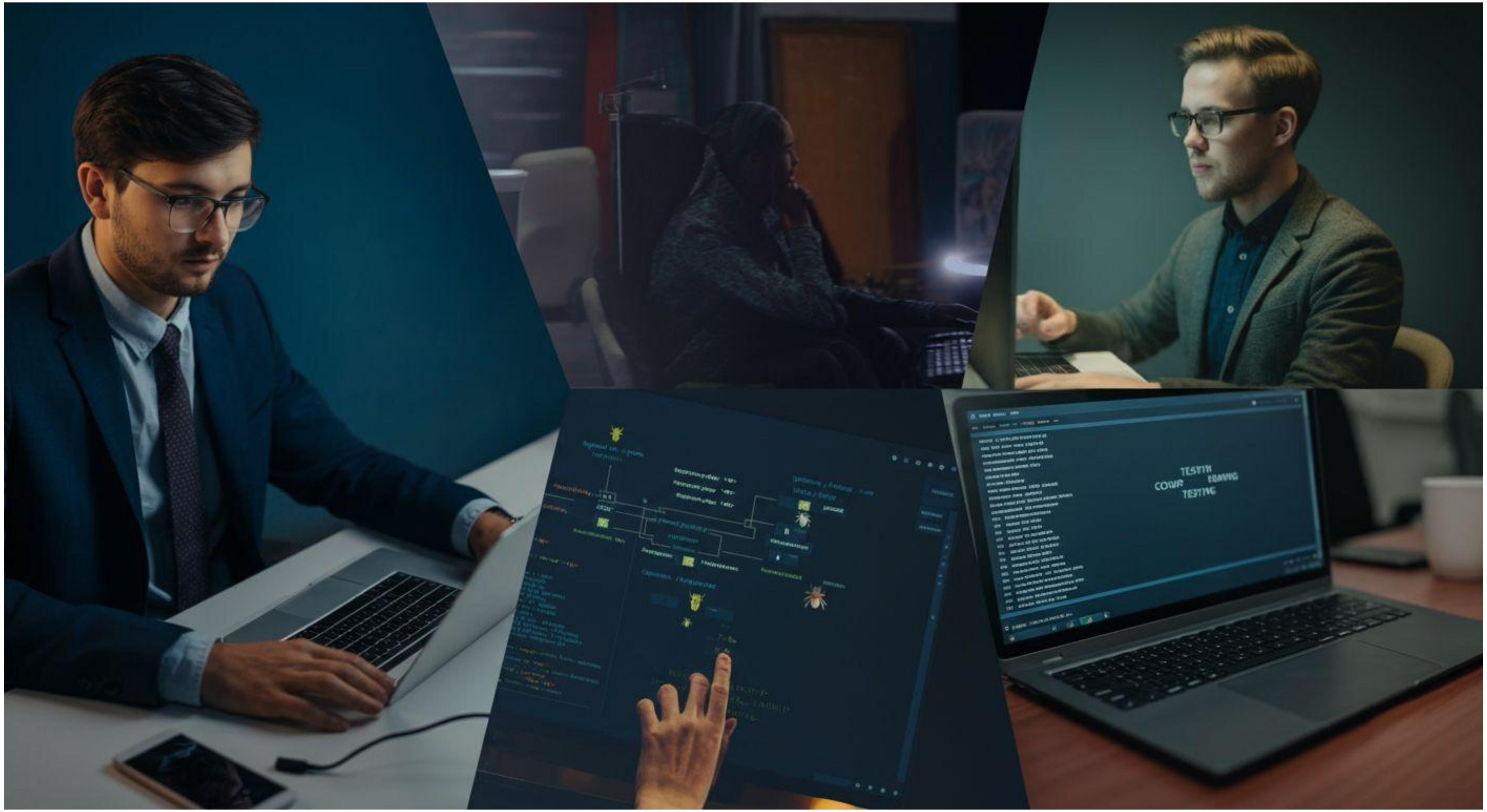
APPLIED SOFTWARE TESTING



Intro

Meet your instructor

Who Am I?

Why I am teaching?

- *To learn*
- *To share what I know*

Intro

Muhammad Kamran

Contour Software

Experience: Quality Assurance | Testing | Automation (22 Years)
Visiting Faculty at **M. Ali Jinnah University, FAST-NU, SZABIST, Jinnah University for Women**

Certifications, Training & Diploma :

CSM® | CSPO® | CSQE | SAFe® 5.0 | SFPC® | ACP | Kanban Foundation KIKF® | SFC® | Six Sigma Yellow Belt® | CSFPC® | SMPC® | Project Management

Reach me 👇

LinkedIn # muhammadkamran22

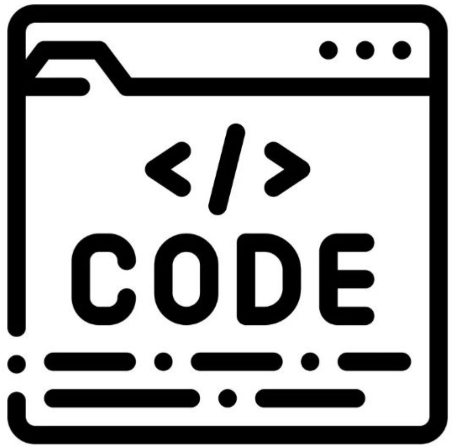
Class Introduction

Name

Work

Purpose of pursuing MS

Teaching Style



Student Evaluation:

Discussion and Case Studies

Class Assignments / Home Assignments

Group Project

Class Participation / Attendance

Mid

Final Exams

Content

- What is Testing?
- Why is Testing Necessary?
- Quality vs Testing
- Testing Objectives
- General Testing Principles
- Test Process

What is TESTING?

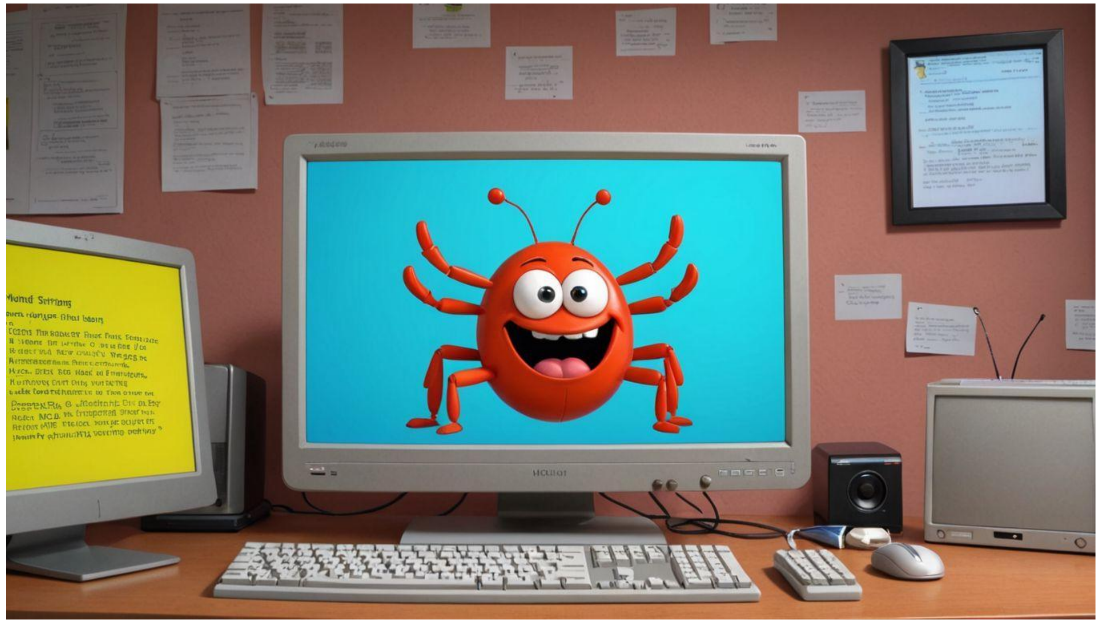
Most people have an experience with software that did not work as expected. Software that does not work correctly can lead to many problems, including:

- Loss of Money
- Time
- Business Reputation
- Injury or Death

"Software testing is **a way to assess the quality** of the software and to reduce the risk of software failure in operation."

