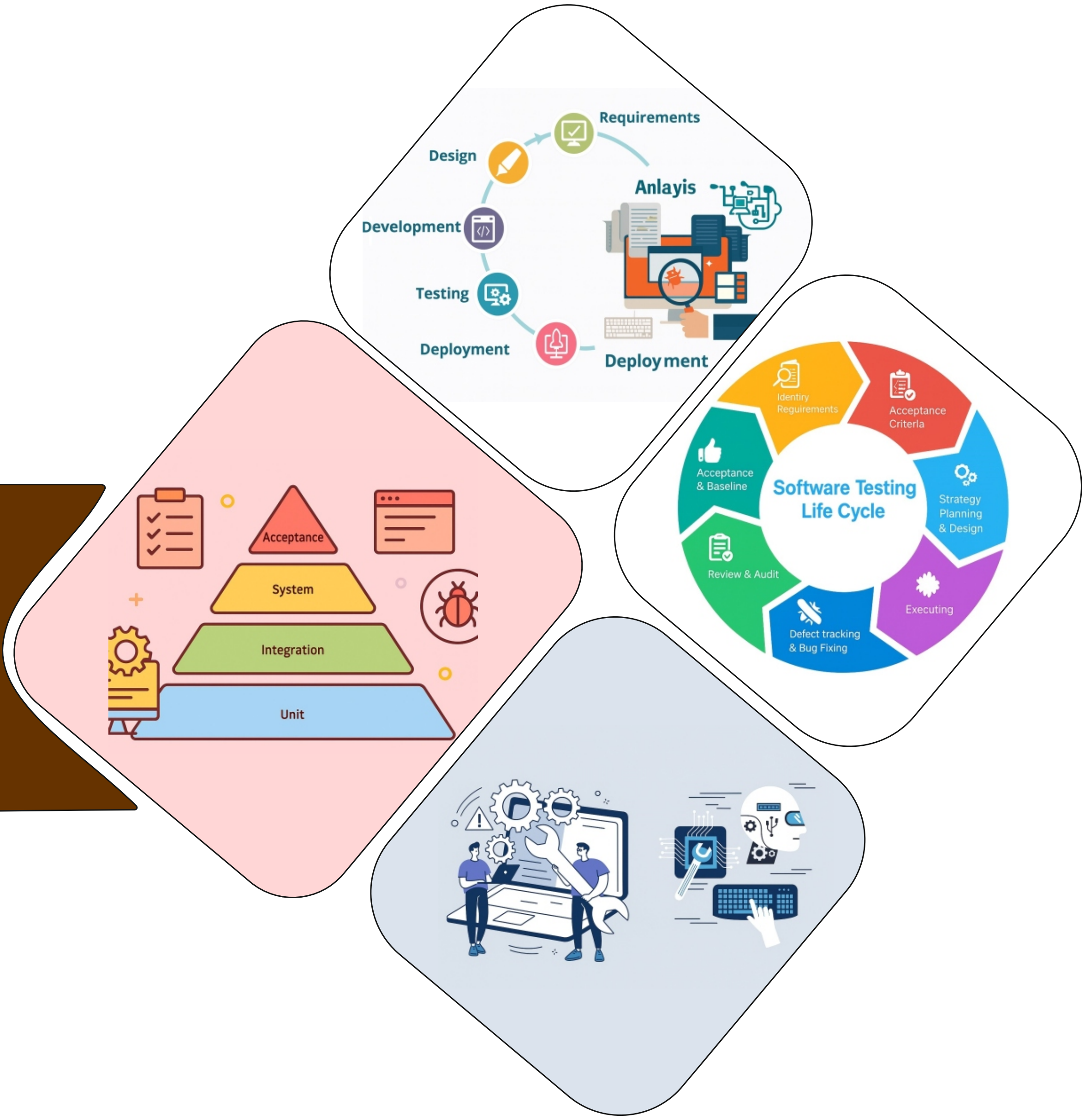
SCHEME :K

Name :___________________________
Roll No.:_________ Year : 20 ___ 20 __
Exam Seat No. :____________________

LABORATORY MANUAL FOR

SOFTWARE TESTING (316314)



COMPUTER ENGINEERING GROUP

MAHARASHTRA STATE BOARD OF
TECHNICAL EDUCATION, MUMBAI
(Autonomous)(ISO21001:2018)(ISO/IEC27001:2013)

Vision

To ensure that the Diploma level Technical Education constantly matches the latest requirements of Technology and industry and includes the all-round personal development of students including social concerns and to become globally competitive, technology led organization.

Mission

We, at MSBTE are committed to offer the best in class academic services to the students and institutes to enhance the delight of industry and society. This will be achieved through continual improvement in management practices adopted in the process of curriculum design, development, implementation, evaluation and monitoring system along with adequate faculty development programmes

Core Values

MSBTE believes in the following:

- Skill development in line with industry requirements
- Industry readiness and improved employability of Diploma holders
- Synergistic relationship with industry
- Collective and Cooperative development of all stake holders
- Technological interventions in societal development
- Access to uniform quality technical education

A Practical Manual

for

Software Testing

(316314)

Semester-VI

CM/ CO/ CW / IF/ IH/ SE

Maharashtra State Board of Technical Education, Mumbai

(Autonomous) (ISO 21001:2018)(ISO/IEC 27001:2013)

‘K’ Scheme Curriculum

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

(Autonomous) (ISO 21001:2018)(ISO/IEC 27001:2013)

Address: 4th floor, Govt. Polytechnic Building, 49,
Kherwadi, Bandra (E), Mumbai- 400 051

Tel: 022 62542100

Email: secretary@msbte.com

Maharashtra State Board of Technical Education

Certificate

This is to certify that Mr./ Ms ……………………………………………………… Roll No……….…... of…………………………Semester of Diploma in………...…………………………………………...........of **Institute Name** …………..……....……………………………….(**Inst.Code:**……..……...) has completed the term work satisfactorily in Course **Software Testing (Course Code:316314)** for the academic year 20…….. to 20…….. as prescribed in the curriculum.

Place ……………….   Enrollment No……………………

Date: …....................   Exam Seat No. …………………......

Course Teacher   **Head of the Department**   **Principal**

Seal of
the
Institute

Preface

The primary focus of any engineering laboratory/field work in the technical education system is to develop the much needed industry relevant competencies and skills. With this in view, MSBTE embarked on this innovative 'K' Scheme curricula for engineering Diploma programmes with outcome-based education as the focus and accordingly, relatively large amount of time is allotted for the practical work. This displays the great importance of laboratory work making each teacher, instructor and student to realize that every minute of the laboratory time need to be effectively utilized to develop these outcomes, rather than doing other mundane activities. Therefore, for the successful implementation of this outcome-based curriculum, every practical has been designed to serve as a 'vehicle' to develop this industry identified competency in every student. The practical skills are difficult to develop through 'chalk and duster' activity in the classroom situation. Accordingly, the “K” scheme laboratory manual development team designed the practical's to focus on outcomes, rather than the traditional age old practice of conducting practical's to verify the theory (which may become a byproduct along the way).

This laboratory manual is designed to help all stakeholders, especially the students, teachers and instructors to develop in the student the pre-determined outcomes. It is expected from each student that at least a day in advance, they have to thoroughly read the concerned practical procedure that they will do the next day and understand minimum theoretical background associated with the practical. Every practical in this manual begins by identifying the competency, industry relevant skills, course outcomes and practical outcomes which serve as a key focal point for doing the practical. Students will then become aware about the skills they will achieve through procedure shown there and necessary precautions to be taken, which will help them to apply in solving real-world problems in their professional life.

This manual also provides guidelines to teachers and instructors to effectively facilitate student-centered lab activities through each practical exercise by arranging and managing necessary resources in order that the students follow the procedures and precautions systematically ensuring the achievement of outcomes in the students.

Operating systems are an essential part of any computer system. Similarly, a course on operating systems is an essential part of any computer group. We hope that students will also find it useful. It provides a clear description of practical performance, execution and working of Operating System.

Although all care has been taken to check for mistakes in this laboratory manual, yet it is impossible to claim perfection especially as this is the first edition. Any such errors and suggestions for improvement can be brought to our notice and are highly welcome.

Program Outcomes (POs) to be achieved through Course:

PO1	Basic and Discipline specific knowledge: Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
PO2	Problem analysis: Identify and analyses well-defined engineering problems using codified standard methods.
PO3	Design/ development of solutions: Design solutions for well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
PO4	Engineering Tools, Experimentation and Testing: Apply modern engineering tools and appropriate technique to conduct standard tests and measurements.
PO5	Engineering practices for society, sustainability and environment: Apply appropriate technology in context of society, sustainability, environment and ethical practices.
PO6	Project Management: Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
PO7	Life-long learning: Ability to analyses individual needs and engage in updating in the context of technological changes.

List of Relevant Skills

The following industry-relevant skills of the competency “Perform Software Testing and Quality Assurance Activities” are expected to be developed in you by performing the practicals of this laboratory manual:

1. Design and execute various types of test cases based on software requirements and specifications.
2. Apply black-box testing techniques such as equivalence partitioning, boundary value analysis, decision tables, and state transition testing.
3. Perform white-box testing techniques including statement coverage, branch coverage, and condition coverage.
4. Use automated testing tools (e.g., Selenium, JUnit) to execute and validate test scripts.
5. Prepare and maintain test documentation such as test plans, test scenarios, test case sheets, and defect reports.
6. Identify, log, track, and analyze software defects using bug tracking tools (e.g., Bugzilla, Mantis, JIRA).
7. Conduct performance, load, and stress testing using relevant tools to evaluate system robustness.
8. Apply regression testing and retesting methods to ensure software stability after modifications.
9. Understand and use software quality assurance (SQA) processes, standards, and metrics for evaluating software quality.
10. Collaborate effectively in testing teams and follow testing workflows within the Software Development Life Cycle (SDLC) and STLC.

Practical Course Outcome Matrix

Course Outcomes (COs)

CO1	Explain various software testing methods.
CO2	Prepare test cases for different levels of testing.
CO3	Prepare test plan for a given application.
CO4	Create defect report for a given application.
CO5	Apply automation testing tools to test software.

Sr. No.	Title of the Experiment	CO1	CO2	CO3	CO4	CO5
1	*Design Test cases for purchase order management system specification	✓				
2	*Design test cases for Inventory management System based on System Specification	✓				
3	Design test cases for calculator to verify its functionality (Black Box Testing)	✓				
4	Design test cases for hostel admission form	✓				
5	*Design test cases for different tasks (OTP Verification, Image Upload) in any software module using black box testing	✓				
6	Test various functionality of railway reservation system		✓			
7	*Validate login procedure for E-Commerce Application (Flipkart or Amazon)		✓			
8	Test functionality of Web Pages of any website		✓			
9	*Design Test Cases for Control and Decision Making Statements (Use C Language)		✓			
10	*Design test cases for online mobile recharge		✓			
11	*Design test cases for flight ticket booking system		✓			
12	Design test plan and cases for elevator			✓		
13	*Design test plan and test cases for Notepad (MS Window based) Application			✓		

14	Prepare test report for any website			✓		
15	*Design test cases and test summary report for a travel booking application			✓		
16	*Design test plan and test cases for the login functionality of a social media application			✓		
17	*Generate Defect Report for Library Management System				✓	
18	*Validate Defect Report for ATM Machine				✓	
19	Execute Test Cases to Generate Defect Report for any login form				✓	
20	Defect Report for Hostel Admission Form				✓	
21	*Installation and Configuration of Selenium IDE					✓
22	*Test Case Design and Execution for Notepad (Windows-Based) Using Selenium IDE					✓
23	Test Case Design and Execution for MS Word application using Selenium IDE					✓
24	*Installation and Configuration of Selenium WebDriver					✓
25	*Browser Automation with WebDriver					✓
26	Handling Multiple Windows and Tabs in WebDriver					✓

Guidelines to Teachers

1. Teachers should align the explanation of the topic to teaching learning outcome (TLOs).
2. Refer to laboratory learning outcome (LLOs) for the execution of the practical to focus on the defined objectives.
3. Promote life-long learning by training the students to equip themselves with essential knowledge, skills and attitudes.
4. If required, provide demonstration for the practical emphasizing on the skills that the student should achieve.
5. Teachers should give opportunity to the students for exhibiting their skills after the demonstration.
6. Provide feedback and/or suggestions and share insights to improve effectiveness.
7. Assess students' skill achievement related to COs of each unit.

Instructions for Students

1. 100% attendance is compulsory for all practical sessions.
2. Students must adhere to ethical practices.
3. All the students must follow the schedule of practical sessions, complete the assigned work/activity and submit the assignment in stipulated time as instructed by the course teacher.
4. Students shall listen carefully the lecture given by teacher about importance of subject, learning structure, course outcomes.
5. Students shall understand the purpose of experiment and its practical implementation.
6. Students shall write the answers of the questions during practical.
7. Student should feel free to discuss any difficulty faced during the conduct of practical.
8. Student shall attempt to develop related hands-on skills and gain confidence.
9. Students should develop habit to submit the write-ups on the scheduled dates and time.

Content Page

List of Practical and Formative Assessment Sheet

Sr. No	Practical Title	Date of Performance	Date of Submission	Assessment Marks (25)	Teacher's Sign	Remark
1	*Design Test cases for purchase order management system specification					
2	*Design test cases for Inventory management System based on System Specification					
3	Design test cases for calculator to verify its functionality (Black Box Testing)					
4	Design test cases for hostel admission form					
5	*Design test cases for different tasks (OTP Verification, Image Upload) in any software module using black box testing					
6	Test various functionality of railway reservation system					
7	*Validate login procedure for E-Commerce Application (Flipkart or Amazon)					
8	Test functionality of Web Pages of any website					
9	*Design Test Cases for Control and Decision Making Statements (Use C Language)					
10	*Design test cases for online mobile recharge					
11	*Design test cases for flight ticket booking system					
12	Design test plan and cases for elevator					
13	*Design test plan and test cases for Notepad (MS Window based) Application					

14	Prepare test report for any website					
15	*Design test cases and test summary report for a travel booking application					
16	*Design test plan and test cases for the login functionality of a social media application					
17	*Generate Defect Report for Library Management System					
18	*Validate Defect Report for ATM Machine					
19	Execute Test Cases to Generate Defect Report for any login form					
20	Defect Report for Hostel Admission Form					
21	*Installation and Configuration of Selenium IDE					
22	*Test Case Design and Execution for Notepad (Windows-Based) Using Selenium IDE					
23	Test Case Design and Execution for MS Word application using Selenium IDE					
24	*Installation and Configuration of Selenium WebDriver					
25	*Browser Automation with WebDriver					
26	Handling Multiple Windows and Tabs in WebDriver					
				Total		

***Total marks to be transferred to proforma published by MSBTE**

Note:

- '*' Marked Practical’s (LLOs) Are mandatory.
- Minimum 80% of above list of lab experiment are to be performed.
- Judicial mix of LLOs are to be performed to achieve desired outcomes.

Practical No. 1:*Design test cases for purchase order management based on system specification

I Practical Significance

This practical emphasizes the structured design and documentation of test cases for a Purchase Management System (PMS). It enables learners to understand functional testing, apply equivalence partitioning, and utilize drivers and stubs during early testing stages. The experiment builds foundational skills for requirement-based testing and defect identification in enterprise applications.

II Industry / Employer Expected Outcome

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO1 - Explain various software testing methods.

IV Laboratory Learning Outcome(s)

LLO 1.1: Write test cases for Purchase Management System.

V Relevant Affective Domain related Outcomes

1. Responsibility: Following proper testing process and documentation standards.
2. Attention to Detail: Ensuring each test case covers all functional and edge scenarios.
3. Professional Ethics: Accurately reporting actual results and failures without bias.

VI Relevant Theoretical Background

A Purchase Management System (PMS) manages supplier records, purchase orders, and approval workflows. Testing ensures that purchase requests, supplier lookups, and calculations work as per specifications.

- When parts of the system are under development:
- Drivers simulate upper modules that invoke lower-level functions.
- Stubs simulate lower modules that return dummy responses.
- Testing Techniques Used:
- Equivalence Class Partitioning (ECP)
- Boundary Value Analysis (BVA)
- Requirement-Based Testing.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Analyze the Purchase Order Management module specification.
2. Identify key functionalities (Create Order, Get Supplier Details, Validate Inputs).
3. Determine where to apply drivers and stubs for simulation.
4. Derive test scenarios and test cases based on specifications.
5. Use the given table format to record test cases, expected results, and actual results.
6. Execute the test cases manually or through automation tools.
7. Document Pass/Fail status and summarize findings.

IX Example

In the Purchase Order Management system:

The function createPurchaseOrder() is implemented, but the main UI module is not ready → a Driver is used to test it.

The function getSupplierDetails() is not yet implemented → a Stub is used to simulate supplier data.

Test Case ID	Objective	Steps (Input)	Expected Output	Actual Output	Status
TC_PO_001	Verify order creation with valid inputs	createPurchaseOrder(SupplierID=S001, ItemID=I102, Qty=10)	Purchase Order created successfully	Purchase Order created successfully	**Pass**
TC_PO_002	Check boundary value for quantity	createPurchaseOrder(SupplierID=S001, ItemID=I102, Qty=0)	Error: “Quantity must be greater than 0”	Error: “Quantity must be greater than 0”	**Pass**
TC_SUP_001	Validate response for invalid supplier ID	getSupplierDetails(SupplierID=S999)	Error: “Supplier not found”	Error: “Supplier not found”	**Pass**

TC_SUP_002	Simulate valid supplier detail retrieval	getSupplierDetails(SupplierID=S001)	(S001, ABC Traders)	(S001, ABC Traders)	**Pass**
TC_PO_003	Verify total amount calculation	createPurchaseOrder(SupplierID=S001, ItemID=I101, Qty=5, UnitPrice=100)	Total = 500	Total = 500	**Pass**
TC_PO_004	Validate error message for null supplier ID	createPurchaseOrder(SupplierID=NULL, ItemID=I101, Qty=5)	Error: “Supplier ID cannot be null”	Error: “Invalid Input” (incorrect format)	Fail

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for Purchase Order Management using equivalence partitioning.
2. Identify two integration points where stubs and drivers can be applied.
3. Execute all designed test cases and record Expected Vs Actual outputs.
4. Recommend improvements to enhance test case coverage and system reliability.

Space for Answer

………………………………………………………………………………………………………..

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

……………………………………………………………………………………………………….

