

QA Workshop – Day 3

BIT – 5th Semester (Professional Class)

Facilitator: **Neha Rouniyar**



Test Design Techniques

Effective test design is vital for quality software, helping create robust, manageable test cases by maximizing coverage and minimizing redundancy.

These are necessary for :



Structured Test Cases

Methods for creating effective and robust test scenarios.



Optimize Effort

Streamline testing processes and reduce redundant work.



Maximize Coverage

Ensure comprehensive testing across all critical areas to avoid missing key issues.

Types include :

Equivalence Partitioning

Divides input data into equivalent partitions, testing one representative from each for similar behavior.

Boundary Value Analysis

Tests the 'edges' of input domains to identify defects where valid and invalid ranges meet.

Decision Table Testing

Systematically maps combinations of conditions and actions for complex business rules, ensuring comprehensive logical path coverage.



Boundary Value Analysis (BVA)

BVA is a technique where test cases are designed at the "edges" or limits of valid input ranges. Bugs frequently occur at these boundaries.

- **Principle:** Focus on values just inside, on, and just outside valid boundaries.
- **Example:** For an age input field allowing 18 to 60 years:
 - 17 (below lower limit)
 - 18 (lower boundary)
 - 19 (just above lower)
 - 59 (just below upper)
 - 60 (upper boundary)
 - 61 (above upper limit)

BVA Example: Password Length

Consider a password field requiring 8 to 20 characters. BVA significantly reduces the number of test cases needed.



7 Characters

(Invalid)



8 Characters

(Valid - Lower boundary)



9 Characters

(Valid - Just above)



19 Characters

(Valid - Just below)



20 Characters

(Valid - Upper boundary)



21 Characters

(Invalid)

Equivalence Partitioning (EP)

EP divides input data into equivalence classes (partitions) where all values in a class are expected to behave similarly. You only need to test one representative value from each partition.

- **Principle:** Create valid and invalid groups; test one value per group.
- **Example:** For a 'Marks' input field (0-100):

1	Partition 1: < 0 (Invalid Range)
2	Partition 2: 0 - 100 (Valid Range)
3	Partition 3: > 100 (Invalid Range)

Representative values to test: **-5** (Invalid), **50** (Valid), **120** (Invalid).

Kestall Dac Ramags: Maæbo vat Data Raigue	
VALILT	INVITAB
- TES -	Æ71H:²
- TEST	- Test 84℥', st. Vatruæ

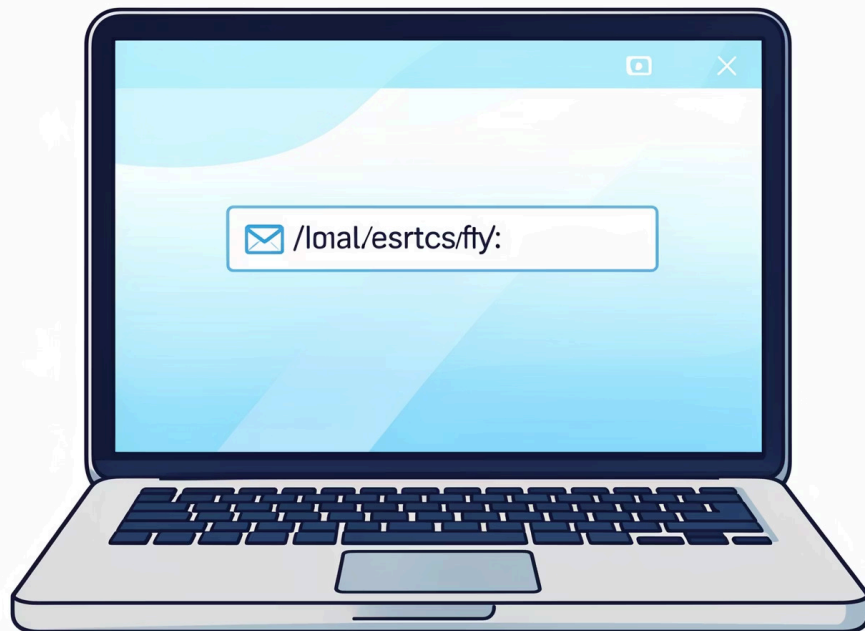
EP Example: Email Field

Applying Equivalence Partitioning to an email input field helps identify critical test cases without exhaustive testing.