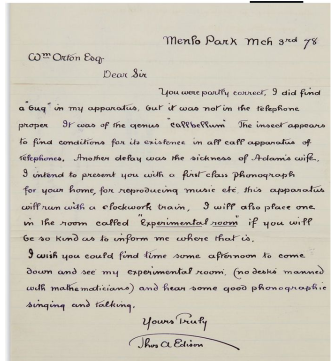
What is TESTING?

Origin of the Term “Bug”



What is TESTING?

- Many people think the term ***bug*** was invented by the computer programmer.
- Thomas Edison reportedly used the term "bug" in the 1870s to refer to technical issues in electrical circuits.
- In his notes on his lighting inventions, one of his entries reads, **“Awful lot of bugs still.”**
- Engineers and inventors during the early 1900s used "bug" to describe problems in machinery or systems.



Menlo Park Mch 3rd 78

Wm Orton Esqr

Dear Sir

You were partly correct, I did find a "bug" in my apparatus, but it was not in the telephone proper It was of the genus "callbellum" The insect appears to find conditions for its existence in all call apparatus of telephones. Another delay was the sickness of Adam's wife. I intend to present you with a first class phonograph for your home, for reproducing music etc. this apparatus will run with a clockwork train, I will also place one in the room called "Experimental room" if you will be so kind as to inform me where that is.

I wish you could find time some afternoon to come down and see my experimental room, (no desks manned with mathematicians) and hear some good phonographic singing and talking.

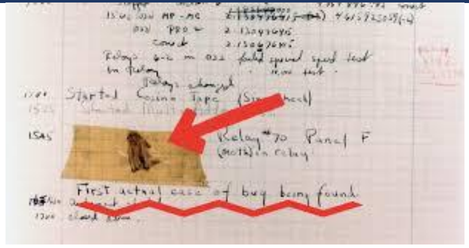
Yours Truly

Thos A Edison

What is TESTING?

The first computer BUG??

What is TESTING?


Started Cosine Tape (Sine check)
Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.

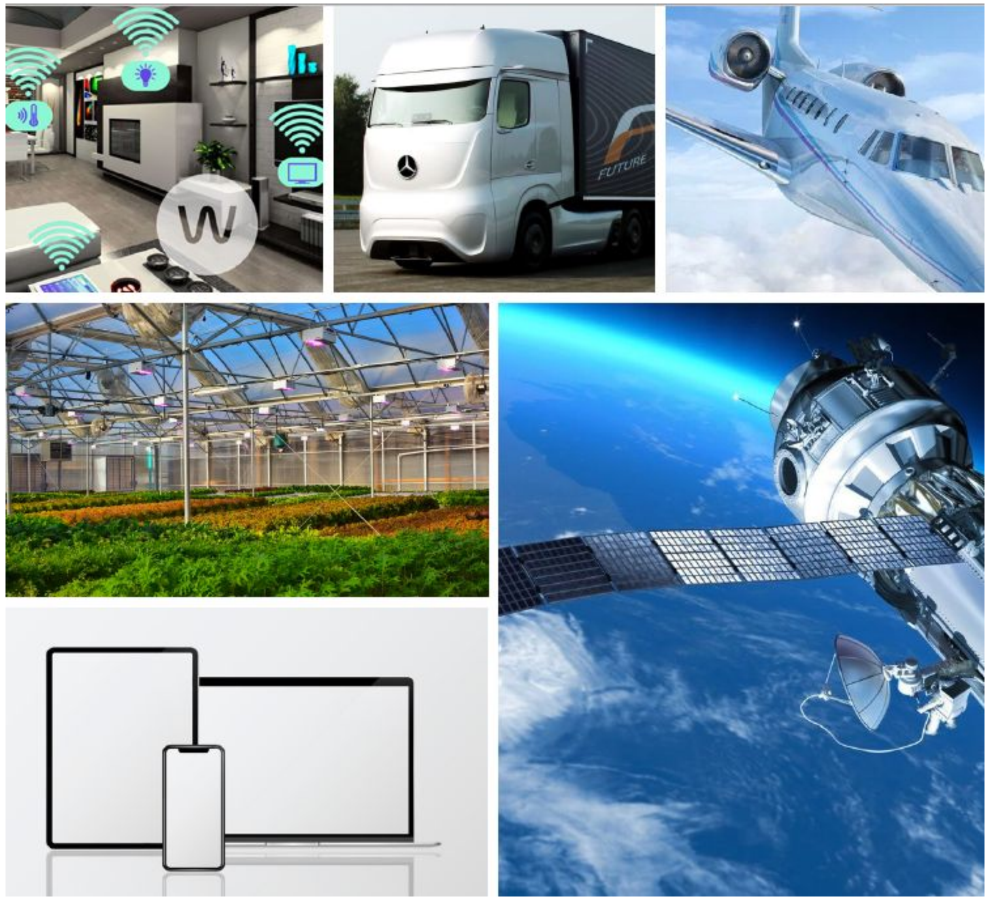
This bug literally a bug.

- This “**first actual case of bug being found**" was recorded by computer scientist **Grace Hopper**
- On **9 Sep 1947**, while working on the **Harvard Mark II**, an electromechanical computer, Hopper and her team found that the machine had malfunctioned.
- Upon investigation, they discovered that a **butterfly** had become stuck in one of the relays, causing a malfunction.
- They removed the butterfly and **"debugged"** the computer, taping the insect into their logbook with the notation **"First actual case of bug being found."**
- This event popularized the use of the term **"bug"** to refer to unexplained glitches or errors in machines, particularly computers.

What is TESTING?

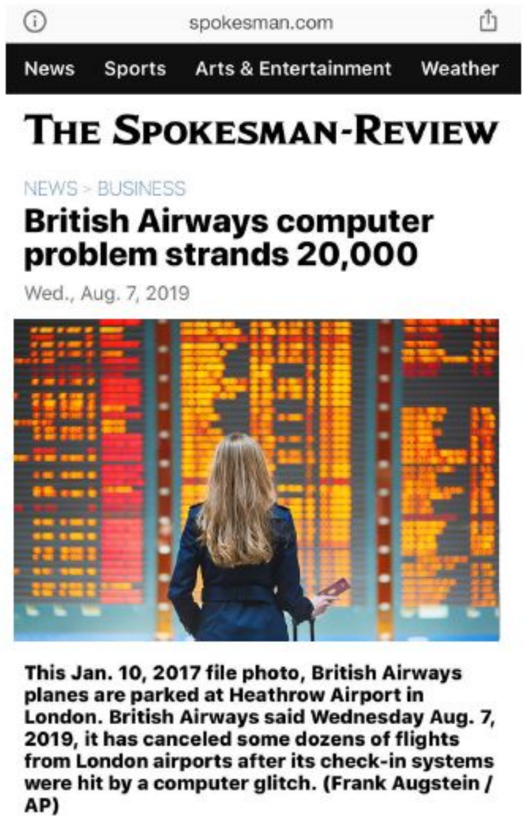
What is TESTING?

Nowadays, software is **everywhere**!



What is TESTING?