…………..………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). Software Testing: Principles and Practices. Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). Software Testing. Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard – Software Testing Fundamentals. Available at: https://infyspringboard.onwingspan.com
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163

XIII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Correct design of test cases from specification	20 %
2	Appropriate use of Drivers and Stubs	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 2: *Design test cases for Inventory Management System based on system specification

I Practical Significance

This practical demonstrates how to design, document, and execute test cases for an **Inventory Management System (IMS)**. It provides hands-on experience in functional testing, applying test design techniques like equivalence partitioning and boundary value analysis, and verifying business logic related to stock updates, item tracking, and reorder levels. Students also learn how to validate system behavior through simulated drivers and stubs when modules are incomplete.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO1 - Explain various software testing methods.

IV Laboratory Learning Outcome(s)

LLO 2.1: Write test cases for Inventory Management System.

V Relevant Affective Domain related Outcomes

1. Responsibility: Following systematic test documentation and review procedures.
2. Attention to Detail: Ensuring accuracy and completeness of test case steps and data.
3. Professional Ethics: Reporting results and bugs transparently and ethically.

VI Relevant Theoretical Background

An Inventory Management System (IMS) tracks stock levels, manages reorder points, and records product movements across warehouses. Testing ensures that stock updates, item additions, deletions, and reorder alerts perform correctly.

When the complete system is unavailable:

- Drivers are used to simulate main modules that trigger inventory functions.
- Stubs simulate sub-modules like supplier data or warehouse tracking that are under development.

Testing Techniques Used:

- Equivalence Class Partitioning (ECP)
- Boundary Value Analysis (BVA)
- Requirement-Based Testing

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Analyze the Inventory Management System specification and identify core functions such as
2. Add Item, Update Stock, Check Reorder Level, and Delete Item.
3. Identify modules that are not implemented and determine where Drivers and Stubs are required.
4. Define test scenarios based on functionality and business rules.
5. Design and document test cases using the given table format.
6. Execute test cases, record actual outputs, and mark Pass/Fail status.
7. Prepare a summary report showing results and defect identification.

IX Example

In the Inventory Management System:

The updateStock() function is implemented, but the *main Inventory Dashboard* is not ready → **Driver** is used.

The getSupplierInfo() function is not yet developed → **Stub** is used to simulate supplier responses.

Test Case ID	**Objective**	**Steps (Input)**	**Expected Output**	**Actual Output**	**Status**
TC_INV_001	Verify item addition with valid details	addItem(ItemID=I1001, Name=Mouse, Qty=50, Price=250)	Item added successfully	Item added successfully	**Pass**
TC_INV_002	Validate boundary value for stock quantity	addItem(ItemID=I1002, Name=Keyboard, Qty=0)	Error: “Quantity must be greater than 0”	Error: “Quantity must be greater than 0”	**Pass**

TC_INV_003	Simulate supplier lookup for unavailable supplier	getSupplierInfo(SupplierID =S999)	Error: “Supplier not found”	Error: “Supplier not found”	**Pass**
TC_INV_004	Simulate supplier lookup for valid supplier	getSupplierInfo(SupplierID =S001)	Returns (S001, TechDistributors)	Returns (S001, TechDistributors)	**Pass**
TC_INV_005	Verify stock update after sales transaction	updateStock(ItemID=I1001 , SoldQty=5)	Stock updated: Remaining Qty = 45	Stock updated: Remaining Qty = 45	**Pass**
TC_INV_006	Validate reorder alert threshold	updateStock(ItemID=I1002 , Qty=5, Threshold=10)	Warning: “Reorder required”	**No warning displayed**	**Fail**

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for IMS using equivalence partitioning and BVA.
2. Identify two scenarios where stubs and drivers are needed for incomplete modules.
3. Execute all designed test cases and record actual results.
4. Create a defect report highlighting issues discovered during testing.
5. Write a summary analysis comparing expected vs. actual system behavior

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XI References

1. Chauhan, N. (2016). Software Testing: Principles and Practices. Oxford University Press.
2. ISBN: 9780198061847.
3. Singh, Y. (2012). Software Testing. Cambridge University Press. ISBN: 9781107652781.
4. Infosys Springboard – Software Testing Fundamentals.

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Correct design of test cases from specification	20 %
2	Appropriate use of Drivers and Stubs	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 3: Design test cases for Calculator Application to verify its functionality (Black Box Testing)

I Practical Significance

This practical demonstrates how to apply **Black Box Testing techniques** such as **Equivalence Class Partitioning (ECP)** and **Boundary Value Analysis (BVA)** to design and execute test cases for a simple calculator application. It enables learners to understand how to validate input–output behavior without accessing source code, ensuring correctness, accuracy, and completeness of arithmetic operations.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO1 - Explain various software testing methods..

IV Laboratory Learning Outcome(s)

LLO 3.1: Write test cases for a simple Calculator Application.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

A Calculator Application performs arithmetic operations such as addition, subtraction, multiplication, and division.

Black Box Testing focuses on validating the functional behavior of the application without considering its internal logic or code structure.

Testers verify whether inputs produce the expected outputs based on functional requirements.

Testing Techniques Used:

- Equivalence Class Partitioning (ECP) – Dividing inputs into valid and invalid ranges.
- Boundary Value Analysis (BVA) – Testing limits at input boundaries.
- Error Guessing – Predicting possible user input errors and testing them.

VII Recourses Required:

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Study the functional specifications of the calculator application.
2. Identify supported operations (Add, Subtract, Multiply, Divide).
3. Derive test cases for each operation using Black Box Testing principles.
4. Record inputs, expected outputs, and actual outputs in the test case table.
5. Execute each test case manually or using an automated script.
6. Compare expected vs. actual results and record *Pass/Fail* status.
7. Summarize results and analyze failed cases.

IX Example

The Calculator Application accepts two numerical inputs and performs one of the four basic arithmetic operations. It should display the correct result or an appropriate error message for invalid operations such as division by zero or character inputs.

Test Case ID	Objective	Steps (Input)	Expected Output	Actual Output	Status
TC_CAL_001	Verify addition of two positive integers	add(5, 3)	8	8	**Pass**
TC_CAL_002	Verify subtraction with negative result	subtract(4, 8)	-4	-4	**Pass**
TC_CAL_003	Verify multiplication with zero	multiply(6, 0)	0	0	**Pass**
TC_CAL_004	Validate division of two positive numbers	divide(10, 2)	5	5	**Pass**
TC_CAL_005	Verify division by zero handling	divide(10, 0)	Error: “Division by zero not allowed”	Error: “Division by zero not allowed”	**Pass**
TC_CAL_006	Verify incorrect output defect (simulation)	add(7, -2)	5	6 (**Bug identified**)	**Fail**

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases covering negative numbers and floating-point values.
2. Apply Boundary Value Analysis (BVA) for inputs near operational limits (e.g., ±9999).
3. Execute test cases for each arithmetic operation and record results.
4. Identify and log any defects or unexpected outputs.
5. Summarize test coverage and defect detection effectiveness.

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

XI References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard – Software Testing Fundamentals. Available at: https://infyspringboard.onwingspan.com
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks – Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Design of test cases using Black Box approach	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 4: Design test cases for Hostel Admission Form

I Practical Significance

This practical provides hands-on experience in form validation and functional testing through the design and execution of test cases for a Hostel Admission Form. Students will apply Black Box Testing techniques such as Equivalence Class Partitioning (ECP) and Boundary Value Analysis (BVA) to verify input field validations, mandatory entries, and correct form submission behavior. This experiment helps learners understand how to ensure data integrity, accuracy, and robustness in form-based user interfaces.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO1 - Explain various software testing methods.

IV Laboratory Learning Outcome(s)

LLO 4.1: Write test cases for Hostel Admission Form.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

A Hostel Admission Form typically collects student data such as Name, Roll Number, Department, Contact Information, Address, Gender, and Room Preferences.

Testing ensures that all fields are validated, mandatory fields cannot be left blank, and invalid data entries are handled appropriately.

Testing Techniques Used

- Equivalence Class Partitioning (ECP): Test valid and invalid inputs for each field.
- Boundary Value Analysis (BVA): Validate field limits (e.g., age range, character length).
- Error Guessing: Predict possible mistakes (e.g., wrong email format, special characters in name).

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Study the Hostel Admission Form fields and validation rules.
2. Identify all mandatory and optional fields.
3. Apply ECP and BVA to design test cases for valid and invalid inputs.
4. Record each test case with inputs, expected and actual results, and status.
5. Execute test cases manually in the web or desktop form.
6. Compare expected vs. actual outcomes and mark Pass/Fail.
7. Summarize findings and highlight validation defects.

IX Example

The Hostel Admission Form includes fields:

- Name
- Roll Number
- Department
- Mobile Number
- Email ID
- Gender
- Age
- Address

Each field must follow specific input rules (e.g., Mobile Number = 10 digits, Email = valid format, Age between 16–30).

Test Case ID	**Objective**	**Steps (Input)**	**Expected Output**	**Actual Output**	**Status**
TC_HA_001	Verify valid name entry	Enter “Riya Sharma”	Accepted	Accepted	**Pass**
TC_HA_002	Validate 10-digit number rule	Enter 9876543210	Accepted	Accepted	**Pass**
TC_HA_003	Check for valid email format	Enter riya.sharma@gmail.com	Accepted	Accepted	**Pass**

TC_HA_004	Validate invalid email format	Enter riya.sharma@com	Error: “Invalid Email Format”	Error: “Invalid Email Format”	**Pass**
TC_HA_005	Validate age boundary	Enter 15	Error: “Age must be between 16–30”	Error: “Age must be between 16–30”	**Pass**
TC_HA_006	Validate alphanumeric constraint	Enter 123@@@	Error: “Invalid Roll Number Format”	**Accepted (Bug detected)**	**Fail**

X Precautions

1. Verify all mandatory fields before submission.
2. Test field validations with both valid and invalid data.
3. Ensure database constraints match the validation rules.
4. Avoid reusing test data for multiple executions.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for fields such as Gender, Address, and Department.
2. Create negative test cases to test blank fields and invalid formats.
3. Execute the form with both valid and invalid data and record outputs.
4. Prepare a summary report listing all failed cases and reasons.
5. Recommend ways to improve validation effectiveness in future releases.

Space for Answer

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

……………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard – Software Testing Fundamentals. Available at: https://infyspringboard.onwingspan.com
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks – Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XIII Assessment Scheme (25 Marks)

Performance Indicators		**Weightage**
Process related (15 Marks)		**60 %**
1	Correct design of test cases for field validation	20 %
2	Effective use of ECP and BVA	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 5:* Design test cases for different tasks (OTP Verification, Image Upload) in any software module using Black Box Testing

I Practical Significance

This practical focuses on designing and executing Black Box Test Cases for two common application features — OTP (One-Time Password) Verification and Image Upload. It helps students apply Equivalence Class Partitioning (ECP) and Boundary Value Analysis (BVA) techniques to verify both data validation and functional correctness. These test cases simulate real-world software modules such as login systems and file upload services, emphasizing validation, user experience, and error handling.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality

III Course Level Learning Outcomes(s)

CO1 - Explain various software testing methods

IV Laboratory Learning Outcome(s)

LLO 5.1: Write test cases for different tasks (OTP Verification, Image Upload) using Equivalence Partitioning and Boundary Value Analysis of Black Box Testing.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

OTP Verification ensures that a user is authenticated through a randomly generated numeric code (usually 4–6 digits). Testing involves checking input formats, boundary conditions, timeouts, and invalid attempts.

Image Upload allows users to select and submit an image file. Testing focuses on file size limits, file formats, upload errors, and boundary validations.

Testing Techniques Used

- **Equivalence Class Partitioning (ECP)** — Identify valid and invalid data classes for OTP or file uploads.
- **Boundary Value Analysis (BVA)** — Test extreme input values (e.g., smallest/largest file sizes, shortest/longest OTP).

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Identify functionalities for OTP verification and image upload.
2. Apply Equivalence Class Partitioning (ECP) to classify valid and invalid inputs.
3. Apply Boundary Value Analysis (BVA) to test input limits (e.g., OTP length, file size).
4. Create test case tables with Inputs, Expected Outputs, and Actual Outputs.
5. Execute test cases manually on the target application.
6. Record results and determine Pass or Fail status.
7. Summarize and analyze results.

IX Example

A user logs in to an online **Student Admission Portal** that uses **two sequential modules**:

1. **OTP Verification Module** – to authenticate the user before uploading documents.
2. **Image Upload Module** – to upload a profile photo after successful OTP verification.

Test Case ID	Objective	Steps (Input)	Expected Output	Actual Output	Status
TC_COM_001	Verify valid OTP allows proceeding to upload module	Enter valid 6-digit OTP: 458912	OTP verified, “Proceed to Upload” button enabled	OTP verified, Upload option enabled	**Pass**
TC_COM_002	Verify invalid OTP blocks upload access	Enter invalid OTP: 123456	Error: “Invalid OTP” and Upload module locked	Error displayed, Upload disabled	**Pass**
TC_COM_003	Verify OTP expiry behavior	Enter valid OTP after timeout (3 mins)	Error: “OTP Expired”	Error displayed correctly	**Pass**
TC_COM_004	Verify valid image upload after	Upload profile.jpg (1.2MB)	“Image Uploaded Successfully”	“Image Uploaded Successfully”	**Pass**

	successful OTP	after OTP success			
TC_COM_005	Validate invalid file type upload (after valid OTP)	Upload resume.pdf	Error: “Unsupported File Type”	Error: “Unsupported File Type”	**Pass**
TC_COM_006	Verify upload bypass after failed OTP (defect scenario)	Enter invalid OTP → Upload profile.png	Should not allow upload (OTP invalid)	**Upload successful (Bug Detected)**	**Fail**

X Precautions

1. Always test boundary and invalid input combinations.
2. Use realistic OTPs and file sizes for practical testing.
3. Re-verify failed test cases after defect correction.
4. Ensure consistent test environments for repeated runs.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five new test cases combining OTP and image upload in one workflow.
2. Apply Equivalence Partitioning for invalid OTP input ranges.
3. Apply BVA for image size validation near the 5MB threshold.
4. Record all test executions and defect logs.

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks – Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 6: Test various functionalities of Railway Reservation System

I Practical Significance

This practical provides experience in designing and executing functional test cases for a Railway Reservation System (RRS). Students learn to apply both Black Box Testing techniques and requirement-based testing to ensure reliability, accuracy, and usability of an integrated software system. Testing the RRS helps validate major modules such as user registration, train search, ticket booking, fare calculation, seat availability, and payment processing — all critical for real-world software reliability.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO2 - Prepare test cases for different levels of testing

IV Laboratory Learning Outcome(s)

LLO 6.1**:** Write test cases for Railway Reservation System.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

A **Railway Reservation System (RRS)** allows users to search trains, check seat availability, book tickets, and process payments.

Testing ensures that:

- The system performs correctly for valid and invalid data.
- Errors and exceptions are handled properly.

Testing Techniques Applied:

- **Equivalence Class Partitioning (ECP)**: Test valid and invalid input data (e.g., train number, date, passenger details).
- **Boundary Value Analysis (BVA)**: Check ticket limits, age boundaries, and seat count limits.
- **Requirement-Based Testing**: Validate workflow as per the system specification.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Study the functional requirements of the Railway Reservation System (RRS).
2. Identify major modules such as Login, Search Train, Seat Availability, Fare Calculation, and Booking.
3. Apply ECP and BVA techniques to derive valid and invalid test data.
4. Document test cases with steps, expected and actual outputs, and status.
5. Execute test cases manually or using automation scripts.
6. Record and analyze results to detect errors or inconsistencies.
7. Summarize the outcome of testing with observations.

IX Example

A registered user logs into the Railway Reservation System, searches for a train between two stations, checks seat availability, and proceeds to book tickets. The system must validate all inputs, compute fares correctly, handle invalid data gracefully, and ensure seat allocation limits are respected.

Test Case ID	**Objective**	**Steps (Input)**	**Expected Output**	**Actual Output**	**Status**
TC_RRS_001	Verify train search with valid route	From: *Pune*, To: *Mumbai*, Date: *12/10/2025*	List of available trains displayed	List displayed successfully	**Pass**
TC_RRS_002	Validate invalid route input	From: *Pune*, To: *XYZ*	Error: “Invalid Destination”	Error displayed correctly	**Pass**
TC_RRS_003	Check seat availability within limit	Train No: *11009*, Class: *Sleeper*	“Seats Available: 25”	Seats shown correctly	**Pass**
TC_RRS_004	Verify booking with valid passenger details	Name: *Riya Shah*, Age: *25*, Class: *AC 3-Tier*	“Ticket Booked Successfully” with PNR	Ticket booked successfully	**Pass**

TC_RRS_ 005	Validate fare for senior citizen discount	Age: *65*, Route: *Pune–Delhi*	Fare: 850 (after discount)	Fare: 850	**Pass**
TC_RRS_ 006	Validate exceeding ticket booking limit	Try booking 7 tickets in one transaction	Error: “Maximum 6 passengers allowed”	**System allows 7 tickets (Bug Detected)**	**Fail**

X Precautions

1. Verify that all input fields are tested for valid and invalid entries.
2. Ensure date and route validations match real-world rules.
3. Check database consistency after each successful booking.
4. Re-test failed cases after defect correction.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for passenger detail and payment modules.
2. Apply Equivalence Class Partitioning (ECP) for route and class selection inputs.
3. Apply Boundary Value Analysis (BVA) for ticket quantity validation.

Space for Answer

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XIII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Application of ECP and BVA in system testing	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 7: *Validate login procedure for E-Commerce Application (Flipkart or Amazon)

I Practical Significance

This practical introduces functional and boundary testing techniques for validating the login process in an E-Commerce Application (such as Flipkart or Amazon). It focuses on ensuring that valid credentials grant access, while invalid, missing, or malformed inputs are handled securely. This practical also helps learners understand how to apply Equivalence Class Partitioning (ECP) and Boundary Value Analysis (BVA) in Black Box Testing for authentication modules.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO2 - Prepare test cases for different levels of testing

IV Laboratory Learning Outcome(s)

LLO 7.1**:** Prepare test cases for E-Commerce Login Form.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently

VI Relevant Theoretical Background

A **Login Form** is the entry point to most E-Commerce platforms and includes essential input fields such as **Email / Mobile Number** and **Password**.

Testing ensures that:

- Users with valid credentials can successfully log in.
- Invalid, empty, or malformed inputs are rejected with appropriate error messages.
- Security measures (like CAPTCHA, timeout, and encryption) function as intended.

Testing Techniques Used

- **Equivalence Class Partitioning (ECP):** Divides inputs into valid and invalid categories (e.g., correct vs. incorrect password).
- **Boundary Value Analysis (BVA):** Validates input length limits for email and password.
- **Error Guessing:** Identifies potential vulnerabilities (e.g., SQL injection, blank fields).