Valid Email Partition

Includes correctly formatted emails:

- abc@gmail.com
- test@yahoo.com
- user.name@domain.co.uk



Invalid Email Partitions

Includes common errors:

- Missing "@" symbol
- Missing "." in domain
- Emails with spaces
- Invalid characters (e.g., # \$ % ^)



Instead of testing dozens of email formats, choose 1-2 from each partition.

BVA vs. EP: A Simple View

Both techniques are crucial for efficient testing. Using them together provides robust test coverage.

BVA	Tests specific limits and boundaries of input ranges.	Age: 17, 18, 19, 59, 60, 61
EP	Tests distinct groups of valid and invalid inputs.	Valid vs. Invalid input types (e.g., Marks: -5, 50, 120)

Combining BVA and EP creates a powerful testing strategy that optimizes time and identifies more defects.

Decision Table Testing (Basics)

Decision Table Testing is a technique used to test different combinations of inputs and their corresponding outputs. It's excellent for clearly defining complex business logic and identifying missing conditions or rules.

Clear & Structured

Provides a systematic and structured way to map conditions to actions, making test case design transparent.

Prevents Gaps

Helps identify all possible combinations, ensuring no conditions or business rules are overlooked.

Complex Logic

Ideal for testing systems with multiple interdependent rules, such as forms, calculations, or approval workflows.

Decision Table Structure

- **Conditions:** Input factors or variables that influence the outcome.
- **Actions:** The expected outputs, system responses, or resulting behaviors.
- **Rules:** Each column represents a unique combination of conditions leading to specific actions.

Mini Example: Password Rules

Condition	Rule 1	Rule 2	Rule 3	Rule 4
Length \geq 8	Y	Y	N	N
Has number	Y	N	Y	N
Output	Pass	Fail	Fail	Fail



Exploratory Testing

Exploratory testing is a powerful, hands-on approach where testers simultaneously design and execute tests, using their experience to uncover unexpected bugs.

 www.saucedemo.com

Swag Labs

Sauce Labs Swag Labs app



No Fixed Test Cases

Testers rely on intuition and creativity to navigate the application.



Experience-Driven

Leverages the tester's knowledge and observational skills.



Finds Hidden Bugs

Excellent for discovering edge cases and usability issues that might be missed by scripted tests.

 Ideal when requirements are unclear, the application is new, or quick feedback is essential.

Exploratory Testing Examples

Imagine testing an e-commerce website. Exploratory testing encourages dynamic interaction to expose potential defects.

Random Searches

Enter unusual keywords and observe results.

Rapid Actions

Quickly add/remove items, change quantities multiple times.

Invalid Inputs

Try invalid coupon codes, special characters in fields.

Unusual Navigation

Repeatedly use the back button, open many tabs.

This dynamic approach often uncovers real-world issues that standard, planned tests might overlook, leading to a more robust product.

Negative Testing

Negative testing verifies that a system correctly handles invalid, unexpected, or out-of-range inputs. It ensures graceful error management and prevents crashes or data corruption when presented with non-standard data.

Invalid Data Types

Entering text where a number is expected (e.g., "abc" in an age field).

Out-of-Range Values

Submitting values beyond acceptable limits (e.g., a quantity of -1 or an age of 200).

Special Characters

Using unsupported characters or malicious scripts in input fields.