So are software bugs...



spokesman.com

News Sports Arts & Entertainment Weather

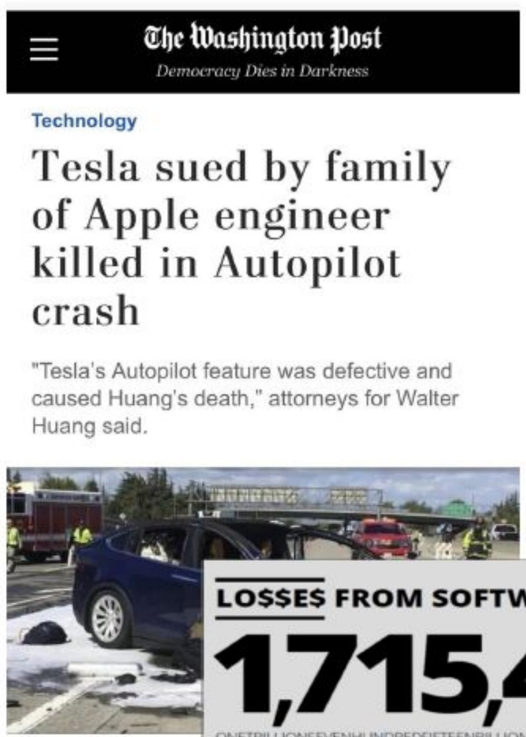
THE SPOKESMAN-REVIEW

NEWS > BUSINESS

British Airways computer problem strands 20,000

Wed., Aug. 7, 2019

This Jan. 10, 2017 file photo, British Airways planes are parked at Heathrow Airport in London. British Airways said Wednesday Aug. 7, 2019, it has canceled some dozens of flights from London airports after its check-in systems were hit by a computer glitch. (Frank Augstein / AP)



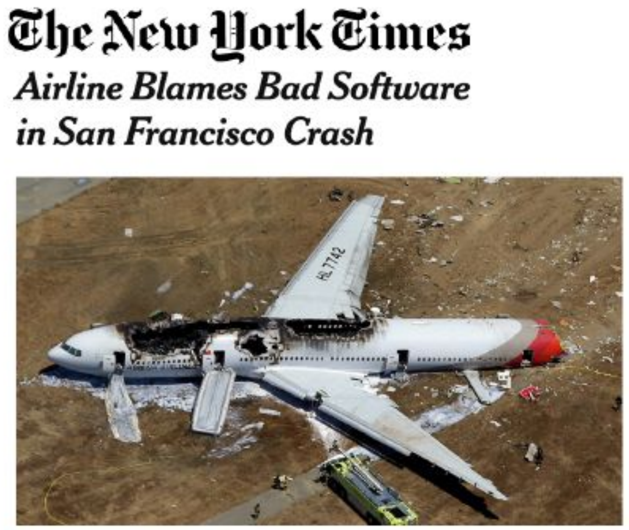
The Washington Post

Democracy Dies in Darkness

Technology

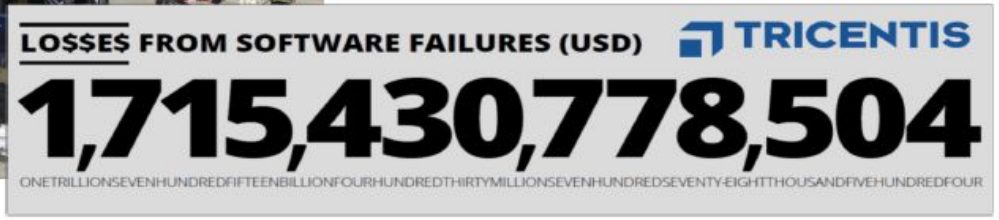
Tesla sued by family of Apple engineer killed in Autopilot crash

"Tesla's Autopilot feature was defective and caused Huang's death," attorneys for Walter Huang said.



The New York Times

Airline Blames Bad Software in San Francisco Crash



LO$$E$ FROM SOFTWARE FAILURES (USD) TRICENTIS

1,715,430,778,504

ONETRILLIONSEVENHUNDREDFIFTEENBILLIONFOURHUNDREDTHIRTYMILLIONSEVENHUNDREDSEVENTY-EIGHTTHOUSANDFIVEHUNDREDFOUR

What is TESTING?

Famous Software Disasters

What is TESTING?

The Millennium bug (Y2K), 2000

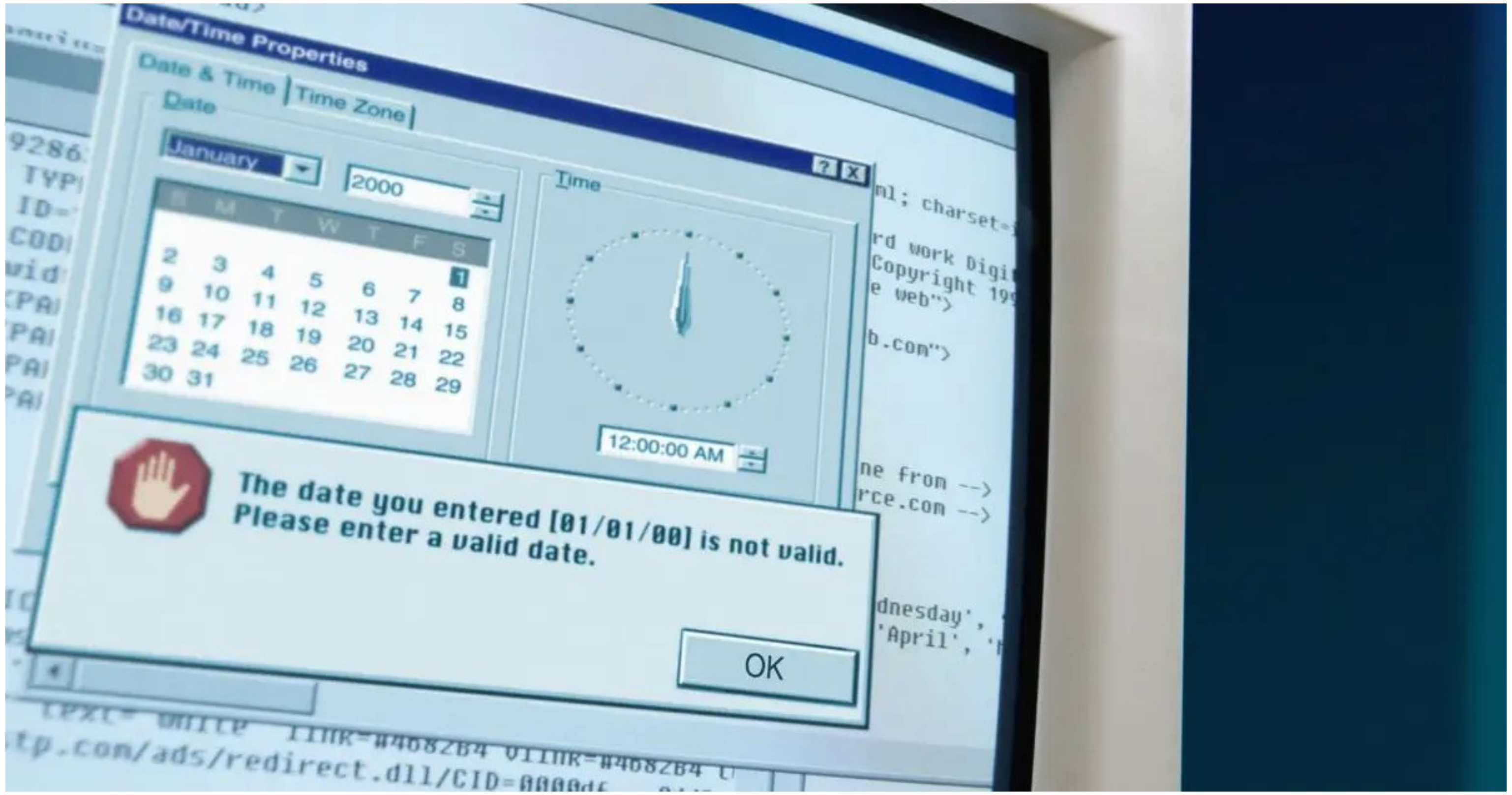
What is TESTING?