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Open the Flipkart or Amazon login page.
2. Study the fields, their validation rules, and constraints.
3. Identify input equivalence classes for Email/Mobile Number and Password.
4. Apply ECP and BVA to design valid and invalid test cases.
5. Execute the test cases manually and record the results.
6. Observe error messages, redirection, and authentication behavior.
7. Document outcomes as *Pass* or *Fail* and summarize findings.

IX Example

A user attempts to log in to an **E-Commerce Portal** (e.g., Flipkart/Amazon) using **Email/Mobile Number** and **Password**.

The system must:

- Accept valid credentials and redirect to the user dashboard.
- Reject incorrect credentials with descriptive error messages.
- Handle blank and invalid inputs gracefully without crashing.

Test Case ID	**Objective**	**Steps (Input)**	**Expected Output**	**Actual Output**	**Status**
TC_ECOM_001	Verify login with valid email and password	Email: riya@gmail.com, Password: Riya@123	Login successful → Redirect to dashboard	Login successful	**Pass**
TC_ECOM_002	Validate login with invalid email format	Email: riyagmail.com, Password: Riya@123	Error: “Invalid email address”	Error displayed correctly	**Pass**
TC_ECOM_003	Validate password length	Email: riya@gmail.com, Password: Riya1	Error: “Password must be at	Error displayed correctly	**Pass**

	boundary (min 8 chars)		least 8 characters”		
TC_ECOM_004	Verify incorrect password handling	Email: riya@gmail.com, Password: Wrong@123	Error: “Incorrect password”	Error: “Incorrect password”	**Pass**
TC_ECOM_005	Verify blank input field validation	Email: *(blank)*, Password: Riya@123	Error: “Email cannot be empty”	Error displayed correctly	**Pass**
TC_ECOM_006	Check SQL injection vulnerability	Email: admin' OR 1=1 --, Password: test	Error: “Invalid credentials”	**Login successful (Bug Detected)**	**Fail**

X Precautions

1. Use only authorized test accounts (never real user credentials).
2. Avoid multiple rapid login attempts to prevent system lockouts.
3. Verify browser compatibility for input validations.
4. Log out after every successful session.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for login fields using ECP and BVA.
2. Test the system’s response for multiple failed login attempts (e.g., lockout feature).
3. Apply negative testing for empty, whitespace, and invalid credentials.
4. Record the expected vs. actual outputs in your test log.

Space for Answer

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XIII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	**60 %**
1	Application of ECP and BVA for login validation	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 8: Test functionality of web pages of any website.

I Practical Significance

This practical introduces web application testing concepts with a focus on verifying functionality, navigation, responsiveness, and form behavior of any website. Students will learn to design and execute Black Box Test Cases for website components such as navigation menus, hyperlinks, input forms, and multimedia content. This exercise ensures that all visible and interactive elements of a website perform as intended.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality

III Course Level Learning Outcomes(s)

CO2 - Prepare test cases for different levels of testing.

IV Laboratory Learning Outcome(s)

LLO 8.1: Write test cases for web page testing of any website.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Web Page Testing focuses on validating functional, navigational, and visual aspects of a website.
Testing ensures:

- Links and buttons work correctly.
- Forms accept valid input and reject invalid data.
- Pages load within acceptable time.
- Design remains consistent across devices and browsers.

Testing Techniques Used:

- Black Box Testing: To validate functionality without accessing the code.
- Equivalence Class Partitioning (ECP): To test valid and invalid inputs.
- Boundary Value Analysis (BVA): For field limits (e.g., character input, file upload size).

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Select a live or local website to be tested (e.g., Wikipedia, institutional site).
2. Identify the major pages and interactive elements (links, forms, buttons).
3. Apply **Black Box Testing** to test navigation and field validation.
4. Design and document test cases covering positive and negative scenarios.
5. Execute test cases manually on multiple browsers.
6. Record *Expected Output*, *Actual Output*, and *Status* (Pass/Fail).
7. Summarize overall website functionality and observations.

IX Example

The tester evaluates core functionalities of the **Wikipedia home page**, including search operations, navigation menus, and language selection features. The objective is to verify form submission, hyperlink navigation, and user interface behavior under various conditions.

Test Case ID	Objective	Steps (Input)	Expected Output	Actual Output	Status
TC_WEB_001	Verify website loads successfully	Open https://www.wikipedia.org	Homepage displayed correctly	Homepage displayed successfully	**Pass**
TC_WEB_002	Verify link to English Wikipedia	Click on “English” language link	Redirects to https://en.wikipedia.org	Redirects correctly	**Pass**
TC_WEB_003	Validate search with valid term	Enter Software Testing and click search	Results page displays relevant articles	Relevant articles displayed	**Pass**

TC_WEB_004	Validate search with blank input	Leave search box empty → Click search	Error or no result message displayed	No message displayed (page reloads silently)	**Fail**
TC_WEB_005	Validate contact form (if present)	Enter valid name, email, and message	Confirmation or success message displayed	Confirmation message displayed	**Pass**
TC_WEB_006	Check for broken hyperlink	Click on "Privacy Policy" link	Opens correct Privacy Policy page	Opens correct page	**Pass**

X Precautions

1. Always test authorized and public pages only.
2. Clear browser cache before each new test execution.
3. Use multiple browsers (Chrome, Edge, Firefox) for comparison.
4. Re-verify failed test cases after defect correction.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for testing different web elements (buttons, images, dropdowns, forms).
2. Validate website responsiveness on mobile and desktop views.
3. Perform negative testing for invalid inputs in search or forms.
4. Execute web page tests using at least two browsers and record observations

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XI References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Application of Black Box Testing for web validation	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			Dated Signature of Teacher
Process-related Assessment 15 marks	Product-related Assessment 10 marks	Total (25 marks)	

Practical No. 9:* Design Test Cases for Control and Decision-Making Statements (Use C Language).

I Practical Significance

This practical enables students to test fundamental control and decision-making constructs in the C language through effective test case design.By applying White Box Testing and Black Box Testing principles, students will understand how program logic, iteration limits, and condition branches affect software behavior.The objective is to verify whether each control statement produces the correct output for given conditions and handles edge cases properly.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO2 - Prepare test cases for different levels of testing

IV Laboratory Learning Outcome(s)

LLO 9.1: Design test cases for for...loop
LLO 9.2: Design test cases for do...while loop
LLO 9.3: Design test cases for switch case
LLO 9.4: Design test cases for if...else statements

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Control and decision-making statements guide program execution flow:

- For Loop: Executes code repeatedly for a defined number of iterations.
- Do...While Loop: Executes at least once, then repeats while a condition is true.
- Switch Case: Executes one block based on a specific value or condition.
- If...Else: Executes conditional blocks based on logical expressions.

Testing Techniques Used:

- White Box Testing: For loop/branch coverage and path validation.
- Black Box Testing: For input–output validation based on conditions.
- Boundary Value Analysis (BVA): For testing iteration and condition limits.
- Equivalence Class Partitioning (ECP): To test valid and invalid inputs.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.), Turbo C Compiler		

VIII Procedure

1. Write small C programs using different control statements.
2. Identify inputs, expected outputs, and test conditions.
3. Design test cases using BVA and ECP.
4. Execute programs for each test case.
5. Compare expected vs. actual outputs and record results.
6. Analyze failed cases and fix logical errors.

IX Example

A. For Loop Example

Program:

```
#include <stdio.h>
int main() {
   int i;
   for (i = 1; i <= 5; i++)
      printf("%d ", i);
   return 0;
}
```

Test Cases for For Loop

Test Case ID	**Objective**	**Input**	**Expected Output**	**Actual Output**	**Status**
TC_FOR_001	Verify loop executes 5 times	i = 1 to 5	1 2 3 4 5	1 2 3 4 5	**Pass**
TC_FOR_002	Check boundary value at 0	i = 0 to 5	0 1 2 3 4 5	0 1 2 3 4 5	**Pass**
TC_FOR_003	Verify loop exit condition	i = 6	No output	No output	**Pass**
TC_FOR_004	Detect off-by-one error (simulation)	i = 1 to 6	1 2 3 4 5	1 2 3 4 5 6 (**Bug**)	**Fail**

B. Do...While Loop Example

Program:

```
#include <stdio.h>
int main() {
   int i = 1;
   do {
      printf("%d ", i);
      i++;
   } while (i <= 3);
   return 0;
}
```

Test Cases for Do...While Loop

Test Case ID	Objective	Input	Expected Output	Actual Output	Status
TC_DW_001	Verify loop executes minimum once	i = 1	1 2 3	1 2 3	**Pass**
TC_DW_002	Validate exit condition	i = 4	4	4	**Pass**
TC_DW_003	Detect incorrect condition	i = 1	1 2 3 4 (**extra iteration**)	1 2 3 4	**Fail**

C. Switch Case Example

Program:

```
#include <stdio.h>
int main() {
   int choice = 2;
   switch (choice) {
      case 1: printf("Option 1"); break;
      case 2: printf("Option 2"); break;
      case 3: printf("Option 3"); break;
      default: printf("Invalid choice");
   }
   return 0;
}
```

Test Cases for Switch Case

Test Case ID	Objective	Input	Expected Output	Actual Output	Status
TC_SW_001	Verify correct case execution	choice = 2	Option 2	Option 2	**Pass**
TC_SW_002	Verify default case execution	choice = 5	Invalid choice	Invalid choice	**Pass**
TC_SW_003	Detect missing break defect	choice = 1	Option 1	Option 1 Option 2 (**Bug**)	**Fail**

D. If...Else Example

Program:

```
#include <stdio.h>
int main() {
    int num = 10;
    if (num % 2 == 0)
        printf("Even");
    else
        printf("Odd");
    return 0;}
```

Test Cases for If...Else Statement

Test Case ID	**Objective**	**Input**	**Expected Output**	**Actual Output**	**Status**
TC_IF_001	Check even number condition	num = 10	Even	Even	**Pass**
TC_IF_002	Check odd number condition	num = 7	Odd	Odd	**Pass**
TC_IF_003	Detect inverted logic (simulation)	num = 8	Even	Odd (**Bug**)	**Fail**

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write a C program for factorial calculation using for loop and test it.
2. Design five new test cases for do...while to test condition variations.
3. Create a decision table for a switch case structure.
4. Execute all control statement programs and record results.

Space for Answer

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

XI References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Application of testing on control structures	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 10: *Design test cases for Online Mobile Recharge..

I Practical Significance

This practical focuses on designing and executing functional test cases for an Online Mobile Recharge System(like Paytm, PhonePe, or Jio).Students learn how to apply Black Box Testing, Equivalence Class Partitioning (ECP), and Boundary Value Analysis (BVA) to verify critical components such as mobile number validation, operator selection, recharge amount limits, and payment functionality.This exercise emphasizes accuracy, security, and reliability of transaction-based systems.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO2 - Prepare test cases for different levels of testing.

IV Laboratory Learning Outcome(s)

LLO 10.1: Prepare test cases for online mobile recharge.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

An **Online Mobile Recharge System** allows users to enter their mobile number, select an operator, input a recharge amount, and process payment.

Testing ensures that:

- Valid inputs result in successful recharge.
- Invalid entries (wrong number, invalid amount, etc.) are handled properly.
- Payment gateway interactions work reliably and securely.

Testing Techniques Used:

- **Black Box Testing:** Focused on functional validation without accessing code.
- **Equivalence Class Partitioning (ECP):** For valid/invalid mobile numbers and recharge amounts.
- **Boundary Value Analysis (BVA):** For minimum/maximum recharge limits.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Open any online recharge platform (Paytm / PhonePe / Jio).
2. Study the input fields and business rules (mobile number, operator, amount).
3. Apply ECP and BVA to design valid and invalid test cases.
4. Record each test case with expected and actual outcomes.
5. Execute the test cases manually in the test environment.
6. Compare expected vs. actual results and mark Pass/Fail.
7. Summarize findings and identify any system defects.

IX Example

A user wants to recharge their prepaid mobile number using an online recharge portal. The form requires a 10-digit mobile number, operator selection, and recharge amount (₹10–₹1500). The system must validate all inputs and confirm successful recharge or display appropriate error messages**.**

Test Case ID	**Objective**	**Steps (Input)**	**Expected Output**	**Actual Output**	**Status**
TC_RECH_001	Verify recharge with valid mobile number	Enter 9876543210, Operator: *Jio*, Amount: *₹199*	Recharge successful message	Recharge successful	Pass
TC_RECH_002	Validate recharge with invalid mobile number (<10 digits)	Enter 98765, Operator: *Airtel*, Amount: *₹100*	Error: “Enter valid 10-digit mobile number”	Error displayed correctly	Pass
TC_RECH_003	Validate boundary value – minimum amount	Enter valid number, Amount: *₹10*	Recharge successful	Recharge successful	Pass

TC_RECH_004	Validate boundary value – exceeding maximum amount	Enter valid number, Amount: *₹2000*	Error: “Maximum amount is ₹1500”	Error displayed correctly	Pass
TC_RECH_005	Validate failed transaction	Enter valid details but simulate payment failure	Error: “Payment failed, please try again”	Error displayed correctly	Pass
TC_RECH_006	Detect missing validation defect (simulation)	Enter *blank* mobile number, Amount: *₹100*	Error: “Mobile number required”	Recharge processed (Bug Detected)	Fail

X Precautions

1. Use dummy accounts or test environments for recharge simulation.
2. Never use real bank or UPI credentials in testing.
3. Ensure proper validation of all fields before clicking submit.
4. Record timestamps and transaction IDs for each execution.

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five new test cases for recharge using different operators.
2. Perform negative testing for blank and special character inputs.
3. Execute each test case and record expected vs. actual outcomes.
4. Identify at least one defect and suggest a fix.

Space for Answer

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………..

…………………………………………………………………………………………………

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

………………………………………………………………………………………………..

……………………………………………………………………………………………….

………………………………………………………………………………………………..

XI References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at:

https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XII Assessment Scheme (25 Marks)

	Performance Indicators	**Weightage**
	Process related (15 Marks)	**60 %**
1	Application of ECP and BVA in recharge module testing	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 11: *Design test cases for Flight Ticket Booking System.

I Practical Significance

This practical introduces students to the design and execution of functional test cases for a Flight Ticket Booking System, similar to real-world platforms like MakeMyTrip, IRCTC Air, Goibibo, or EaseMyTrip.Students will apply Black Box Testing, Equivalence Class Partitioning (ECP), and Boundary Value Analysis (BVA) to validate the booking workflow — including passenger data entry, flight selection, seat class, payment, and ticket confirmation.The focus is on ensuring accuracy, data validation, and smooth transaction flow across all modules.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO2 - Prepare test cases for different levels of testing.

IV Laboratory Learning Outcome(s)

LLO 11.1: Prepare test cases for Flight Ticket Booking System.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

A **Flight Ticket Booking System** enables users to search for flights, select dates, choose class, enter passenger details, and make payments.

Testing ensures that:

- Input validation is properly implemented.
- Flight and fare details are accurate.
- Payment and ticket generation processes work seamlessly.

Testing Techniques Used:

- **Black Box Testing:** To validate system functionality without internal code access.
- **Equivalence Class Partitioning (ECP):** For valid and invalid inputs (e.g., departure/arrival cities, dates, fare amounts).
- **Boundary Value Analysis (BVA):** For input limits like maximum passengers and date ranges.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Open the flight booking application (real or demo).
2. Identify major modules — search flight, select flight, passenger entry, payment, confirmation.
3. Analyze input fields (source, destination, date, passenger count, payment method).
4. Apply ECP and BVA to design valid and invalid test cases.
5. Record expected and actual results for each case.
6. Execute cases manually.

IX Example

A user visits an online Flight Booking Portal to book a one-way flight from Pune to Delhi. The booking process includes entering travel details, selecting flights, adding passenger information, confirming fare, and making a payment. The system should validate all input fields, ensure correct fare calculation, and generate an e-ticket after successful payment.

Test Case ID	**Objective**	**Steps (Input)**	**Expected Output**	**Actual Output**	**Status**
TC_FLT_001	Verify valid route and date	From: *Pune*, To: *Delhi*, Date: *15/10/2025*	List of available flights displayed	Flights displayed correctly	**Pass**
TC_FLT_002	Validate invalid route	From: *Pune*, To: *XYZ*	Error: “Invalid destination”	Error displayed correctly	**Pass**
TC_FLT_003	Validate age and name fields	Name: *Riya Shah*, Age: *25*	Passenger data accepted	Passenger data accepted	**Pass**
TC_FLT_004	Check boundary for passenger count	Passengers = *6*	Booking successful	Booking successful	**Pass**
TC_FLT_005	Simulate payment failure	Enter valid data → Payment fails	Error: “Payment failed, please retry”	Error displayed correctly	**Pass**

TC_FLT_006	Detect ticket generation without payment	Enter valid data → Skip payment	Ticket should not be generated	**Ticket generated (Bug Detected)**	**Fail**

X Precautions

1. Ensure all mandatory fields are validated.
2. Avoid testing on live financial systems.
3. Use mock payment or sandbox environments.
4. Retest failed cases after bug resolution.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design five additional test cases for round-trip booking validation.
2. Apply BVA to check limits for passengers (1–6).
3. Perform negative testing for invalid city names and expired dates.
4. Record expected vs. actual results in tabular format.

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at: https://www.geeksforgeeks.org/difference-between-sanity-testing-and-smoke-testing/

XIII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Application of ECP and BVA in booking test cases	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 12: Design Test Plan and Test Cases for Elevator.

I Practical Significance

This practical introduces the concept of Test Planning for real-world embedded control systems, using the example of an Elevator (Lift) Control System. Students learn to develop a structured Test Plan, design test cases, and apply Black Box and White Box Testing techniques to validate system functionality, safety, and control logic. Elevator systems involve hardware–software interaction, making this practical crucial for understanding system integration testing and safety-critical validation.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO3 - Prepare test plan for a given application.

IV Laboratory Learning Outcome(s)

LLO 12.1: Write test plan and test cases for Elevator System.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

An **Elevator System** is a safety-critical control system that moves between floors based on user input. The system must correctly handle inputs (floor selection, door operations) and outputs (motor movement, indicator signals).

Testing ensures that:

- The elevator responds correctly to button inputs.
- Door operations synchronize with movement.
- Overload, emergency stop, and boundary conditions are safely handled.

Testing Techniques Applied:

- **Black Box Testing:** Validating system behavior based on input–output.
- **White Box Testing:** Verifying control flow logic and safety constraints.
- **State Transition Testing:** Ensuring proper state changes (idle → moving → door open).