Null or Empty Values

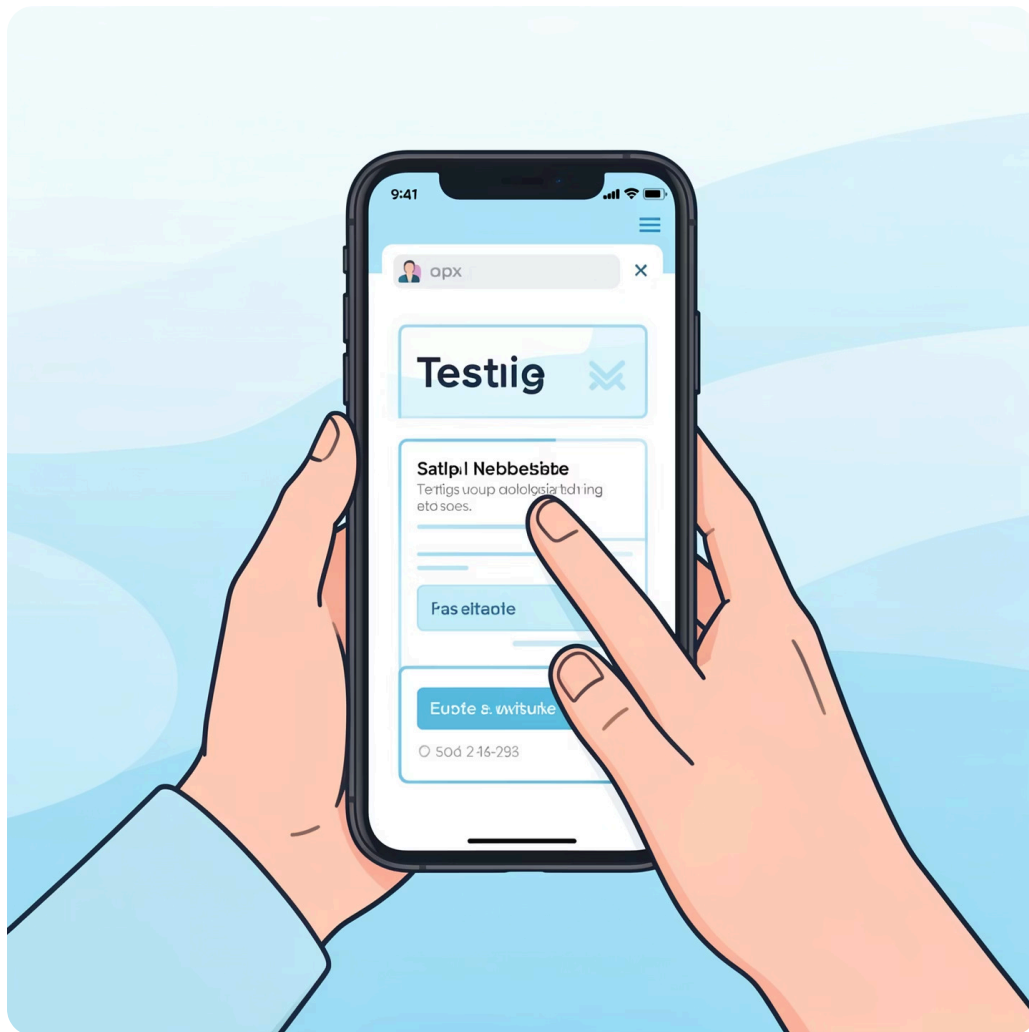
Leaving required fields blank or submitting null data where input is mandatory.

Activity & Key Takeaways

Apply what you've learned. Choose a simple input field and practice these techniques.

Hands-on Activity:

- Select an input field (age, password, phone number).
- Generate BVA test values.
- Define EP partitions.
- Perform 10 minutes of exploratory testing on any website.



Summary of Techniques:

- **BVA:** Crucial for testing boundaries and limits.
- **EP:** Efficiently tests valid and invalid data partitions.
- **Exploratory:** Uncovers real-world issues through free-form testing.
- **Negative :** validates how a system handles invalid input

Mastering these techniques will significantly improve your testing effectiveness.



Questions & Discussion

QA Assignment – Session 3

1

Step 1: Choose an App

- **Apps:** eSewa/Khalti, Daraz, Foodmandu, Calculator, Khan Academy or any app of your choice (non social media only)

2

Step 2: Identify Testing Types

- Functional, Performance, Usability, Security, Compatibility, Regression
- List **at least 5 types** with a short explanation

3

Step 3: Design Test Scenarios

- **BVA:** Pick an input, identify valid & invalid boundaries, write 4–6 scenarios
- **Equivalence Partitioning:** Divide inputs into valid & invalid groups, write 4–6 scenarios

4

Step 4: Submission

- Prepare a **document** with app, testing types, BVA & EP scenarios
- Share before or bring it to the next class