The Millennium bug (Y2K), 2000

WEEKLY WORLD NEWS
THE COMPUTER CRASH OF THE MILLENNIUM!
JANUARY 1, 2000
ALL BANKS WILL
FOOD SUP
WILL BE D
THE DAY THE EARTH WILL STAND STILL!
DOMINO EFFECT
A WORLDWIDE DEPRESSION!
WEEKLY WORLD NEWS
THE FINAL DAYS
DOOMSDAY COUNTDOWN
ALSO INSIDE
ARMAGEDDON
YEAR 2000 COMPUTER BUG will turn machine against man!
Hundreds of planes will fall out of the sky!
Cars will stop dead in their tracks!
Nuclear missiles will launch themselves!

What is TESTING?

The Millennium bug (Y2K), 2000



The date you entered [01/01/00] is not valid.
Please enter a valid date.

What is TESTING?

The Millennium bug (Y2K), 2000

- The first baby born in the new millennium in Denmark was 100 years old at birth, according to the hospital computer system
- A man returning a VHS tape to a video rental store in New York was billed for **$91,000** dollars because the system thought he was returning it 100 years late
- U.S. spy satellites malfunctioned, sending unreadable data, for three days.
- Banks calculate rate of interest for 100 years

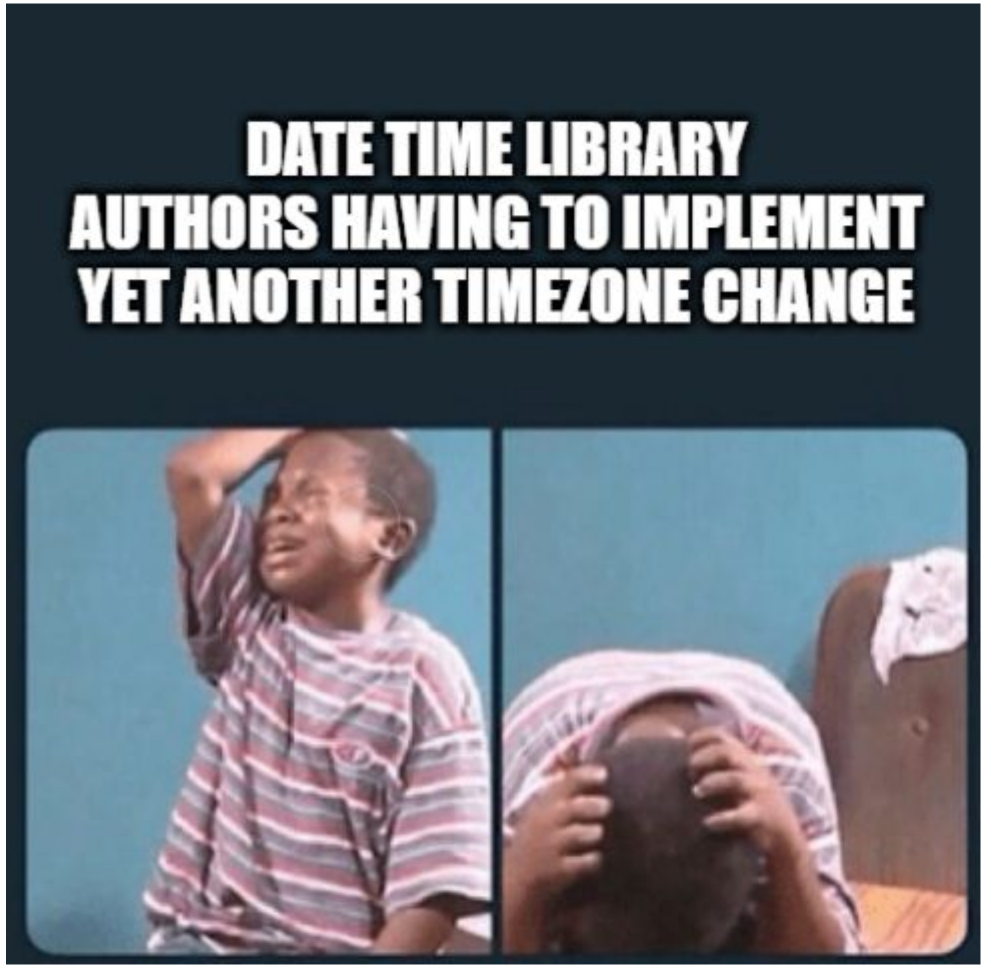
What is TESTING?

The Millennium bug (Y2K), 2000

The solution:

The date was simply expanded to a four-digit number in **programming architectures** and upgrade the system accordingly, but it costs **$300 billon.**

What is TESTING?


DATE TIME LIBRARY AUTHORS HAVING TO IMPLEMENT YET ANOTHER TIMEZONE CHANGE

What is TESTING?

Knight Capital Group Trading Disaster (2012)

- **What Happened:** A software glitch in Knight Capital's trading algorithm caused it to make accidental stock trades, losing **$440 million** in **45 minutes.**

- **Cause:** Outdated code was accidentally deployed, leading to inconsistent behavior in the trading system.

- **Impact:** The company nearly went bankrupt and was acquired soon after.

- **Lesson:** Proper deployment processes and testing are critical in financial systems.

What is TESTING?

Facebook-Cambridge Analytica (2018)

- **What Happened:** A third-party app collected data from 87 million Facebook users without their consent, which was then used for political advertising.
- **Cause:** Weak data privacy controls and lack of oversight in Facebook's API.
- **Impact:** Massive public backlash, regulatory scrutiny, and a **$5 billion** fine for Facebook.
- **Lesson:** Data privacy and ethical considerations must be prioritized in software design.

What is TESTING?

Toyota's Unintended Acceleration (2009-2011)

- Software defects in the electronic throttle control system were linked to unintended acceleration in Toyota vehicles.
- **Impact:** The issue led to a recall of over 9 million vehicles, numerous accidents, and a $1.2 billion settlement, emphasizing the importance of software quality in automotive systems.

Samsung Galaxy Note 7 Battery Explosions (2016)

- A combination of hardware design flaws and potentially inadequate software battery management led to the Samsung Galaxy Note 7 overheating and catching fire.
- **Impact:** Samsung had to recall millions of units, resulting in a loss of $5 billion and damaging the brand's reputation

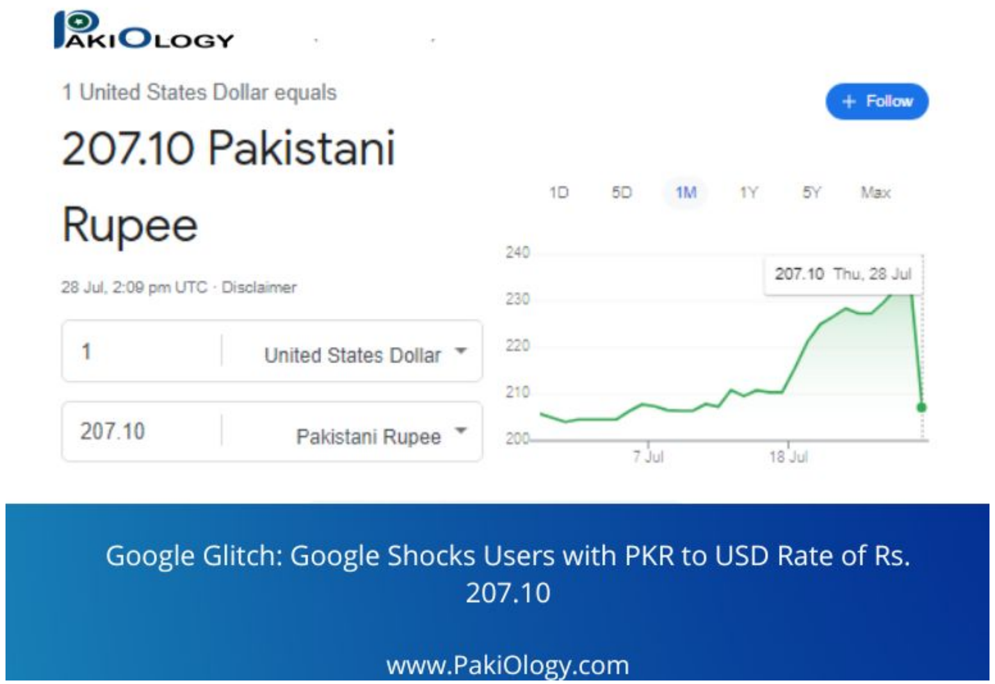
What is TESTING?