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Study the functional and safety specifications of the elevator system.
2. Identify input conditions (button presses, sensor triggers).
3. Prepare a structured test plan including objectives, scope, and techniques.
4. Design test cases based on functional, boundary, and state conditions.
5. Execute tests on simulation or hardware model.
6. Record outputs, compare with expected results, and mark pass/fail.
7. Document findings and suggest corrective actions.

IX Example

The elevator system operates between **5 floors (1–5)**. It responds to floor selection, door open/close commands, overload, and emergency stop signals. The elevator must not move when doors are open, overloaded, or under emergency conditions.

Test Case ID	**Objective**	**Input / Action**	**Expected Output**	**Actual Output**	**Status**
TC_ELV_001	Verify valid floor selection	Press “3” while at floor 1	Elevator moves upward to 3	Moves upward correctly	**Pass**
TC_ELV_002	Validate invalid floor request	Press “6” (out of range)	Error: “Invalid Floor”	Error displayed correctly	**Pass**
TC_ELV_003	Verify door closes before movement	Press “2” when door open	Elevator waits until door closes, then moves	Works as expected	**Pass**
TC_ELV_004	Check system halts under overload	Simulate 600kg load (limit: 500kg)	Elevator remains idle, alarm triggered	Alarm triggered correctly	**Pass**
TC_ELV_005	Validate emergency stop functionality	Press “Stop” while moving	Elevator halts immediately, alarm sounds	Elevator halts, alarm active	**Pass**

TC_ELV_006	Detect door movement defect (simulation)	Door opens during movement	Should lock door during motion	**Door opens mid-move (Bug Detected)**	**Fail**

Test Plan for Elevator System

Component	**Description**
Test Plan ID:	TP_ELEV_001
Tested System:	Elevator Control System
Prepared By:	[Student Name]
Version:	1.0
Date:	[Insert Date]
Objective:	To verify correct functional and safety behavior of the elevator system under various operating conditions.
Scope:	The test covers door control, floor selection, direction indication, overload, and emergency stop operations.
Test Environment:	Simulated elevator control program / embedded test rig
Testing Techniques:	Black Box, White Box, State Transition Testing
Resources Required:	PC, Elevator simulation software, documentation
Entry Criteria:	Elevator control program is developed and deployable.
Exit Criteria:	All critical test cases executed with no severe defects remaining.
Test Deliverables:	Test Plan Document, Test Case Report, Test Log, Defect Report
Risks / Constraints:	Hardware unavailability, timing issues, emergency conditions.
Responsibilities:	Tester – Executes cases, records results; Supervisor – Validates test process.

X Precautions

1. Always ensure simulated environment safety when testing movement logic.
2. Verify correct timing between door, motor, and indicator signals.
3. Avoid running multiple elevator commands simultaneously.
4. Re-test after any modification in control logic.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Apply Boundary Value Analysis (BVA) to test overload and floor limits.
2. Prepare a State Transition Diagram for elevator operation (Idle → Moving → Door Open).
3. Execute all test cases and record observations.
4. Generate a Test Summary Report including results, issues, and corrective actions.

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163
5. GeeksforGeeks–Sanity vs Smoke Testing. Available at:

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	**60 %**
1	Preparation of structured test plan	20 %
2	Design and execution of test cases	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Pactical No. 13: Design Test Plan and Test Cases for Notepad (MS Window-based) Application.

I Practical Significance

This practical focuses on preparing a formal Test Plan and designing functional and GUI-based test cases for the Notepad Application in Windows OS. The goal is to understand how to test desktop-based applications, verifying their core operations such as creating, editing, saving, and formatting text files. Students will apply Black Box Testing, GUI Testing, and Functional Testing techniques to ensure software correctness, usability, and reliability.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO3 - Prepare test plan for a given application.

IV Laboratory Learning Outcome(s)

LLO 13.1: Write test plan and test cases for Notepad Application.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Notepad is a basic text editor that supports simple file operations and text formatting. Testing Notepad involves validating both functional and GUI components to ensure proper file management and user interaction.

Modules to Test:

- File Menu: New, Open, Save, Save As, Exit
- Edit Menu: Cut, Copy, Paste, Undo, Redo
- Format Menu: Word Wrap, Font
- Help Menu: About Notepad

Testing Techniques Used:

- **Black Box Testing:** To verify visible functionalities (inputs/outputs).
- **GUI Testing:** To validate the look and behavior of buttons, menus, and dialogs.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.), Notepad		

VIII Procedure

1. Launch Notepad and observe the interface components.
2. Identify main menus, submenus, and dialog boxes.
3. Prepare a test plan including objectives, scope, and techniques.
4. Design functional and GUI test cases based on features.
5. Execute each test case and record actual outcomes.
6. Compare with expected results and mark *Pass/Fail*.
7. Prepare test summary and suggest improvements.

IX Example

Test the main functionalities of **Notepad**, including file creation, saving, editing, and formatting. Validate that the interface behaves as expected under normal and abnormal conditions.

Test Case ID	Objective	Steps (Input)	Expected Output	Actual Output	Status
TC_NPAD_001	Verify new file creation	Click *File* → *New*	Blank new document opens	Blank document opens successfully	**Pass**
TC_NPAD_002	Validate saving a new file	Type "Hello World" → Save as test.txt	File saved successfully	File saved successfully	**Pass**
TC_NPAD_003	Verify undo functionality	Type "Hello" → Undo	Text cleared	Text cleared	**Pass**
TC_NPAD_004	Check word wrap toggling	Enable Word Wrap → Type long text	Text wraps to next line	Text continues on same line (**Bug**)	**Fail**
TC_NPAD_005	Verify "About" dialog opens	Click *Help* → *About Notepad*	About dialog displayed	Dialog displayed	**Pass**

TC_NPAD_006	Validate exit operation	Click *File → Exit*	Application closes gracefully	Application closes correctly	**Pass**

Test Plan for Notepad Application

Component	**Description**
Test Plan ID:	TP_NPAD_001
System Under Test (SUT):	Notepad (Windows-based Application)
Version:	1.0
Prepared By:	[Student Name]
Date:	[Insert Date]
Objective:	To verify functionality, usability, and reliability of Notepad application features.
Scope:	File operations, text editing, formatting, and GUI validation.
Testing Techniques:	Functional, GUI, and Black Box Testing
Test Environment:	Windows 10 / 11 OS
Tools Required:	Microsoft Notepad
Entry Criteria:	Application installed and operational.
Exit Criteria:	All test cases executed; critical defects resolved.
Deliverables:	Test Plan Document, Test Case Report, Test Summary
Risks / Constraints:	OS compatibility issues, unexpected crashes.
Responsibilities:	Tester – Executes test cases; Instructor – Reviews results.

X Precautions

1. Save all work before executing “Exit” test cases.
2. Ensure testing is done on the correct Windows OS version.
3. Validate the same functionality multiple times to detect intermittent issues.
4. Avoid using restricted folders for saving files.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. What are the key components that should be tested in the Notepad application?
2. Design a test case to verify “Save As” functionality for existing files.
3. How does GUI testing differ from functional testing?
4. Explain the importance of test planning in desktop application testing.
5. Suggest negative test cases for file saving and opening invalid files.

Space for Answer

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII References

1. Chauhan, N. (2016). *Software Testing: Principles and Practices.* Oxford University Press. ISBN: 9780198061847.
2. Singh, Y. (2012). *Software Testing.* Cambridge University Press. ISBN: 9781107652781.
3. Infosys Springboard–SoftwareTestingFundamentals.
4. NPTEL – Software Testing Course. Available at: https://nptel.ac.in/courses/106101163

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		**60 %**
1	Preparation of structured test plan	20 %
2	Design and execution of test cases	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		**40 %**
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		**100 %**

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 14: Prepare test report for any website.

I Practical Significance

This practical focuses on preparing test report for any website. A well-written test report highlights the defects found, test coverage, unresolved issues, and potential risks, which assists the team in making informed decisions about fixes, improvements, and release readiness. It also serves as an important reference document for future testing cycles, audits, and maintenance activities.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO3 - Prepare test plan for a given application.

IV Laboratory Learning Outcome(s)

LLO 14.1 Create test report of executed test cases for any website.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

A test report is an essential document in software testing that summarizes the overall testing activities, processes, and results of a software application or website. It provides clear and concise information about what was tested, how it was tested, and what outcomes were achieved. The main purpose of a test report is to communicate the quality of the software to stakeholders such as developers, project managers, and clients. It helps them understand whether the website meets the defined requirements, functions as expected and is ready for deployment. A well-prepared test report includes details such as test objectives, test scope, features tested, features not tested, test environments, test execution results, defect details, risks, and recommendations. It ensures transparency in the testing process and allows stakeholders to make informed decisions. Test reports also help track progress, identify trends in defects, improve future testing cycles, and maintain overall software quality.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Define the purpose and scope of website testing.
2. Gather all executed test cases and results.
3. Document test environment, browse and tools used.
4. Note passed, failed, and skipped test cases.
5. Finalize the test Report.

IX Example

Project: ABC Online Shopping Website
Tested By: John Doe
Test Date: 28-Oct-2025
Environment: Windows 10, Chrome v120, 50 Mbps

- **Test Summary Report Identifier :**

TSR-ONLINE-SHOP-2025-01 – Unique ID for the test summary report.

- **Summary:**

Testing covered major modules including registration, login, product search, cart, checkout, payment, and order confirmation to ensure smooth functionality and usability.

- **Variances:**

Minor delays occurred due to payment gateway issues and server downtime. Some low-priority UI and older-browser tests were deferred.

- **Comprehensiveness:**

High and medium priority features received full coverage. Critical flows (authentication to order generation) were thoroughly tested; non-critical areas like wishlist and reviews received limited testing.

- **Summary of Results:**

Most tests passed. Issues found included image loading problems, discount calculation errors, and occasional slow performance. Critical bugs were fixed; minor cosmetic issues deferred.

- **Evaluation:**

The website is stable, user-ready, and meets quality standards. No major defects remain open.

- **Activities Summary:**

Activities included requirement review, test planning, case design, environment setup, functional/UI/security testing, defect management, retesting, regression, and some automation for repetitive tasks.

- **Approvals:**

QA Lead, Test Manager, and Project Manager approved the report. The system is recommended for production release.

X Precautions

1. Note down the expected output for each test case carefully and generate test report.
2. Select required test methodology and module(s).

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Prepare a test report for any website of your choice (e.g., online shopping, college portal, or banking site).
2. Create a table containing Test case ID, Test case Objective, Input data, Expected result, Actual result, and Status for at least five test cases.
3. Calculate the pass percentage if 4 out of 6 test cases have passed.

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII References

1. https://testsigma.com/blog/test-plan/
2. https://www.geeksforgeeks.org/test-plan-software-testing/
3. https://www.pubnub.com/how-to/test-cases-for-chat-application/
4. https://www.softwaretestinghelp.com
5. http://spojtoolkit.com/TestCaseGenerator/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test report design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed test report	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 15: *Design test cases and test summary report for a travel booking application.

I Practical Significance

This practical focuses on designing test cases and preparing a test summary report for a travel booking application it ensures the reliability and correctness of all functionalities, such as flight searches, hotel bookings, payments, and cancellations. It validates that the application meets user requirements, reduces the risk of failures, and improves overall quality and user experience. The documented test cases and summary reports provide clear evidence of testing for stakeholders, support future maintenance and upgrades through reusable test cases and enhance confidence in the application's stability and performance.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO3 - Prepare test plan for a given application.

IV Laboratory Learning Outcome(s)

LLO 15.1 Prepare test cases and test summary report for a travel booking application.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Designing test cases and preparing a test summary report for a travel booking application requires a structured software testing approach to ensure the reliability, usability, and functionality of the system. A travel booking application typically involves multiple integrated modules such as user registration, login, search for destinations, booking of flights or hotels, payment processing, and confirmation generation. Each module must be thoroughly tested through well-designed test cases that define test inputs, steps, preconditions, and expected results. Test case design is based on requirement analysis, where testers identify functional and non-functional requirements to create valid, invalid, and boundary scenarios ensuring complete test coverage. Techniques such as equivalence partitioning, boundary value analysis, decision tables, and use-case-based testing are applied to derive effective test cases. After executing these test cases, a test summary report is prepared to present the overall testing activities and results. The test summary report includes details such as the number of test cases executed, passed, failed, defects discovered, testing environment, scope, limitations, and the overall quality assessment of the travel booking application. This report provides transparency, helps stakeholders evaluate system readiness, and supports decision-making for the release of the application.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Define the purpose and scope of travel booking application.
2. Gather all executed test cases and results.
3. Note passed, failed, and skipped test cases.
4. Include bug IDs, severity, and status.
5. Compare expected vs. actual outcomes.
6. Finalize the summary Report.

IX Example

Test cases for Travel Booking Application for login Module.

Test case ID	**Testcase objective**	**Input data**	**Expected result**	**Actual result**	**Status**
TC1	Login with valid credentials	Enter valid email & password → Click Login	User should successfully log in and redirect to dashboard	User successfully log in and redirect to dashboard	Pass
TC2	Login with invalid password	Enter valid email & invalid password → Click Login	Error message “Invalid credentials” should appear	Error message “Invalid credentials” appeared	Pass

TC3	Login with empty fields	Leave fields blank → Click Login	Validation message should appear for required fields	Validation message appeared for required fields	Pass
TC4	Forgot password functionality	Click “Forgot Password” → Enter email → Submit	Reset link send to email	Reset link send to email was delayed	Fail
TC5	Session timeout	Log in → Stay idle 20 mins	Auto logout and message displayed	Auto logout and message displayed	Pass

- **Test Summary Report Identifier :**

TSR-TRAVEL-LOGIN-2025-01– Unique ID for the test summary report.

1. **Summary:**
   Testing of the Login Module verified secure login, validations, error messages, session handling, and forgot-password behavior. Five test cases were executed: valid login, invalid login, empty fields, forgot password, and session timeout.
2. **Variances:**
   All tests ran as planned except session timeout, which required extra time due to extended server session settings.
3. **Comprehensiveness:**
   Complete coverage was achieved for all high- and medium-priority scenarios, including validation, error handling, and session management.
4. **Results:**
   - **4 passed:** valid login, invalid login, empty fields, session timeout
   - **1 failed:** forgot password (delay in reset email) Defect logged; retest scheduled for next build.
5. **Evaluation:**
   Module is stable, secure, and functionally reliable. Only one minor defect remains. Ready for integration with Search and Booking modules.
6. **Activities:**
   Requirement review, test case design, environment setup, execution of five tests, defect logging, session timeout verification, and browser checks (Chrome & Firefox).
7. **Approvals:**
   Approved by Test Lead, QA Manager, and Project Manager for integration and further testing.

X Precautions

1. Note down the expected output for each test case carefully and generate summary report.
2. Select required test methodology and module(s).

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write at least five positive and five negative test cases for the login module of the travel booking application.
2. Generate test cases for the ‘Search and Filter’ functionality used for selecting travel dates, destination, and number of passengers.
3. Generate test summary report for the travel booking application

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

…………………………………………………………………………………………

……………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………

……………………………………………………………………………………………

XII References

1. https://testsigma.com/blog/test-plan/
2. https://www.geeksforgeeks.org/test-plan-software-testing/
3. https://www.pubnub.com/how-to/test-cases-for-chat-application/
4. https://www.softwaretestinghelp.com
5. http://spojtoolkit.com/TestCaseGenerator/

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	60 %
1	Application of ECP and BVA in test report design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	40 %
4	Quality and completeness of designed test report	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 16: *Design test plan and test cases for the login functionality of a social media application.

I Practical Significance

This practical focuses on Test Plan which is a document that describes a scope of testing, test strategy, objectives, effort, schedule and resources required. Its main purpose is to guide the whole testing process and used mostly by Project Managers or Tests Engineers. The scope of work is defined at the beginning of the testing process. A project team should clearly understand what features and functions to be tested and which ones are out of scope. This practical is useful for students to test any social media application by preparing its test cases and test plan.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO3 - Prepare test plan for a given application.

IV Laboratory Learning Outcome(s)

LLO 16.1 Write test plan and test cases for the login functionality of a social media application.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Testing the login functionality of a social media application is a fundamental aspect of software quality assurance, as it ensures secure and correct user authentication. A test plan provides a structured approach, defining the scope, objectives, resources, and schedule of testing activities, while **test cases** specify the inputs, expected outcomes, and execution steps to validate each scenario. For login functionality, test cases typically cover valid and invalid credentials, boundary conditions, account lockout mechanisms, password recovery, and session management. The theoretical principles of functional testing, security testing, and requirement validation guide the design of these test cases, ensuring that the system is robust, secure, and provides a reliable user experience. Proper documentation of test plans and cases also supports repeatable testing and facilitates maintenance and future.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Open the social media application for login functionality.
2. Determine the scope that need to be tested and that are NOT to be tested.
3. Prepare the document of Test Strategy.
4. Evaluate the test estimate.
5. Decide Entry and Exit criteria.
6. Plan when and how to test and decide how the test results will be evaluated and define test exit criterion.
7. Test artifacts delivered as part of test execution.
8. Define the management information including the metrics required and defect resolution and risk issues.
9. Ensure that the test documentation generates repeatable test assets.

IX Example

TEST PLAN FOR LOGIN FUNCTIONALITY

Version: 1.0

Test Plan ID: TP_10

1. **Introduction:**

   The plan verifies that the Social Media App's Login feature works correctly, securely, and handles invalid inputs properly.