25 June 2009

What is TESTING?

Google Bugs ☺️

- 28 July 2022 *Currency exchange bug*


1 United States Dollar equals

207.10 Pakistani Rupee

28 Jul, 2:09 pm UTC · Disclaimer

1 United States Dollar

207.10 Pakistani Rupee

Google Glitch: Google Shocks Users with PKR to USD Rate of Rs. 207.10

www.PakiOlogy.com

What is TESTING?

Some other famous software failures:

- Starbucks was forced to close about 60 percent of stores in the U.S and Canada due to software failure in its POS system. At one point, the store served coffee for free as they were unable to process the transaction.
- Some of Amazon's third-party retailers saw their product price is reduced to 1p due to a software glitch. They were left with heavy losses.
- Boeing 737 Max Crashes (2018-2019) cause 346 deaths.
- Security Issues in Zoom App

What is TESTING?

- **1979:** Software testing is a process of executing a program or application with the intent of finding the errors.
- **1983:** Testing is an activity that measures the Software Quality.
- **2002:** Testing is a concurrent lifecycle process of engineering, using and maintaining testware in order to measure and improve quality of software being tested.

What is TESTING?

Modern Definition:

The **process of verifying and validating** that a software program, application or product:

- Meets the business and technical requirements.
- Works as expected.

Why is Testing Necessary?

Software Testing is necessary because we all make mistakes. Some of those mistakes are unimportant, but some of them are expensive and dangerous.

"We need to check everything and anything we produce because things can always go wrong – **humans make mistakes all the time**."

Quality VS Testing

What are the benefits of Testing?

- **Cost-Effective**: Testing any project on time helps you to save your money for the long term. Bugs caught in the earlier stage of software testing; cost less to fix.
- **Security**: People are looking for trusted products. It helps in removing risks and problems earlier.
- **Product quality:** Testing ensures a quality product is delivered to customers.
- **Customer Satisfaction:** The main aim of any product is to give satisfaction to their customers.

Quality VS Testing

"Quality software is reasonably bug or defect free, delivered on time and within budget, meets requirements and expectations, and is maintainable."

Perception in Testing

Baber

Shaheen

Perception in Testing

Abdul Kalam

Abul Kalam

Find Lion?

Quality VS Testing

GOOD QUALITY ?

Quality VS Testing

Quality

Quality means how good something is. It refers to the features of a product or service that make it valuable or appealing.

Here's how quality is understood in different areas:

- **In business**: Quality means a product or service is good enough for its purpose and meets customer expectations.

- **In engineering and manufacturing**: Quality is about how well a product or service fulfills what customers expect or need.

Quality VS Testing

Key Aspects of Software Quality:

- **Good UI design** – looks and style
- **Good functionality** – it does the job well
- **Reliable** – acceptable level of breakdowns or failure
- **Consistent**
- **Good after sales service**
- **Value for money**

Quality VS Testing

Quality VS Testing

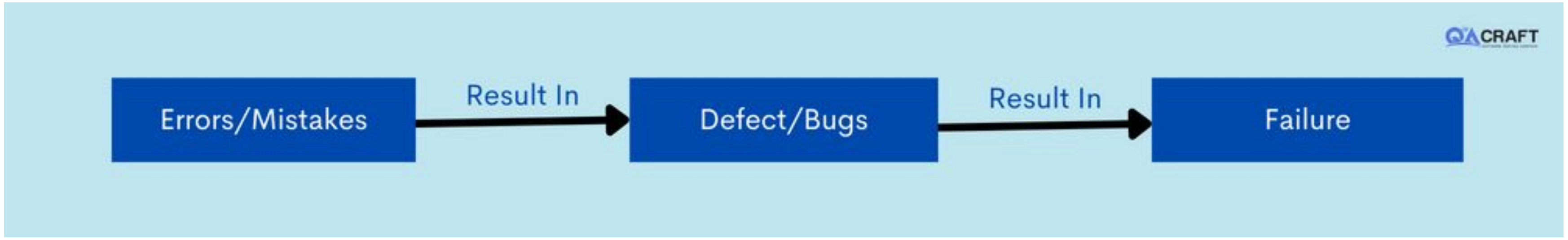
"**TESTING** is the <u>measurement</u> of **QUALITY**"

General Testing Terms

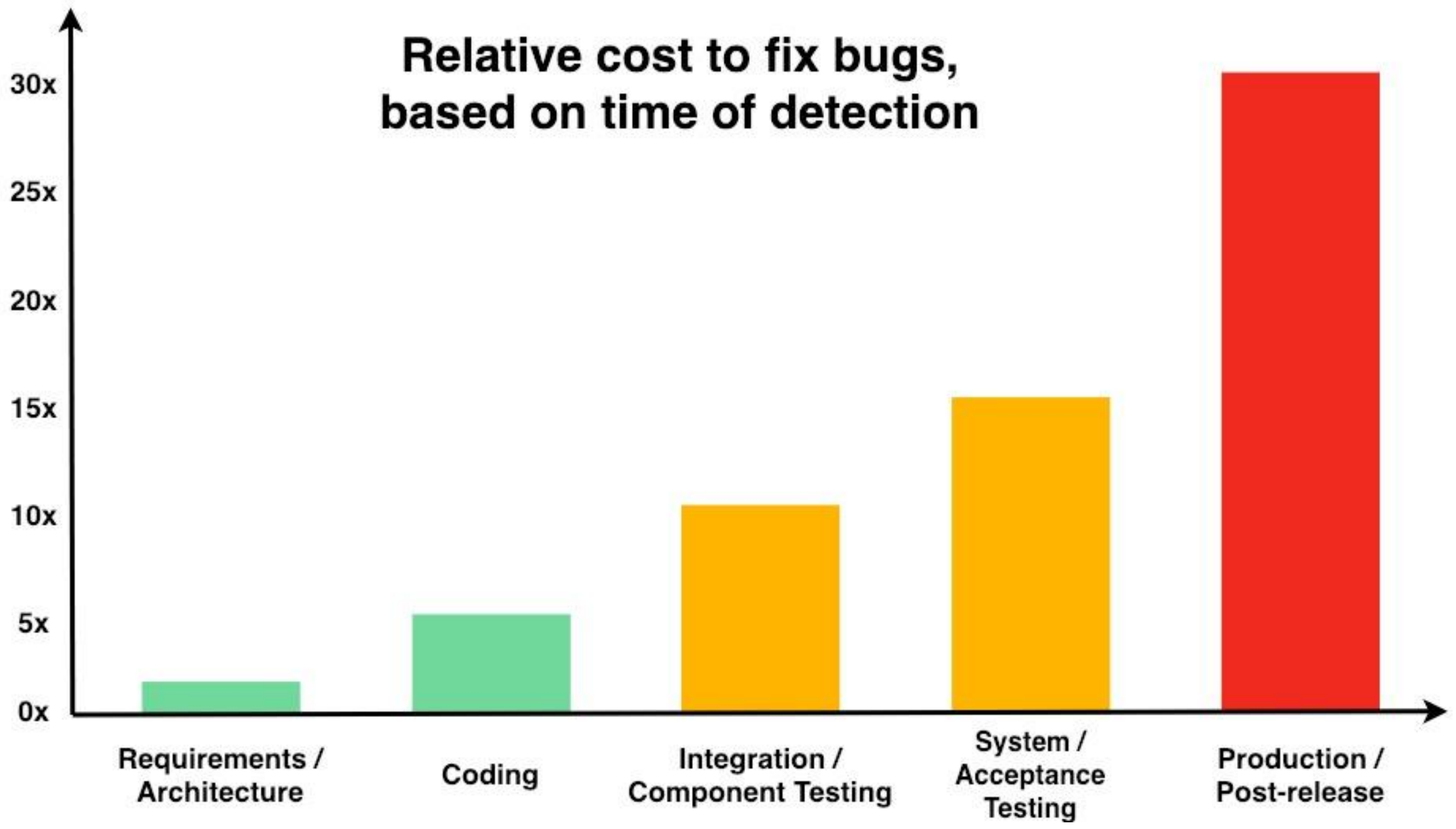
- A human being can make an error (mistake).
- This error can produce a defect (fault, bug) in the program, code, or document.