2. **Test Items (Features):**
   - Valid login
   - Invalid login
   - Empty fields
   - Forgot password
   - Session timeout & logout
   - Remember Me option
3. **Approach:**
   - Functional & non-functional testing
   - Positive & negative tests based on requirements
   - Ad hoc (intuitive) testing

4. **Pass/Fail Criteria:**
    - All high-priority tests must pass
    - Minimum 99% requirements coverage
    - Testing stops on blocking defects; resumes once fixed
    - Deliverables: test plan, test cases, test report
5. **Test Tasks:**
    - Create test plan & test cases
    - Define success criteria
    - Execute tests & evaluate results
    - Prepare test reports
6. **Environment:**
    - **OS:** Windows 10, Android 13
    - **Browser/App:** Chrome v120, App v2.1
    - **Database:** MySQL
    - **Server:** Test server
    - **Tools:** Selenium, Postman
7. **Responsibilities:**
    - QA Lead: Review artifacts
    - Test Engineer: Execute tests & log defects
    - Developer: Fix defects
    - Project Manager: Approve release
8. **Staffing & Training:**
   Requires knowledge of login module, test design techniques, and functional/non-functional testing.
9. **Schedule:**
   Project deadline: **06/12/2019, 5:00 PM**
10. **Risks:**
    - Insufficient testing resources
    - Changing product requirements
11. **Approvals:**
   Team Lead, Test Engineer 1–4

Test Case ID	Test Case Objective	Input Data	Expected Result	Actual Result	Status
TC1	Login with valid credentials	Enter valid email & password → Click **Login**	User should log in successfully and redirect to dashboard	User logged in successfully and redirected to dashboard	Pass
TC2	Login with invalid password	Enter valid email & invalid password → Click **Login**	Error message **"Invalid credentials"** should appear	Error message **"Invalid credentials"** appeared	Pass

TC3	Login with empty fields	Leave both fields blank → Click **Login**	Validation message should appear for required fields	Validation message appeared for required fields	Pass
TC4	Forgot password functionality	Click **Forgot Password →** Enter email → Submit	Reset link should be sent to email	Reset link was sent but delayed	Fail
TC5	Session timeout	Log in → Stay idle for 20 minutes	Auto logout should occur with timeout message displayed	Auto logout occurred with message displayed	Pass

X Precautions

1. Note down the expected output for each test case carefully and generate summary report.
2. Select required test methodology and module(s).

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write five negative test cases for login with invalid credentials.
2. Define the entry and exit criteria for executing login-related test cases.
3. Prepare the test plan along with the test case for Amazon login module.

Space for Answer

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII References

1. https://testsigma.com/blog/test-plan/
2. https://www.geeksforgeeks.org/test-plan-software-testing/
3. https://www.pubnub.com/how-to/test-cases-for-chat-application/
4. https://www.softwaretestinghelp.com
5. http://spojtoolkit.com/TestCaseGenerator/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test plan design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed test plan	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 17: :*Generate Defect Report for Library Management System

I Practical Significance

This practical focuses on defect report which is a consolidated manuscript which describes a defect which was uncover by test engineer. Defect report is used to describe the problem bug which was discovered so that it can be easily identified by developers which can be resolve in later stage. It contains details of necessary steps that are to be followed when the software application which does not give expected output. Defect report is most effective way to communicate, track and explain defects to the stakeholders. In this practical a learner will be able to generate a defect report for Library Management System.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO4: Create defect report for a given application.

IV Laboratory Learning Outcome(s)

LLO 17.1: Prepare defect report after executing test cases for library management system.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Defects are loopholes in applications that are leftover accidentally while development of application. A case where actual result is differed from expected result while performing software test then it is known as defect. A test engineer needs to generate appropriate defect report so that it is easy for developer to understand exact problem with system and same can be resolved. While resolving defect one need to assign appropriate status for each defect. Usually there are several status of defect i.e. New, Assigned, Open, Resolved, Closed etc. Each defect has its severity based on which one can prioritize defects while solving them.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Select appropriate tool to generate defect report(Like any text editor/web based tool).
2. Identify required modules of the system which is to be tested and for whom defect report is to be generated.
3. Identify required parameters such as Severity, Priority and Status etc
4. Prepare and generate Defect report for defects which were uncovered while testing.
5. Verify defect report as per the specification by executing test data on required library management system.

IX Example

DEFECT REPORT

Project Name: Library Management System

Version: 1.0

Test Phase: System Testing

Date: 12-Nov-2025

Prepared By: QA Team

ID	R1
Project	Library Management System
Product	Web Application
Release Version	v1.0
Module	Login
Detected Build Version	V1.1
Summary	Login page does not validate incorrect password; user is allowed to log in with wrong credentials.

Description	When users enter an incorrect password, the system still allows them to access the dashboard. The expected behavior is that the login attempt should be rejected and an appropriate error message should be displayed. This issue compromises the security of the Library Management System and allows unauthorized access.
Steps to Replicate	1. Open the Library Management System login page. 2. Enter a valid username (e.g., **admin01**). 3. Enter an incorrect password (e.g., **wrong@123**). 4. Click on the **Login** button.
Actual Result	User is successfully logged in and redirected to the **Admin Dashboard** despite entering an incorrect password.
Expected Results	System should display an error message: **"Invalid username or password"** and prevent login.
Defect Severity	Critical
Defect Priority	High
Reported By	Test Engineer1
Assigned To	XYZ
Status	Open

X Precautions

1. Note down the expected output for each test case carefully.
2. Maintain the status of defect and manage same.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Generate online library management system with the help of following website. Perform at least 4 tests for same. Prepare defect report for library management system.
   - https://librarika.com/
2. Generate online library management system with the help of following website. Perform at least 3 tests for same. Prepare defect report for library management system.
   - https://www.libib.com/

Space for Answer

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

……………………………………………………………………………………………….

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

…………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII References

1. http://www.learnerswindow.com/library-management-sample-test-cases/
2. http://www.learnerswindow.com/library-management-sample-test-plan/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed defect report	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 18: : *Validate Defect Report for ATM Machine

I Practical Significance

This practical focuses on the defect report that will enable developer to build flawless system and provide best possible application. ATM is one of the most consumed systems in human day to day life. Large pool of people is active user of ATM machine on daily basis. ATM is one of the integral parts of banking system which facilitate use of banking application and make banking convenient for customer. Hence it becomes very much essential to validate functionality of ATM machine and prepare defect report for same. In this practical a learner will be able to generate a defect report for ATM Machine System.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO4: Create defect report for a given application.

IV Laboratory Learning Outcome(s)

LLO 18.1: Prepare defect report after executing test cases for withdrawn of amount from ATM Machine

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Generating a defect report is very much important in a scenario where developers and test engineers are working in different team/different place. This gives clear idea to the developer regarding the flaws that was generated during implementation phase. The test engineer is expected to be precise with defect report so that the development team can track problem easily. During the life span of defect it transits from various phases. Since test engineer generates the defect it becomes more important to track the status of same and update it at regular interval. The status of defect report will enable project manager to track the defect and take necessary measurements.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Select appropriate tool to generate defect report(Like any text editor/web based tool)
2. Identify required modules of the system which is to be tested and for whom defect report is to be generated.
3. Identify required parameters such as Severity, Priority, and Status etc.
4. Prepare and generate Defect report for defects which were uncovered while testing.
5. Verify defect report as per the specification by executing test data on required ATM machine.

IX Example

ID	R1
Project	Cash Simulator Cash (ATM)
Product	http://www.motc.gov.qa/en/ditoolkit/migrantworkers/cash-machine-simulator-atm
Release Version	v1.0
Module	Home Page> Our Programs > Digital Inclusion tools
Detected Build Version	V1.1
Summary	Limited denomination options in cash withdrawal function, restricting cash withdrawal only till 3000
Description	No option of withdrawing of amount excess of 3000.
Steps to Replicate	1) Open the website 2) Select our programs 3) Proceed to Digital Inclusion tools and select cash machine simulator (ATM) 4) Select language and skip to simulator 5) Enter the card 6) Select the account type 7) Go to Other functions and select cash withdrawal
Actual Result	It is displaying limited options of denominations in cash withdrawal option.

Expected Results	It should add more options in denominations in withdrawal function or it should take amount input from the user.
Attachments	
Remarks	Causes inconvenience to the user in terms of limited cash withdrawal options.
Defect Severity	High
Defect Priority	High
Reported By	Test Engineer1
Assigned To	XYZ
Status	Assigned

X Precautions

1. Note down the expected output for each test case carefully.
2. Maintain the status of defect and manage same.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Prepare a defect report for a scenario where the ATM dispenses incorrect cash amount.
2. Prepare a defect validation checklist for ATM software testing.
3. Write test cases to validate the resolution of a defect where the ATM displays “Transaction

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

………………………………………………………………………………………

XII References

1. https://www.geeksforgeeks.org/test-plan-software-testing/
2. https://www.pubnub.com/how-to/test-cases-for-chat-application/
3. https://www.softwaretestinghelp.com
4. http://spojtoolkit.com/TestCaseGenerator/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed defect report	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 19 : Execute Test Cases to Generate Defect Report for any login form

I Practical Significance

This practical focuses on login process indeed provides all types of authentication and authorization so that only legitimated user(s) can have access to the system. In view of providing security to the system one need to have rigorous verification of login module. This module eliminates basic threat which can come towards system from beginners or intermediate sources of attackers. The test engineer needs to verify all kind of testing such as interface testing, SQL Injection, DOS/DDOS etc. In this practical the student will be able to work with different interfaces to execute test cases and generate Defect report for login interface.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO4: Create defect report for a given application.

IV Laboratory Learning Outcome(s)

LLO 19.1 Prepare defect report after executing test cases for any login form.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Login module is directly connected with database of user where it is likely to have most of the confidential information about the user. While delivering an application one need to have a close eye on security. Providing an interface which will look after the authentication and segregating valid user from invalid user become very much important in such systems. The test engineer is expected to design a test case in such a way system is safe in all dimensions and hence while designing test case more emphasis should be given on security testing. The defect report shall be covering all possible ways that system is likely to be attacked and hence finding vulnerabilities within system becomes vital. Once the test engineer can identifies these weaknesses of a system then the developer can work out these highlighted areas to achieve maximum security.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Select appropriate tool to generate defect report (Like any text editor/web based tool)
2. Choose login modules of the system which is to be tested and for whom defect report is to be generated.
3. Identify required parameters such as Severity, Priority, and Status etc.
4. Prepare and generate Defect report for defects which were uncovered while testing.
5. Verify defect report as per the specification by executing test data on required system.

IX Example

Test case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Check cursor position at email or mobile number field	Click on email or mobile number field	Cursor should be placed on the field	Placed the cursor on the field	Pass
TC2	Check cursor position at password field	Click on password field	Cursor should be placed on the password field	Placed the cursor on the password field	Pass
TC3	Check the continue button	Click on continue button	It should redirect to password page	It redirected to the password page.	Pass
TC4	Readability of font	Try to read the contents on login page	Contents should be readable	Contents are readable	Pass
TC5	Testing of spelling of login	Check the spelling of login	Login spelling should be correct	Spelling of Login is correct	Pass

TC6	Testing of hyperlink	Hover the mouse on hyperlink	It should change the cursor and should redirec t to respect ive page on click	Cursor changed and redirects to other page.	Pass

ID	DR-LOGIN-021
Project	Travel Booking Application
Product	Web Portal
Release Version	v1.0
Module	User Login Module
Detected Build Version	V1.1
Summary	Password page does not load when mobile number is entered with country code.
Description	While executing login test scenarios, it was observed that if the user enters a mobile number with the **country code prefix (+91)** and clicks the **Continue** button, the application does not navigate to the password page.The screen stays on the same page with no error message displayed.This defect was not detected in earlier test cases because they used normal (10-digit) numbers.
Steps to Replicate	1. Open the Login page. 2. Click inside the Email/Mobile Number textbox. 3. Enter mobile number in the format: +91 9876543210. 4. Click on the Continue button.
Actual Result	The page does not redirect to the password page; it remains on the same login input screen without any validation message.
Expected Results	The application should accept mobile numbers with country code and redirect the user to the **Password** page. OR It should display a proper validation message like: **“Country code not allowed. Please enter a 10-digit mobile number.”**
Defect Severity	Medium (User is blocked only for this specific input pattern)
Defect Priority	High (Affects real users entering mobile numbers with country code)
Reported By	Test Engineer1
Assigned To	XYZ
Status	Open

X Precautions

1. Mark status for each defect reported.
2. Select required test methodology and module(s).

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Login system simulator with the help of following website. Perform at given task and generate test case. Prepare defect report for Login system.

   Task

   a. Enter invalid user name.
   b. Enter invalid password.
   c. Enter password with only 3 characters.
   d. Enter user name as "invitado" and password as "hgm2015".
   e. https://codepen.io/opensoorce/pen/KQmvdL

2. Consider any web base system which provides login procedure Perform following tests for same. Prepare defect report for Login System.

   Task

   a. Verify forgot password link.
   b. Test user name as "STEPR" and password as "STEPR",
   c. Verify captcha for given system

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII References

1. https://codepen.io/opensoorce/pen/KQmvdL
2. https://www.softwaretestinghelp.com/test-cases-for-login/

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	40 %
4	Quality and completeness of designed defect report	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 20: Defect Report for Hostel Admission Form.

I Practical Significance

This practical focuses on generating a defect report for a Hostel Admission Form lies in ensuring the accuracy, reliability, and usability of the admission process. By identifying and documenting defects such as incorrect field validations, submission errors or workflow issues, testers help prevent errors that could affect student records or administrative efficiency. The defect report provides a clear and structured record of all issues, enabling developers to prioritize and resolve them promptly. This process ensures that the hostel admission system functions smoothly, maintains data integrity, and delivers a secure and user-friendly experience for both students and administrative staff.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO4: Create defect report for a given application.

IV Laboratory Learning Outcome(s)

LLO 20.1: Prepare defect report after executing test cases for hostel admission form..

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Validating and documenting a defect report for a Hostel Admission Form is an essential aspect of software testing and quality assurance. The process is grounded in functional testing, which ensures that all form fields, validations, and workflows perform according to the specified requirements, and negative testing, which evaluates the system’s behavior under invalid or unexpected inputs. Each defect identified during testing—such as incorrect field validation, data submission errors, or workflow inconsistencies—is recorded in a defect report, providing structured information for developers to analyze, prioritize, and resolve the issues. This systematic approach ensures that the admission form functions accurately, data integrity is maintained, and the system is reliable, secure, and user-friendly for students and administrators alike.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Review the functional requirements of the Hostel Admission Form.
2. Identify key fields such as Name, Roll Number, Department, Year, Room Type, and Contact Details.
3. Design and execute test cases for all functionalities like form validation, data submission, and error handling.
4. Compare the actual output of each test case with the expected result.
5. Document each defect with a unique Defect ID and Module Name (Hostel Admission Form).
6. Update the defect status as Fixed, Closed, or Reopened based on the re-test result.
7. Compile all logged defects into a Defect Report Summary showing total, open, closed, and pending defects.
8. Review the final defect report.

IX Example

ID	R1
Project	Hostel Management System
Product	Hostel Admission Portal
Release Version	v1.0
Module	Hostel Admission Form
Detected Build Version	V1.1
Summary	Hostel Admission Form allows submission with missing mandatory fields.
Description	When a user submits the Hostel Admission Form without entering mandatory fields like *Student Name*, *Date of Birth*, *Parent Contact Number*, the system still accepts the form and displays a success message. Mandatory field validation is not triggered.
Steps to Replicate	1. Navigate to the Hostel Admission Portal. 2. Go to Admission → Hostel Admission Form.

	3. Leave mandatory fields blank (e.g., Student Name, DOB, Parent Mobile Number). 4. Click on the Submit button.
Actual Result	Form gets submitted successfully without any validation messages.
Expected Results	System must display validation errors for all mandatory fields and prevent form submission until required data is entered.
Defect Severity	High (may lead to invalid or incomplete student records)
Defect Priority	Medium (needs fix before next release cycle)
Reported By	Test Engineer1
Assigned To	XYZ
Status	Assigned

X Precautions

1. Mark status for each defect reported.
2. Select required test methodology and module(s).

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Design test cases for hostel admission form of your institute.
2. If the “Name” field accepts numbers, what kind of defect is that?
3. If the “Email” field accepts data without ‘@’ symbol, what type of defect is it?
4. If the “Mobile Number” field accepts less than 10 digits, what defect would you log?

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………….

XII References

1. https://codepen.io/opensoorce/pen/KQmvdL
2. https://www.softwaretestinghelp.com/test-cases-for-hostel/

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	40 %
4	Quality and completeness of designed defect report	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 21 : *Installation and Configuration of Selenium IDE.

I Practical Significance

This practical focuses on installing and configuring Selenium IDE lies in its role as a foundational tool for automated web application testing. Selenium IDE provides a user-friendly interface to record, edit, and replay test scripts, enabling testers to efficiently validate web application functionality without extensive programming knowledge. Proper installation and configuration ensure that the tool integrates seamlessly with browsers, supports execution of test cases, and allows for easy export of scripts to other Selenium frameworks. This process helps testers save time, reduce manual errors, and maintain consistent and repeatable test execution, ultimately improving the reliability, efficiency, and quality of web applications.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO5: Apply automation testing tools to test software.

IV Laboratory Learning Outcome(s)

LLO 21.1: Install and configure Selenium IDE to apply automation testing concepts.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Selenium IDE (Integrated Development Environment) is a widely used open-source tool for automating web application testing. It allows testers to record, edit, and replay test scripts directly in a browser, supporting rapid test development without extensive programming knowledge. The installation and configuration of Selenium IDE are essential to ensure proper integration with web browsers and smooth execution of test cases. Theoretically, Selenium IDE is based on the principles of automated functional testing, which helps in validating web application workflows, user interactions, and UI elements efficiently. It also supports exporting test scripts to other Selenium frameworks (like Selenium WebDriver), enabling scalability and maintainability of test automation projects. Proper setup ensures consistent, repeatable, and reliable test execution, forming the foundation for effective automated testing strategies

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Open a web browser such as Google Chrome or Mozilla Firefox.
2. Go to the official Selenium IDE website or browser extension store.
3. In the search bar, type “Selenium IDE”.
4. Select the official Selenium IDE extension developed by SeleniumHQ.
5. Click on the “Add to Chrome” or “Add to Firefox” button.
6. Confirm the installation by clicking “Add Extension” when prompted.
7. Wait for the installation to complete; the Selenium IDE icon will appear on the browser toolbar.
8. Click the Selenium IDE icon to launch the tool.
9. Name your project and click OK to continue.
10. Click on the Run button to execute the test case and observe the results
11. Review the execution log to verify pass or fail status of each command.

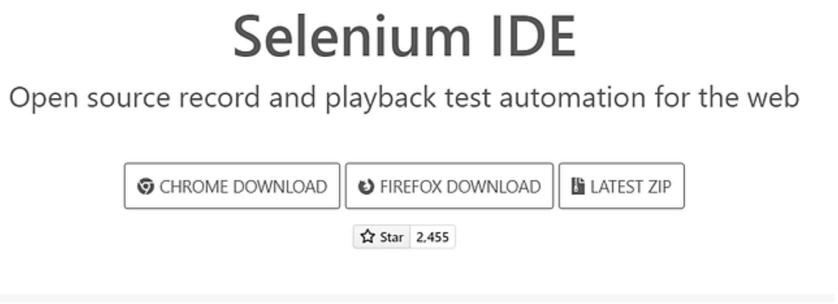
IX Example

Installation of Selenium IDE in Mozilla Firefox

To install Selenium IDE on Mozilla Firefox you need to follow the given steps:

Step 1: Open the Mozilla Firefox browser on your system and then go to the Link.

Step 2: As you click on the link you will be redirected to the Selenium IDE website where you need to choose the Firefox download button to download the extension of Selenium IDE on the Firefox browser.

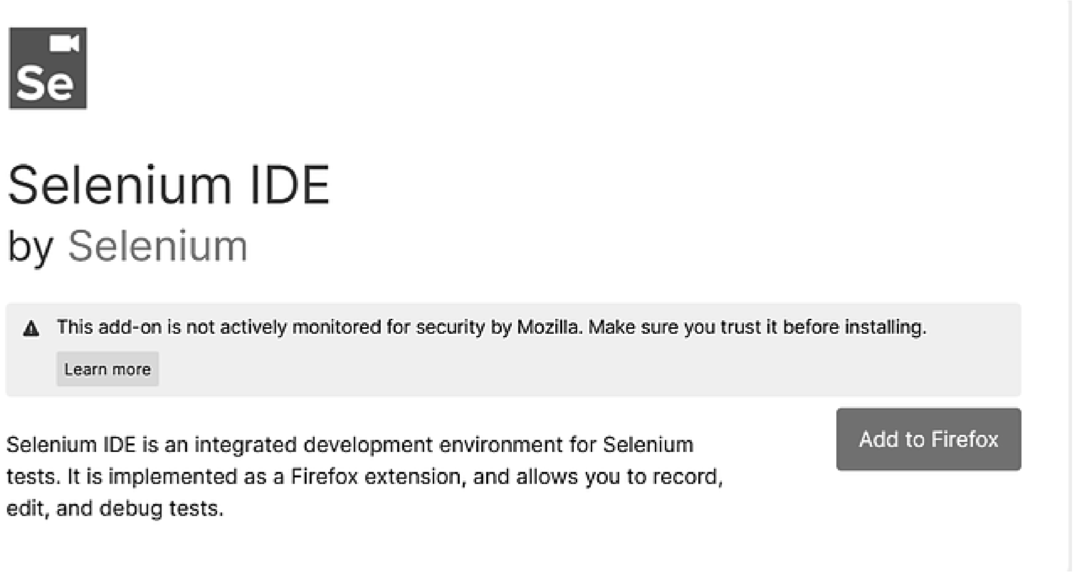


Selenium IDE

Open source record and playback test automation for the web

Selenium IDE Website

Step 3: After clicking on the Firefox download button you need to click on the Add to Firefox button to add the extension of Selenium IDE on your browser. Grant all the permissions and click on the option.



Selenium IDE

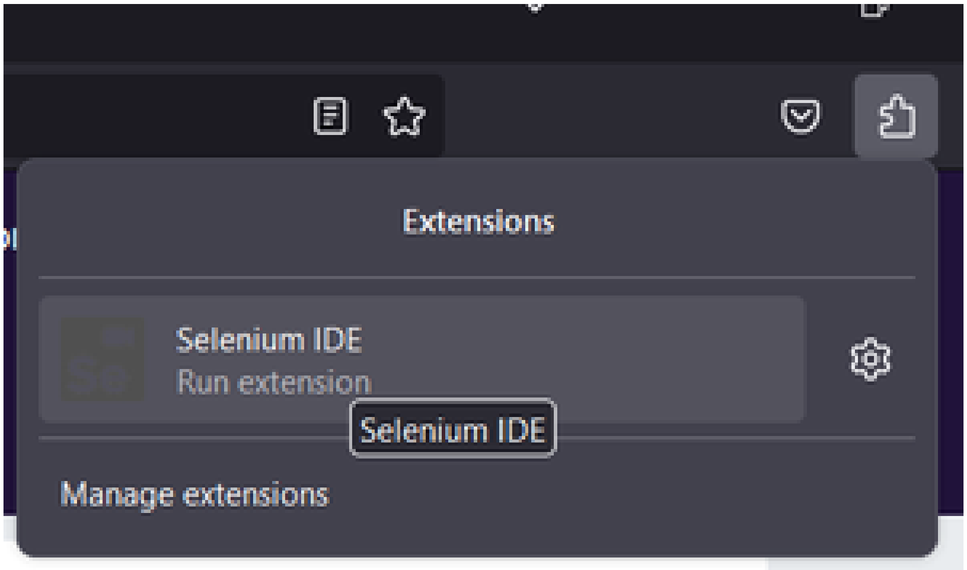
by Selenium

This add-on is not actively monitored for security by Mozilla. Make sure you trust it before installing.

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests.

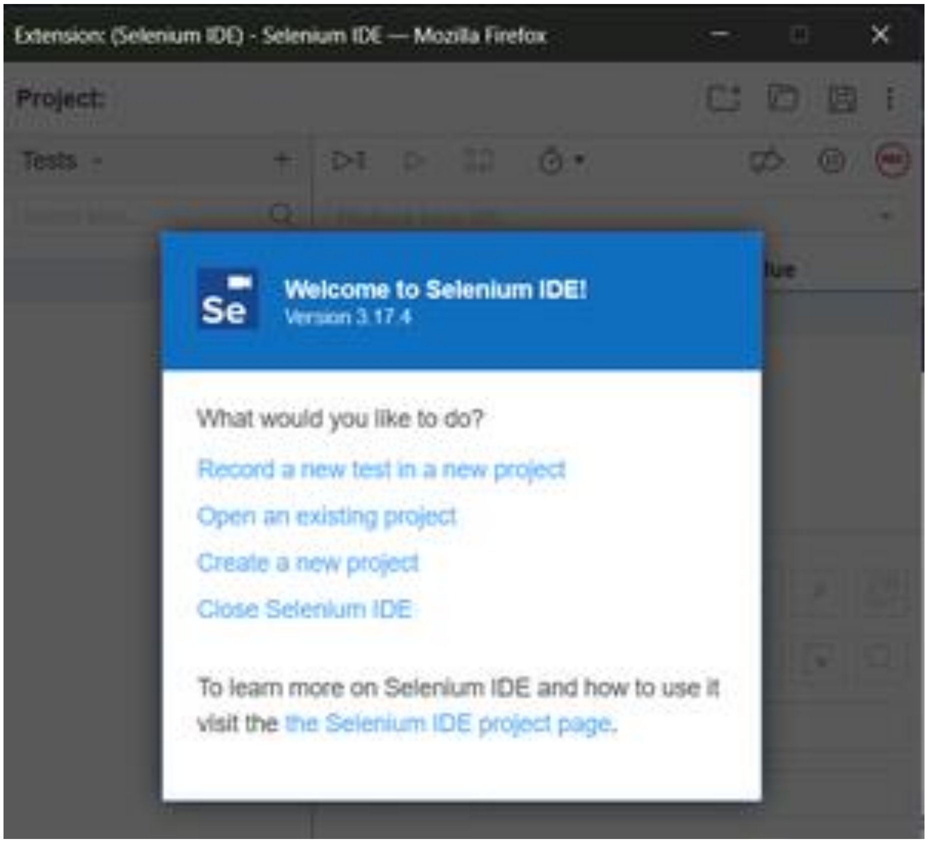
Add to Firefox button

Step 4: Now you need to go to the extension icon which is placed at the top right corner on your browser's screen. A list of extensions will appear on the screen. Choose the Selenium IDE extension.



Selenium IDE extension.

Step 5: Now click on Selenium IDE to start working with Selenium IDE. You can now record a new test in a new project, open an existing project, and also you can create a new project.



Welcome to Selenium IDE!
Version 3.17.4

What would you like to do?

Record a new test in a new project

Open an existing project

Create a new project

Close Selenium IDE

To learn more on Selenium IDE and how to use it visit the the Selenium IDE project page.

Selenium IDE successfully Installed

Selenium IDE has now been successfully added to your Mozilla Firefox web browser.

X Precautions

1. Ensure the Selenium IDE extension is installed from the official browser store to avoid compatibility or security issues.
2. Verify browser settings and permissions

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. How do you configure Selenium IDE for a specific base URL?
2. How do you record a test case in Selenium IDE?
3. Enlist factors considered for selecting a testing tool for test automation.

Space for Answer

………………………………………………………………………………………………………

………………………………………………………………………………………………………

………………………………………………………………………………………………………

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII References

1. https://searchsoftwarequality.techtarget.com/definition/automated-software-testing.
2. https://www.softwaretestinghelp.com/automation-testing-tutorial-1/

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	40 %
4	Quality and completeness of installation	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 22 : *Test Case Design and Execution for Notepad (Windows-Based) Using Selenium IDE

I Practical Significance

This practical focuses on designing and executing test cases for Notepad using Selenium IDE is practically significant as it helps validate the core functionalities of the application, such as creating, editing, saving, and opening text files, in an automated and systematic manner. By automating these test cases, testers can quickly verify multiple scenarios without manual effort, reducing human error and saving time. The process ensures that Notepad performs reliably under different input conditions, maintains data integrity, and responds correctly to user actions, which enhances confidence in the application's stability and usability. Additionally, documenting the test execution results provides clear evidence of system behavior and supports future regression testing.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO5: Apply automation testing tools to test software.

IV Laboratory Learning Outcome(s)

LLO 22.1 Write and run test cases for Notepad using Selenium IDE.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Selenium IDE is an open-source tool primarily designed for automating web-based applications, but it can also be configured to simulate user interactions with desktop-based applications using certain extensions or auxiliary frameworks. Test case design involves creating structured steps that define input actions, expected outcomes, and validation points for Notepad functionalities, based on principles of functional testing and automation testing. Execution of these test cases helps in systematically identifying defects, verifying application behavior, and ensuring correctness of workflows. The theoretical foundation emphasizes automated functional testing, reproducibility of tests, and efficient defect detection, enabling testers to ensure that Notepad performs consistently and meets expected operational requirements.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.), Notepad		

VIII Procedure

1. The scope of Automation needs to be determined in detail before the start of the project.
2. Launch Selenium IDE
3. Choose an appropriate framework.
4. Open Notepad Window based application.
5. Standards have to be followed while writing/recording the scripts for Automation.
6. Create Record uniform scripts, comments, and indentation of the code.
7. User-defined messages should be coded or standardized for Error Logging for testers to understand. (If applicable)
8. Measure metrics- Success of automation capture by using the following metrics parameters such as-
    - Percent of defects found
    - The time required for automation testing for each and every release cycle
    - Minimal Time is taken for release
    - Customer Satisfaction Index
    - Productivity improvement

IX Example

Test case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Test the select all option	Click on select all	All the text should be selected	All the text is selected	Pass
TC2	Cut option	Select the text and click on cut	Selected text should be cut	Selected text is cut	Pass
TC3	Paste option	Click on paste	Contents should be pasted	Contents are pasted	Pass
TC4	Delete option	Select text and click on delete	Contents should be deleted	Contents are deleted	Pass

```
import io.appium.java_client.windows.WindowsDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.By;
import java.net.URL;
import java.nio.file.*;
public class NotepadTest {
    public static void main(String[] args) throws Exception {
        // Start WinAppDriver before running this program
        DesiredCapabilities caps = new DesiredCapabilities();
        caps.setCapability("app", "notepad.exe");
        WindowsDriver driver = new WindowsDriver(new URL("http://127.0.0.1:4723"), caps);
        // Step 1: Type into Notepad
        driver.findElement(By.className("Edit")).sendKeys("Hello World");
        // Step 2: Open File -> Save As
        driver.findElement(By.name("File")).click();
        driver.findElement(By.name("Save As...")).click();
        // Step 3: Enter file name
        Thread.sleep(1000);
        driver.findElement(By.className("Edit")).sendKeys("testfile.txt");
        // Step 4: Click Save
        driver.findElement(By.name("Save")).click();
        // Step 5: Verify file exists
        String path = System.getProperty("user.home") + "\\Documents\\testfile.txt";
        if (Files.exists(Paths.get(path))) {
            System.out.println("Test Passed: File saved successfully.");
        } else {
            System.out.println("Test Failed: File not found.");
        }
        // Close Notepad
        driver.quit();
    }
}
```

X Precautions

1. Note down the expected output for each test case carefully.
2. Select required test methodology and module(s).

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Test the "View" option available in Notepad application. Attached the recorded script.
2. Test the procedure of opening and closing Notepad application using the selenium IDE and attached screen shots.
3. Test the basic "login" functionality for any application using selenium IDE .Write down the related test cases or attached the screen shots.

Space for Answer

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XII References

1. https://searchsoftwarequality.techtarget.com/definition/automated-software-testing.
2. https://www.softwaretestinghelp.com/automation-testing-tutorial-1/

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	40 %
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 23 : Test Case Design and Execution for MS Word application using Selenium IDE.

I Practical Significance

This practical focuses on designing and executing test cases for Notepad using Selenium IDE is practically significant as it helps validate the core functionalities of the application, such as creating, editing, saving, and opening text files, in an automated and systematic manner. By automating these test cases, testers can quickly verify multiple scenarios without manual effort, reducing human error and saving time. The process ensures that Notepad performs reliably under different input conditions, maintains data integrity, and responds correctly to user actions, which enhances confidence in the application's stability and usability. Additionally, documenting the test execution results provides clear evidence of system behavior and supports future regression testing.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO5: Apply automation testing tools to test software.

IV Laboratory Learning Outcome(s)

LLO 23.1 Write and run test cases for MS Word application using Selenium IDE.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

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.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.) Selenium IDE		

VIII Procedure

1. Identify areas within software to automate.
2. Launch Selenium IDE
3. Write test scripts.
4. Develop test suits.
5. Execute test scripts.
6. Build result reports.
7. Find possible bugs or performance issue.

IX Precautions

1. Note down the expected output for each test case carefully.
2. Select required test methodology and module(s).

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write test cases for copy & paste in MS Word using Selenium IDE.
2. Design possible Test cases for Save option in M.S word. Attached the sample test cases.
3. Test the "Insert Table" option available in the MS Word. Write the suitable test cases. Run the Test cases using Selenium IDE.

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………………

…………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XI References

1. https://searchsoftwarequality.techtarget.com/definition/automated-software-testing.
2. https://www.softwaretestinghelp.com/test-case-template-examples/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 24 : *Installation and Configuration of Selenium WebDriver

I Practical Significance

The practical focuses on significance of installing and configuring Selenium WebDriver lies in its role as a core tool for automating web application testing across multiple browsers. Proper installation and setup enable testers to write and execute automated scripts in programming languages such as Java, Python, or C#, allowing them to simulate real user interactions like clicking buttons, entering data, and navigating web pages. This process reduces manual testing effort, improves test accuracy, ensures consistency, and accelerates regression testing. Additionally, a correctly configured WebDriver environment ensures compatibility with various browsers and operating systems, providing reliable and efficient automated testing for web applications.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO5: Apply automation testing tools to test software.

IV Laboratory Learning Outcome(s)

LLO 24.1 Install and configure Selenium WebDriver to apply automation testing concepts.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Selenium WebDriver is a widely used open-source tool for automated browser testing, designed to interact directly with web browsers at the OS level, providing faster and more robust test execution than Selenium IDE. The installation and configuration process ensures that the WebDriver is correctly linked with the chosen programming language bindings and browser drivers, enabling seamless automation of web application workflows. The theoretical foundation is based on automated functional testing, object-based interaction with web elements, and cross-browser compatibility testing, which allows testers to systematically validate user scenarios, detect defects efficiently, and ensure the reliability and correctness of web applications. Proper setup is essential for leveraging the full potential of Selenium WebDriver in scalable and maintainable automation frameworks.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

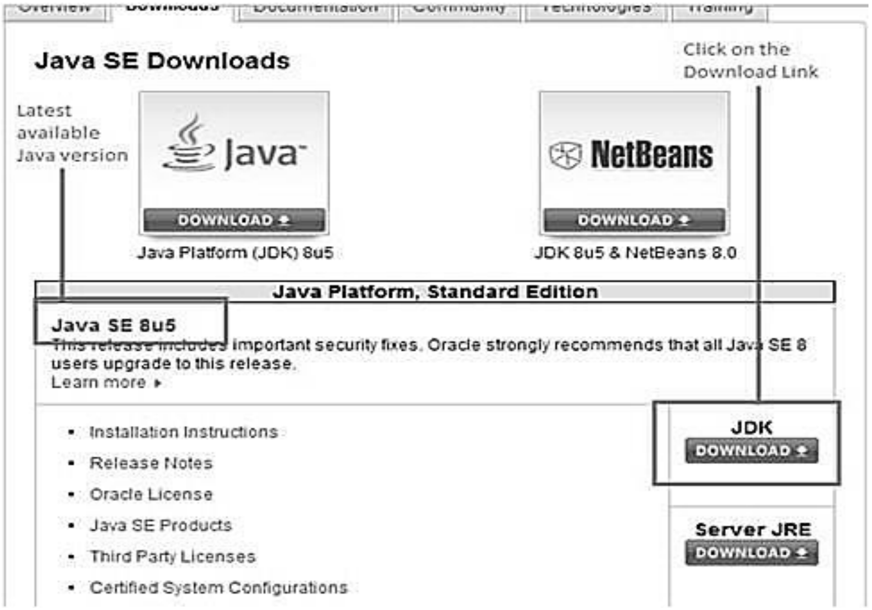
1. Open a Web Browser
2. Download and Install Java (JDK)
3. Install an IDE (Eclipse or IntelliJ IDEA)
4. Download Selenium WebDriver Libraries
5. Download the Browser Driver
6. Configure WebDriver Path
7. Create a New Selenium Project in IDE
8. Write and Run a Sample Test Script

IX Example

Setting up and Installing WebDriver in Eclipse

Java Installation

Step #1: Go to Oracle's official site – "JAVA download", and download Java Platform, Standard Edition. All the recent releases are available on the page.