If the defect in the code is executed:

- The system may fail to perform its intended function.
- The system may perform an unintended action, causing a failure.



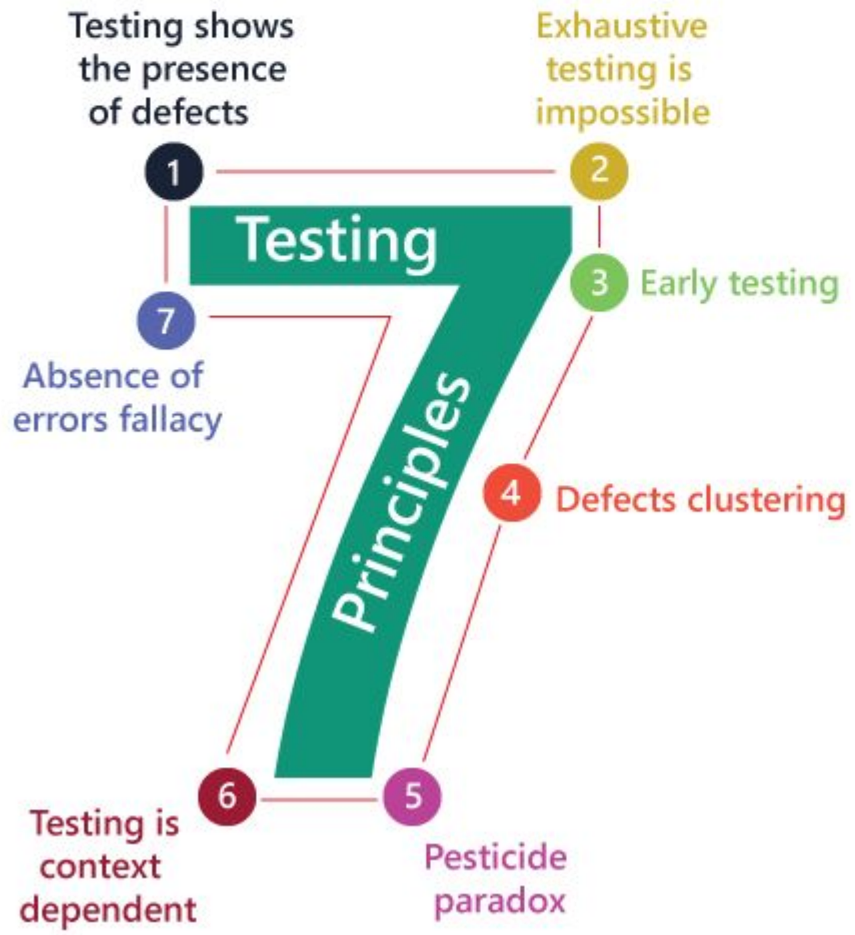
Cost of Defect



Relative cost to fix bugs, based on time of detection
30x
25x
20x
15x
10x
5x
0x
Requirements / Architecture
Coding
Integration / Component Testing
System / Acceptance Testing
Production / Post-release

General Testing Principle

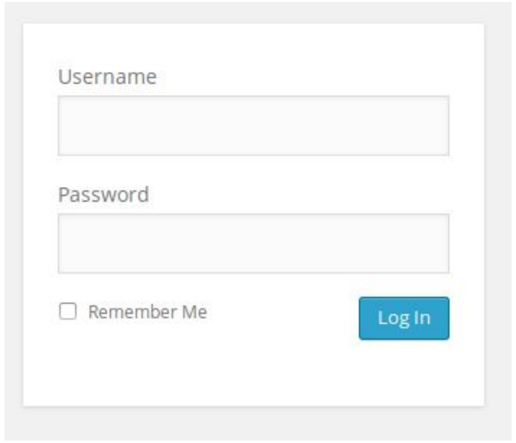
1. Exhaustive testing is not possible
2. Testing shows the presence of defects
3. Early testing saves time and money
5. Beware of the pesticide paradox
6. Defects cluster together
7. Testing is context dependent
8. Absence-of-errors is a fallacy


Testing shows the presence of defects
1
Exhaustive testing is impossible
2
Testing
3
Early testing
7
Absence of errors fallacy
Principles
4
Defects clustering
6
5
Testing is context dependent
Pesticide paradox

Seven Testing Principles

1. Exhaustive testing is impossible

- It is difficult to test all the functionalities with valid and invalid combinations of input data during actual testing.
- Testers must prioritize test cases based on risk, critical functionality, and usage patterns to optimize testing efforts.
- **Exhaustive testing** will take **unlimited efforts** and most of those efforts are ineffective
- We need the optimal amount of testing based on the risk assessment of the application.
- Let’s do an exercises



Seven Testing Principles

Scenarios:

- All the controls including text boxes, buttons, and links are present on the Login page.
- The font specifications of the labels and the text written on the different elements should be clear.
- The size, color, and UI of the different elements are as per the specifications.
- The application's UI is responsive i.e. it should adjust to different screen resolutions and devices.
- As soon as the login page opens, by default, the cursor should remain on the username textbox.
- The user is able to navigate or access the different controls by pressing the 'Tab' key on the keyboard.
- The password is in masked form when entered.
- Verify if the password can be copied or not.
- The user is able to log in by entering valid credentials and clicking on the 'Login' button.
- The user is able to log in by entering valid credentials and pressing the Enter key.
- The user is not able to log in with an invalid username and password.
- The validation message gets displayed in case the user leaves the username or password field blank.
- The validation message is displayed in the case the user exceeds the character limit of the user name and password fields.
- Reset button functionality on the login page. Clicking on it should clear the textbox's content.
- Verify if there is a checkbox with the label "remember password" on the login page.
- Closing the browser should not log out an authenticated user. Launching the application should lead the user to a login state only.
- Verify the limit on the total number of unsuccessful login attempts.

Seven Testing Principles

- In case of incorrect credentials, a message like “incorrect username or password” should be displayed instead of an exact message pointing to the incorrect field. Because a message like “incorrect password” will give a hint to the hacker that the username is correct and he just needs to try a different combination on the password field only.
- Verify the login session timeout duration.
- Clicking the back button doesn’t log out the user once is user is logged in.
- Minimum and Maximum lengths should be set for all the text boxes
- Login credentials in the UPPER case should not be treated as invalid
- Validation message should be shown when special characters are entered in the username field, or when an invalid username and/or password is entered, or when the fields are left blank
- Login credentials, especially passwords, should be stored in the database in an encrypted format

Seven Testing Principles

2. Testing shows the presence of defects:

- The purpose of software testing is to detect software failures.
- Testing can reveal defects in the software, but it cannot prove that the software is defect-free. Even if no defects are found, it does not mean the software is perfect
- Even multiple test phases and several rounds of testing cannot guarantee that the software is 100% bug-free.
- Efficient Testing techniques reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.