Java SE Downloads

Java Platform (JDK) 8u5

JDK 8u5 & NetBeans 8.0

Java Platform, Standard Edition

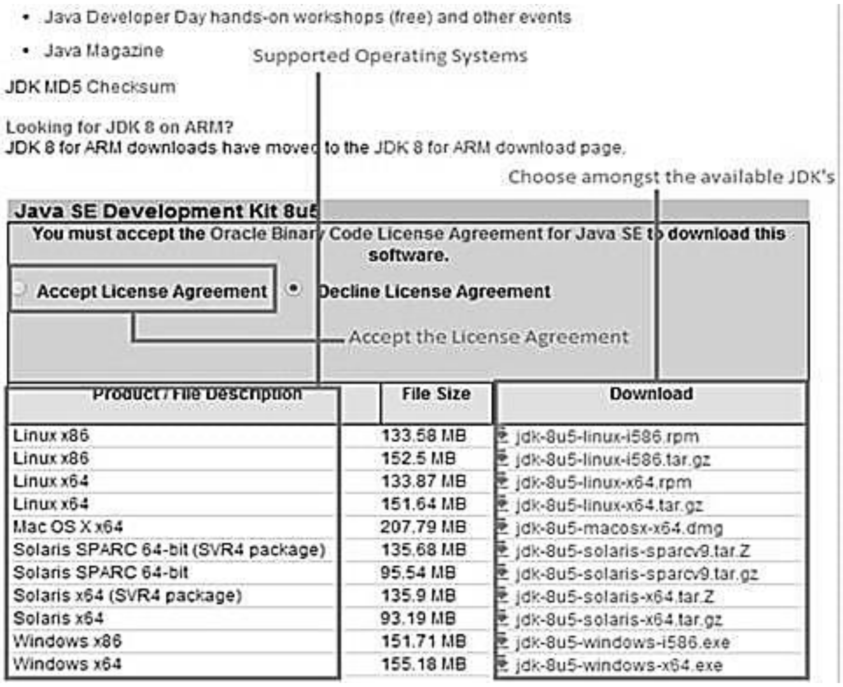
Java SE 8u5

This release includes important security fixes. Oracle strongly recommends that all Java SE 8 users upgrade to this release.

Learn more ▸

- Installation Instructions
- Release Notes
- Oracle License
- Java SE Products
- Third Party Licenses
- Certified System Configurations

Step #2: As soon as you click on the Download button, the following screen will appear. Accept the License Agreement for Java installation and choose amongst the various cataloged Java Development Kit's. Select the one that best suits your system configuration.


Supported Operating Systems
Choose amongst the available JDK's
Accept the License Agreement

- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

JDK MD5 Checksum

Looking for JDK 8 on ARM?
JDK 8 for ARM downloads have moved to the JDK 8 for ARM download page.

Java SE Development Kit 8u5

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

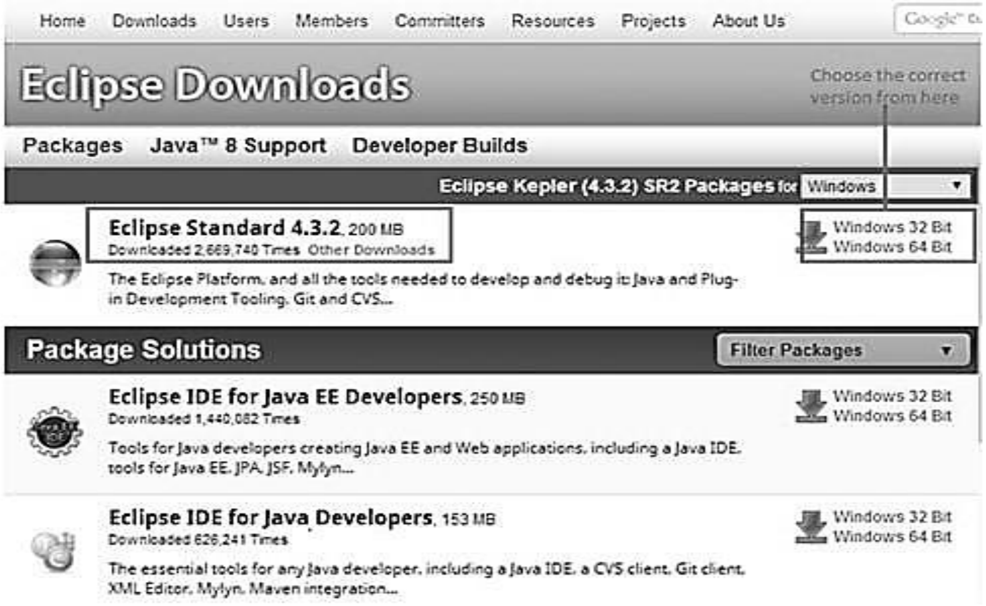
○ Accept License Agreement ● Decline License Agreement

Product / File Description	File Size	Download
Linux x86	133.58 MB	jdk-8u5-linux-i586.rpm
Linux x86	152.5 MB	jdk-8u5-linux-i586.tar.gz
Linux x64	133.87 MB	jdk-8u5-linux-x64.rpm
Linux x64	151.64 MB	jdk-8u5-linux-x64.tar.gz
Mac OS X x64	207.79 MB	jdk-8u5-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	135.68 MB	jdk-8u5-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	95.54 MB	jdk-8u5-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	135.9 MB	jdk-8u5-solaris-x64.tar.Z
Solaris x64	93.19 MB	jdk-8u5-solaris-x64.tar.gz
Windows x86	151.71 MB	jdk-8u5-windows-i586.exe
Windows x64	155.18 MB	jdk-8u5-windows-x64.exe

Remember to download JDK (Java development kit). The kit comes with a JRE (Java Runtime Environment). Thus, the user isn't required to download and install the JRE separately.

Eclipse IDE Installation

Step #1: Go to Eclipse's official website and navigate to its download page – Eclipse download. Download Eclipse IDE for Java EE developers. All the recent releases are available on the page. Make sure you opt for and download the appropriate Eclipse IDE as per your system configuration. There are two download links available for the 64-bit Windows operating system and 32-bit Windows operating system.



Eclipse Downloads

Eclipse Kepler (4.3.2) SR2 Packages for Windows

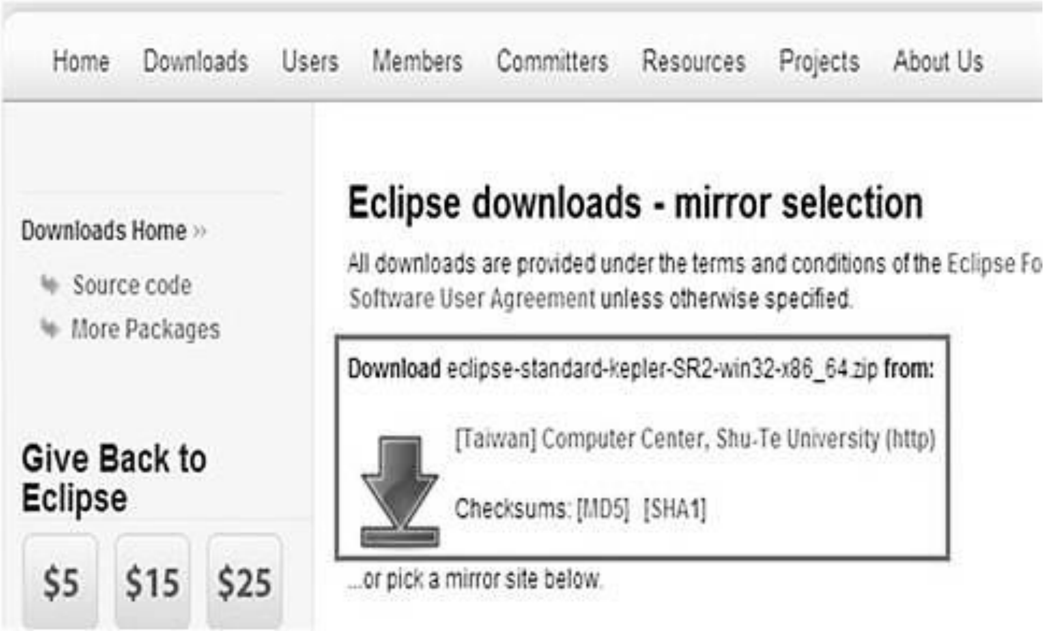
Eclipse Standard 4.3.2, 200 MB
Downloaded 2,669,740 Times Other Downloads
The Eclipse Platform, and all the tools needed to develop and debug it: Java and Plug-in Development Tooling, Git and CVS...
Windows 32 Bit
Windows 64 Bit

Package Solutions

Eclipse IDE for Java EE Developers, 250 MB
Downloaded 1,440,082 Times
Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn...
Windows 32 Bit
Windows 64 Bit

Eclipse IDE for Java Developers, 153 MB
Downloaded 626,241 Times
The essential tools for any Java developer, including a Java IDE, a CVS client, Git client, XML Editor, Mylyn, Maven integration...
Windows 32 Bit
Windows 64 Bit

Step #2: As soon as we click on the download link, the user is redirected to the fresh page securing information about the current download. Click on the download icon and you are done.



Eclipse downloads - mirror selection

All downloads are provided under the terms and conditions of the Eclipse Fo Software User Agreement unless otherwise specified.

Download eclipse-standard-kepler-SR2-win32-x86_64.zip from:

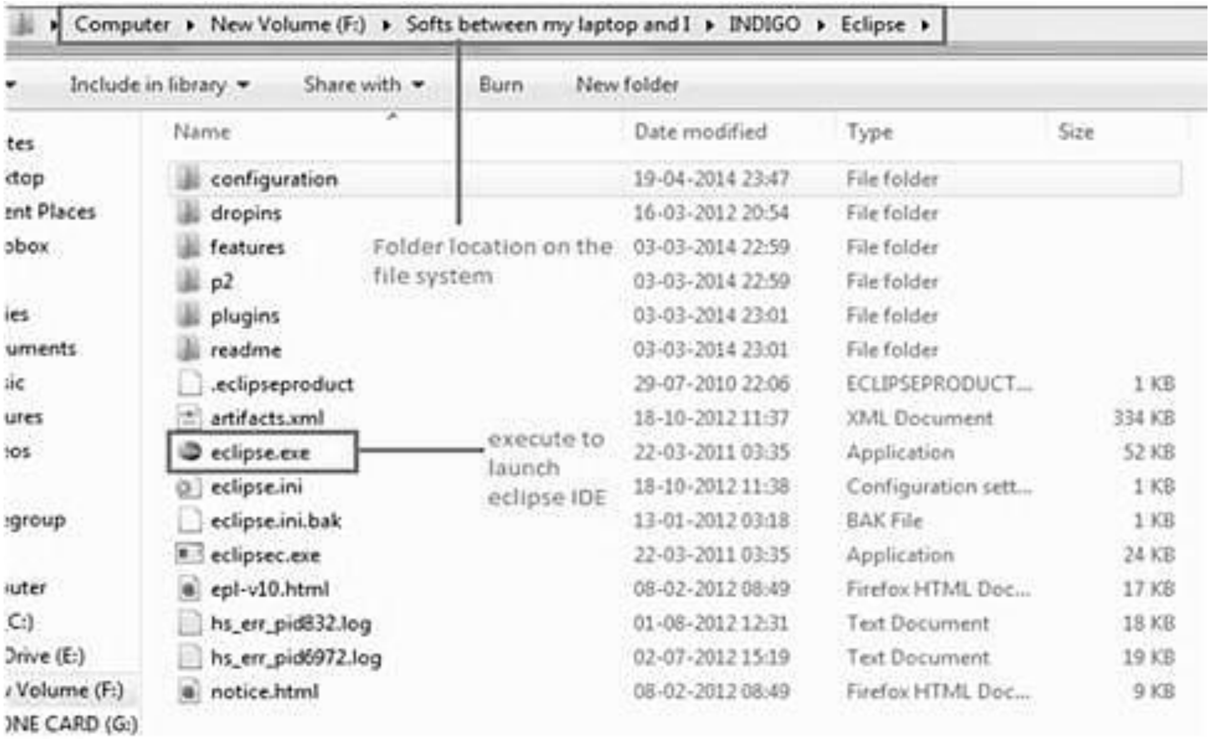
[Taiwan] Computer Center, Shu-Te University (http)

Checksums: [MD5] [SHA1]

...or pick a mirror site below.

It may take a few minutes before you can download the complete zip folder.

Step #3: Once downloaded, copy the folder and place it in the desired location on your file system.



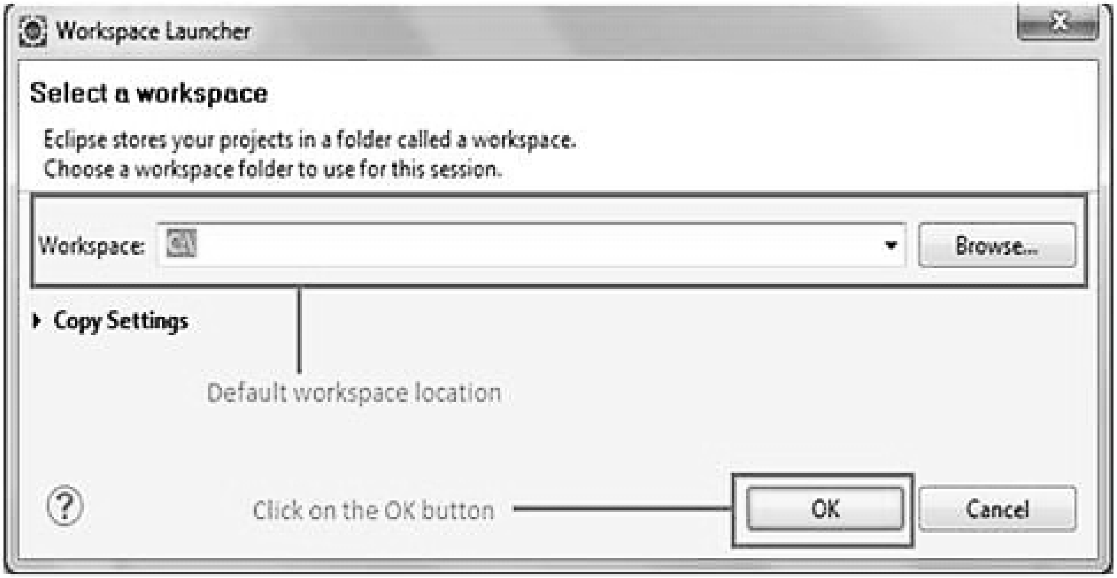
Name	Date modified	Type	Size
configuration	19-04-2014 23:47	File folder	
dropins	16-03-2012 20:54	File folder	
features	03-03-2014 22:59	File folder	
p2	03-03-2014 22:59	File folder	
plugins	03-03-2014 23:01	File folder	
readme	03-03-2014 23:01	File folder	
.eclipseproduct	29-07-2010 22:06	ECLIPSEPRODUCT...	1 KB
artifacts.xml	18-10-2012 11:37	XML Document	334 KB
eclipse.exe	22-03-2011 03:35	Application	52 KB
eclipse.ini	18-10-2012 11:38	Configuration sett...	1 KB
eclipse.ini.bak	13-01-2012 03:18	BAK File	1 KB
eclipsec.exe	22-03-2011 03:35	Application	24 KB
epl-v10.html	08-02-2012 08:49	Firefox HTML Doc...	17 KB
hs_err_pid832.log	01-08-2012 12:31	Text Document	18 KB
hs_err_pid6972.log	02-07-2012 15:19	Text Document	19 KB
notice.html	08-02-2012 08:49	Firefox HTML Doc...	9 KB

Folder location on the file system

execute to launch eclipse IDE

Step #5: Launch the Eclipse IDE using “eclipse.exe” residing inside the eclipse folder. Refer to the above illustration for the same.

Step #6: The application will prompt you to specify the workspace location. Workspace is the location where all your eclipse projects will be residing. Enter/Browse the desired location or the user can simply opt for the default location and click on the OK button.



Workspace Launcher

Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

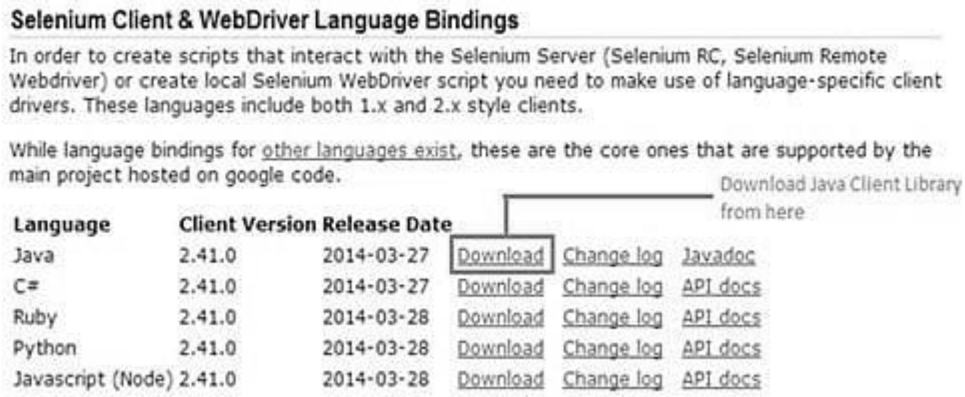
Workspace: C:\

Configuring WebDriver

As we will use Java as the programming language for this series and to create test scripts in Java, we would have to introduce language-specific client drivers. Thus, let us begin with the downloading of Selenium Java Client Libraries.

Download the Selenium Java Client Libraries

Step #1: Go to Selenium's official website and navigate to its download page – "http://docs.seleniumhq.org/download/". Refer to the section in the below illustration where you can find Client Libraries listed for distinct programming languages. Click on the download link for Java Client Library.



Selenium Client & WebDriver Language Bindings

In order to create scripts that interact with the Selenium Server (Selenium RC, Selenium Remote Webdriver) or create local Selenium WebDriver script you need to make use of language-specific client drivers. These languages include both 1.x and 2.x style clients.

While language bindings for other languages exist, these are the core ones that are supported by the main project hosted on google code.

Language	Client Version	Release Date			
Java	2.41.0	2014-03-27	Download	Change log	Javadoc
C#	2.41.0	2014-03-27	Download	Change log	API docs
Ruby	2.41.0	2014-03-28	Download	Change log	API docs
Python	2.41.0	2014-03-28	Download	Change log	API docs
Javascript (Node)	2.41.0	2014-03-28	Download	Change log	API docs

It may take a few minutes before you can download the complete zipped folder.

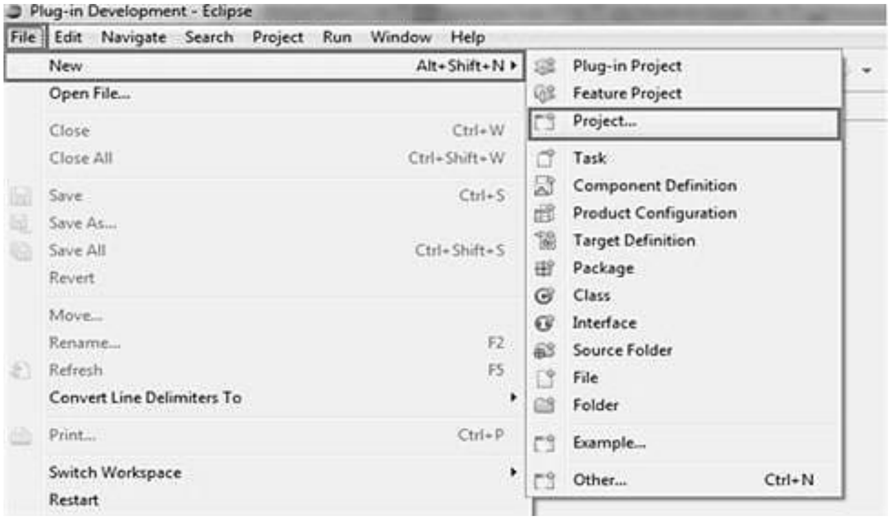
Step #2: Once downloaded, copy the folder and place it in the desired location on your file system.