Seven Testing Principles

Seven Testing Principles

Example :

- We have seen several advertisements for soaps, toothpaste, handwash or disinfectant sprays, etc. on television.
- Consider a handwash advertisement that says on the television that 99% of germs can be removed if that specific handwash is used.
- This clearly proves that the product is not 100% germ-free. Thus, in our testing concept, we can say that no software is defect-free

Seven Testing Principles

3. Early testing saves time and money

- Testers get involved in the early stage to find defects during the **requirement analysis phase**.
- The cost involved in fixing such defects is very less when compared to those that are found during the later stages of testing.
- Testing early in the software development lifecycle helps reduce or eliminate costly changes.
- Testing needs to be performed on requirement documents, specifications or any other type of document so that if requirements are incorrectly defined then it can be **fixed immediately** rather than fixing them in the **development phase**

Seven Testing Principles

4. Defects cluster together

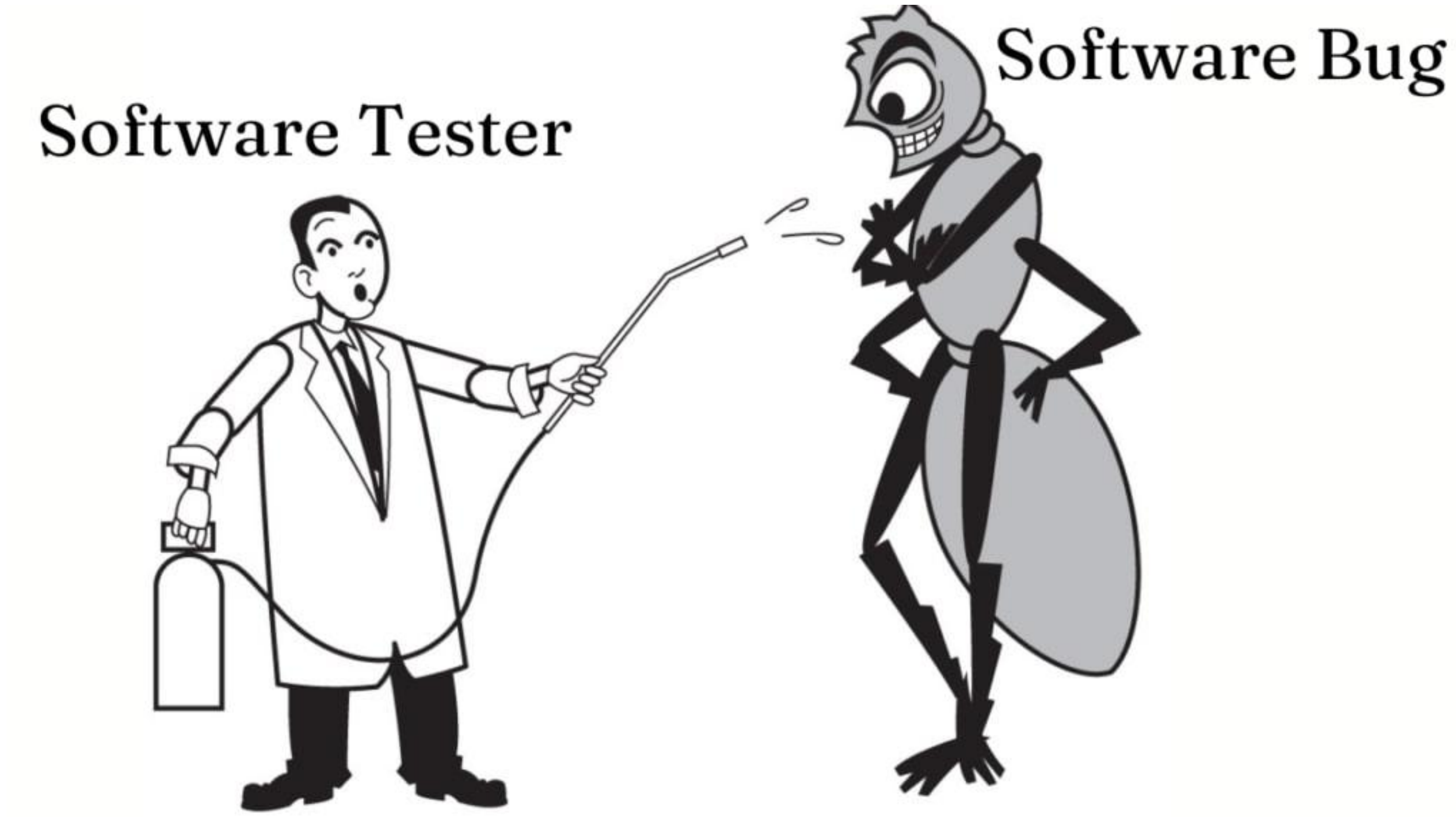
- This principle is an example of the 80:20 rule (also called the Pareto principle) – 80 percent of defects are due to 20 percent of code.
- It is based on the observation that 80 percent of users use 20 percent of the software. It is this 20 percent of the software that will contribute most towards the defects.
- According to the Pareto Principle (*Rule 80-20*), "***80% of the defects come from 20% of the modules and the remaining 20% defects are distributed across the rest 80% of the modules***". So, we should emphasize the testing in 20% of the modules where we face 80% of the errors.
- This principle can help your team focus on the right area - the most popular software part.

- Defect clustering in software testing means that a small module or functionality contains most of the errors or has the most operational failures.

Defects cluster change over time, as the functionality becomes more stable. The testing team should always be on the lookout to find new clusters.

Seven Testing Principles

5. Beware of the pesticide paradox



Seven Testing Principles

5. Beware of the pesticide paradox

- The use of a pesticide makes pests **immune** to its use.

- The pesticide paradox says that if the same tests are repeated, eventually, the same set of test cases will no longer identify any new bugs in the system.

- What's the solution? Rethink and write/modify more and better test cases, especially when it is a crucial part of the application.

- To overcome this 'pesticide paradox,' the test cases need to be regularly reviewed and revised. New and different tests need to be written to exercise different parts of the software to potentially find more defects.

Seven Testing Principles

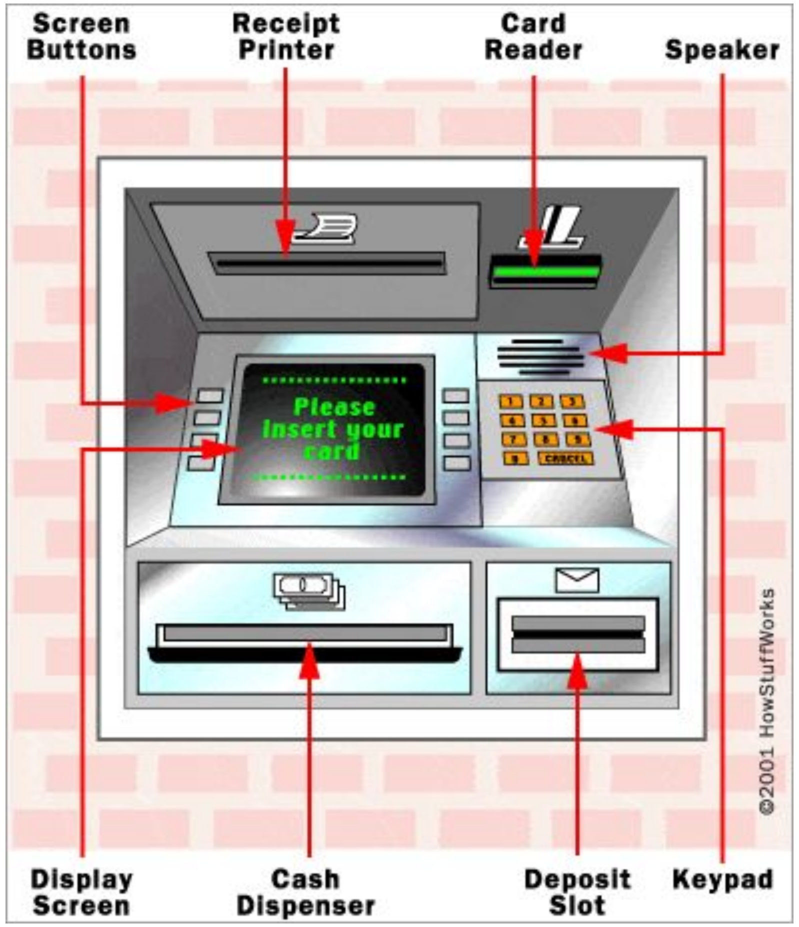
6. Testing is context-dependent

- There are several domains available in the market, such as **banking**, **medical**, **advertisement**, etc. Each domain has its applications with different requirements, functions, and testing purposes. In a nutshell, different domains, and different testing.