Step #3: Extract the zipped folder, a folder named "Selenium-2.41.0.zip" can be seen. The folder embodies all the required jar files which enable users to create test scripts in Java.

Thus, these libraries can be configured in Eclipse IDE.

Configuring Libraries with Eclipse IDE

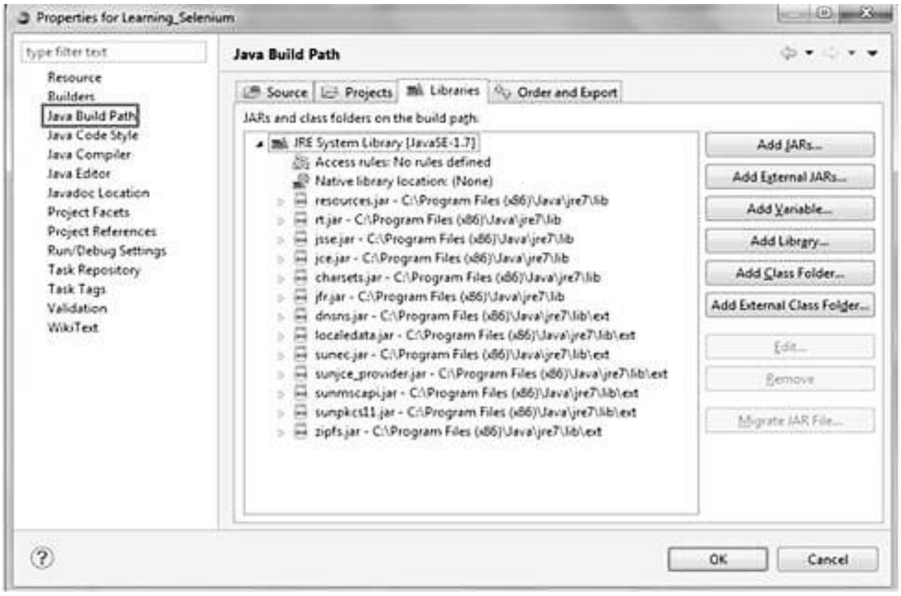
Step #1: Navigate towards Eclipse IDE. Create a new Java-based project following File -> New -> Java Project. Refer to the following figure for the same.



Step #2: Provide a user-defined name for your Java Project. Let us provide the name as Learning_Selenium and Click on the Finish Button. The package explorer panel on the left side of the screen displays the newly created project.

Step #3: Create a new Java class named as "First_WebdriverClass" under the source folder by right-clicking on it and navigating to New -> class.

Step #4: Now let us configure the libraries into our Java project. For this, select the project and Right click on it. Select "Properties" within the listed options. The following screen appears. Select "Java Build Path" from the options.

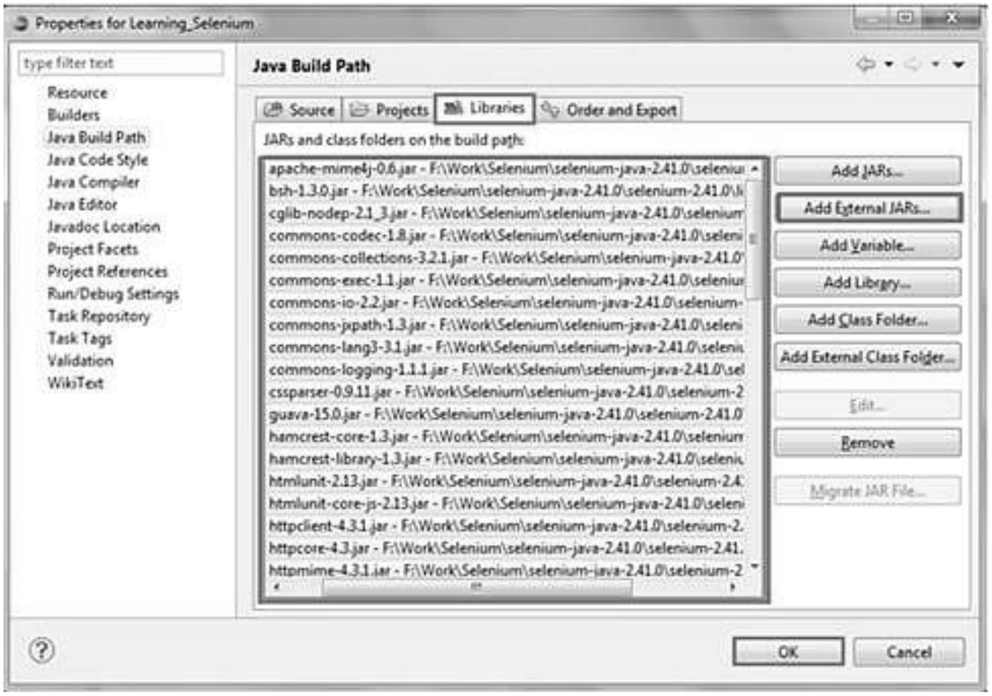


JARs and class folders on the build path:

```
JRE System Library [JavaSE-1.7]
    Access rules: No rules defined
    Native library location: (None)
    resources.jar - C:\Program Files (x86)\Java\jre7\lib
    rt.jar - C:\Program Files (x86)\Java\jre7\lib
    jsse.jar - C:\Program Files (x86)\Java\jre7\lib
    jce.jar - C:\Program Files (x86)\Java\jre7\lib
    charsets.jar - C:\Program Files (x86)\Java\jre7\lib
    jfr.jar - C:\Program Files (x86)\Java\jre7\lib
    dnsns.jar - C:\Program Files (x86)\Java\jre7\lib\ext
    localedata.jar - C:\Program Files (x86)\Java\jre7\lib\ext
    sunec.jar - C:\Program Files (x86)\Java\jre7\lib\ext
    sunjce_provider.jar - C:\Program Files (x86)\Java\jre7\lib\ext
    sunmscapi.jar - C:\Program Files (x86)\Java\jre7\lib\ext
    sunpkcs11.jar - C:\Program Files (x86)\Java\jre7\lib\ext
    zipfs.jar - C:\Program Files (x86)\Java\jre7\lib\ext
```

Step #5: By default, the "Libraries" tab is opened. If not, click on the "Libraries" tab. Then, click on the "Add External Jars…" button. Browse to the location where we have saved the extracted folder for Java Client Libraries.

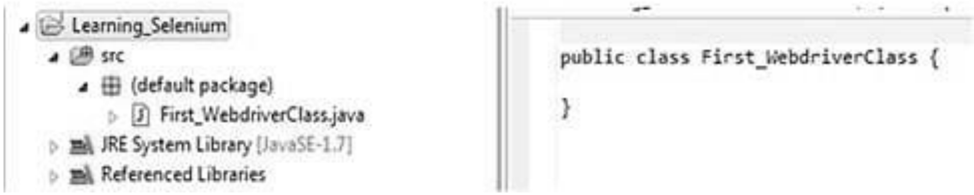
Step #6: Select all the JAR files present in the "selenium-java-2.41.0" folder and click on the open button within the dialogue box. The properties dialogue box should look like the below illustration.



```
apache-mime4j-0.6.jar - F:\Work\Selenium\selenium-java-2.41.0\seleniu
bsh-1.3.0.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2.41.0\li
cglib-nodep-2.1_3.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium
commons-codec-1.8.jar - F:\Work\Selenium\selenium-java-2.41.0\seleni
commons-collections-3.2.1.jar - F:\Work\Selenium\selenium-java-2.41.0\
commons-exec-1.1.jar - F:\Work\Selenium\selenium-java-2.41.0\seleniu
commons-io-2.2.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-
commons-jxpath-1.3.jar - F:\Work\Selenium\selenium-java-2.41.0\seleni
commons-lang3-3.1.jar - F:\Work\Selenium\selenium-java-2.41.0\seleni
commons-logging-1.1.1.jar - F:\Work\Selenium\selenium-java-2.41.0\sel
cssparser-0.9.11.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2
guava-15.0.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2.41.0
hamcrest-core-1.3.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium
hamcrest-library-1.3.jar - F:\Work\Selenium\selenium-java-2.41.0\seleni
htmlunit-2.13.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2.4
htmlunit-core-js-2.13.jar - F:\Work\Selenium\selenium-java-2.41.0\seleni
httpclient-4.3.1.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2.
httpcore-4.3.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2.41.
httpmime-4.3.1.jar - F:\Work\Selenium\selenium-java-2.41.0\selenium-2
```

Step #7: Click on the “OK” button within the dialogue box to complete the configuration part of the Selenium Libraries in our Java project.

The project will look like this:



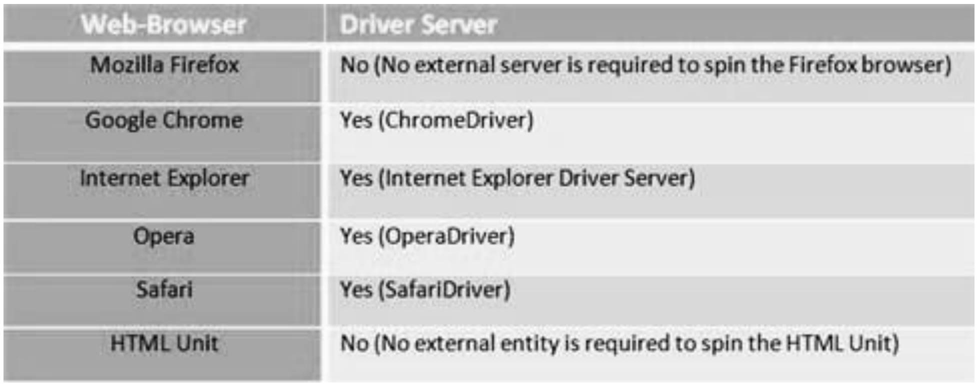
```
public class First_WebdriverClass {

}
```

Available Drivers

There are several driver classes available in WebDriver. In WebDriver, a few of the browsers can be automated directly whereas some of the web browsers require an external entity to automate and execute the test script. This external entity is known as the Driver Server. Thus, the user is required to download the Driver Server for different web browsers. There is a separate Driver Server for each of the web browsers and the user cannot use one Driver Server for web browsers other than the one it is designated for.

Below is the list of available web browsers and their corresponding Server Drivers.



Web-Browser	Driver Server
Mozilla Firefox	No (No external server is required to spin the Firefox browser)
Google Chrome	Yes (ChromeDriver)
Internet Explorer	Yes (Internet Explorer Driver Server)
Opera	Yes (OperaDriver)
Safari	Yes (SafariDriver)
HTML Unit	No (No external entity is required to spin the HTML Unit)

X Precautions

1. Ensure that the WebDriver are fully compatible before starting the configuration.
2. Download all required drivers and libraries from official sources.

XI Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. What are the steps to install Selenium WebDriver?
2. Which programming languages can be used with Selenium WebDriver?
3. Define Metrics and Measurement. Types of Metrics, Product Metrics and Process Metric

Space for Answer

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

……………………………………………………………………………………………

XII References

1. https://searchsoftwarequality.techtarget.com/definition/automated-software-testing.
2.https://www.softwaretestinghelp.com/automation-testing-tutorial-1/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 25 : *Browser Automation with WebDriver

I Practical Significance

This practical focuses on browser automation with Selenium WebDriver is practically significant because it enables testers to automate repetitive and time-consuming tasks involved in web application testing, such as form submissions, navigation, data validation, and workflow verification. Automated browser testing ensures consistent execution, reduces human error, speeds up regression testing, and allows testing across multiple browsers and operating systems efficiently. It also provides reliable validation of application functionality, performance, and user interactions, which improves overall software quality, reduces manual effort, and enhances productivity for QA teams.

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO5: Apply automation testing tools to test software.

IV Laboratory Learning Outcome(s)

LLO 25.1: Apply browser automation techniques using Selenium WebDriver to automate tasks such as opening a URL, navigating, and closing.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

Selenium WebDriver is an open-source tool designed for automating interactions with web browsers by directly communicating with browser engines, allowing precise control over web elements and user actions. The theoretical foundation of browser automation with WebDriver is based on functional testing, automated regression testing, and cross-browser testing principles**,** which ensure that web applications behave as expected under different conditions. Test case design involves defining structured steps with specific inputs, expected outcomes, and validations, while execution allows systematic detection of defects, performance issues, or workflow inconsistencies. Browser automation ensures reproducibility, reliability, and efficiency in testing, forming a key part of modern test automation strategies.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Start the system and open the IDE (e.g., Eclipse, PyCharm, or Visual Studio Code).
2. Create a new project for automation testing.
3. Install or add Selenium WebDriver library to the project.
4. Import the necessary Selenium WebDriver packages into the code.
5. Initialize the WebDriver object for the selected browser.
6. Launch the browser using the WebDriver instance.
7. Open the target website or application URL using the get() method.
8. Locate the required web elements using locators such as id, name, class, xpath, or css selector.
9. Validate the expected output by checking title, URL, or specific element text.
10. Close the browser window using close() or quit() method.
11. Record the test results (Pass/Fail) based on the validation outcome.
12. Prepare the final test report summarizing steps executed and results obtained**.**

IX Precautions

1. Handle computer system with care.
2. Don't forget to save file before execution.
3. Follow safety Practices

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. How do you close a browser window using WebDriver?
2. How do you navigate to a specific URL in Selenium WebDriver? What are the steps to install Selenium WebDriver?
3. How do you launch a browser using Selenium WebDriver?
4. Write a program to open a website (e.g., https://www.google.com) using WebDriver.

Space for Answer

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

…………………………………………………………………………………………………

XI References

1. https://searchsoftwarequality.techtarget.com/definition/automated-software-testing.
2.https://www.softwaretestinghelp.com/automation-testing-tutorial-1/

XII Assessment Scheme (25 Marks)

Performance Indicators		Weightage
Process related (15 Marks)		60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
Product related (10 Marks)		40 %
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
Total 25 Marks		100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	

Practical No. 26 : Handling Multiple Windows and Tabs in WebDriver

I Practical Significance

Handling multiple windows and tabs in Selenium WebDriver is practically significant because most modern web applications involve interactions with multiple browser windows or pop-ups. For example, during login with third-party authentication, downloading files, or opening links in new tabs, testers need to switch between windows to perform validations. This concept Web applications involve interactions with multiple browser windows or pop-ups. For example, during login with third-party authentication, downloading files, or opening links in new tabs, testers need to switch between windows to perform validations. This concept helps automate real-world scenarios where operations span across multiple windows. By using WebDriver methods such as getWindowHandle(), getWindowHandles(), and switchTo().window(), testers can efficiently control different browser contexts, verify data consistency, and ensure that navigation between windows works correctly. Hence, mastering this concept improves test coverage, enhances automation reliability, and simulates real user behavior more accurately

II Industry / Employer Expected Outcome(s)

Develop strong testing skills along with proficiency in tools like Selenium to ensure software quality.

III Course Level Learning Outcomes(s)

CO5: Apply automation testing tools to test software.

IV Laboratory Learning Outcome(s)

LLO 26.1: Apply techniques to automate switching between multiple browser windows or tabs using Selenium WebDriver.

V Relevant Affective Domain related Outcomes

1. Responsibility: Ensuring correctness of input-output testing.
2. Attention to Detail: Observing accuracy in each calculation operation.
3. Ethical Awareness: Reporting defects genuinely and transparently.

VI Relevant Theoretical Background

In Selenium WebDriver, handling multiple windows and tabs is based on the concept of window handles**,** which are unique identifiers assigned to each browser window or tab opened during a session. WebDriver provides methods like getWindowHandle() to retrieve the current window's handle and getWindowHandles() to obtain a set of all open window handles. The switchTo().window(handle) command is then used to shift the driver's focus to a specific window or tab. This allows automated scripts to perform operations such as clicking links that open new windows, verifying content across tabs, or closing unwanted pop-ups. Theoretical understanding of this mechanism is essential to ensure that automated tests can mimic real user actions across multiple browser contexts without losing control of the session.

VII Recourses Required

Sr. No.	Name of Resources	Specifications	Quantity	Remarks
1	Hardware: Computer Systems	Computer i3 or above RAM minimum 2 GB	As per batch Size	--
2	Operating System	Windows 10 onwards / LINUX latest Version		
3	Tools / Software Required	MS Word / Excel / IDE (Python, Java, etc.)		

VIII Procedure

1. Launch the browser using WebDriver.
2. Open the main (parent) webpage using driver.get().
3. Store the main window handle using driver.current_window_handle.
4. Perform an action that opens a new window or tab (e.g., click a link or button).
5. Get all window handles using driver.window_handles.
6. Switch control to the new window using driver.switch_to.window(handle).
7. Perform required actions in the new window or tab.
8. Close the new window using driver.close().
9. Switch control back to the main (parent) window using its handle.
10. Continue remaining operations or close the browser using driver.quit().

IX Precautions

1. Handle computer system with care.
2. Don't forget to save file before execution.
3. Follow safety Practices

X Practical related questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write a WebDriver script to open a webpage and click a link that opens in a new window.
2. How to get the window handle of the parent window and newly opened child windows? Provide an example.
3. Write a program using WebDriver to open multiple tabs and switch between them using window handles.
4. Write a WebDriver script to handle more than two browser windows and perform actions on each one.

5. Write a script to extract text from a newly opened tab and print it in the console.How do you close a browser window using WebDriver?

Space for Answer

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

XI References

1. https://searchsoftwarequality.techtarget.com/definition/automated-software-testing.

2.https://www.softwaretestinghelp.com/automation-testing-tutorial-1/

XII Assessment Scheme (25 Marks)

	Performance Indicators	Weightage
	Process related (15 Marks)	60 %
1	Application of ECP and BVA in test case design	20 %
2	Execution accuracy and documentation quality	30 %
3	Accuracy in execution and documentation	10 %
	Product related (10 Marks)	40 %
4	Quality and completeness of designed test cases	20 %
5	Answer to sample questions	20 %
	Total 25 Marks	100 %

Marks Obtained			**Dated Signature of Teacher**
Process-related Assessment 15 marks	**Product-related Assessment 10 marks**	**Total (25 marks)**	