- Testing, any POS system at a retail store will be different than testing an ATM machine.

Seven Testing Principles

6. Testing is context-dependent



Seven Testing Principles

6. Testing is context-dependent

- **Nature of the Software:**
- The type of application (e.g., web, mobile) will command different testing priorities and techniques.
- For instance, testing a medical device requires demanding safety and reliability standards, while testing a social media app might focus on user experience and performance.

- **Target Audience:**
- The intended users of the software will influence testing to ensure that the application meets their specific needs and expectations. For example, a video game require larger fonts and good graphics.

- **Regulatory Requirements:**
- Industries like finance, healthcare, and aviation often have strict regulations that dictate testing procedures and documentation. Adherence to these standards is crucial to ensure legal compliance and product safety.

Seven Testing Principles

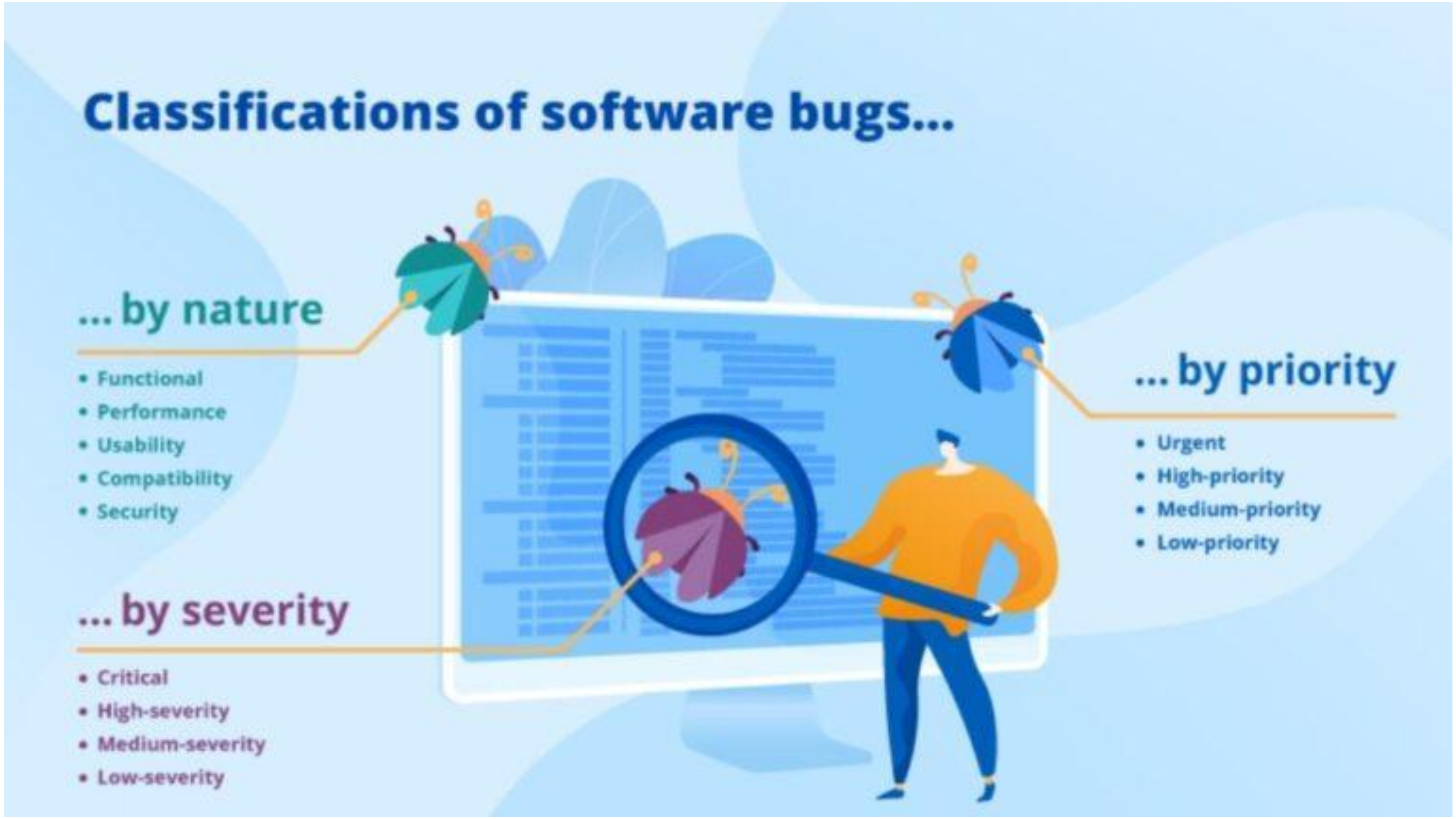
7. Beware of the absence of error fallacy

- It is a common belief that a low defect rate implies the product is a success. This idea is the absence-of-errors **misconception**.
- Zero defects do not mean the software solves end-user problems successfully.
- Sometimes bug-free software can remain unusable if the wrong requirements are incorporated into the software. The software we develop must not only have minimal defects, but it must also meet the needs of the business, otherwise, it becomes unusable.

What the Client Really Need

What Operations Installed

Seven Testing Principles



Testing Objectives

To verify whether all specified requirements have been fulfilled.

- Testing should be to fulfilled according the customer’s needs.
- Testers test the product and ensure the implementation of all the specified requirements
- Tester should create a requirement traceability matrix (***RTM***), which will ensure the mapping of all the test cases to requirements.
- RTM is an effective way to ensure that test cases have got the right requirement coverage.

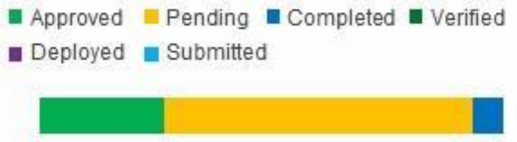
Testing Objectives

Requirements Traceability Matrix with Test Cases

This graph/chart is linked to excel, and changes automatically based on data. Just left click on it and select "edit data".

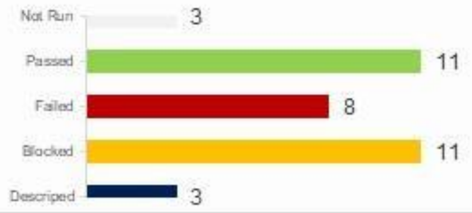
85
Total Requirement

Snapshot of Requirements



1402
Total Linked Test Cases

Snapshot of Test Cases



200
Found Defects

REQ ID / Test Case ID	🟢 0001	🟢 0002	🟢 0003	🟢 0004	🟢 0005	🟢 0006	🔴 0007	🔴 0008	🔴 0009	🟢 0010	🟢 0011	🔴 0012	🔴 0013	🟢 0014	🟢 0015
TC-000001	🟩	🟩	🟩			🟩	🟩	🟩							
TC-000002					🟩	🟩	🟩	🟩	🟥	🟩	🟩	🟩	🟩	🟩	🟩
TC-000003							🟩	🟥	🟥			🟧	🟩	🟩	
TC-000004											🟩	🟩	🟥		
TC-000005													🟩	🟩	🟩
TC-000006						🟧	🟥	🟥			🟩	🟧	🟩		
TC-000007									🟩	🟩	🟩				
TC-000008									🟩						
TC-000009							🟩	🟩	🟩						

Testing Objectives

To validate whether the test object is completed and works as the users/stakeholders expect.

- This idea of testing is called ***Validation***.
- It is the process of checking the product after development.
- Validation can be a manual or **automation**.
- It usually employs various types of testing techniques, i.e., ***Black Box, White Box***, etc.
- Testers perform validation, whereas customers can also validate the product as part of

  User acceptance testing.
- Customer's fulfillment is a major need for any business.

Testing Objectives

Software Testing Objectives

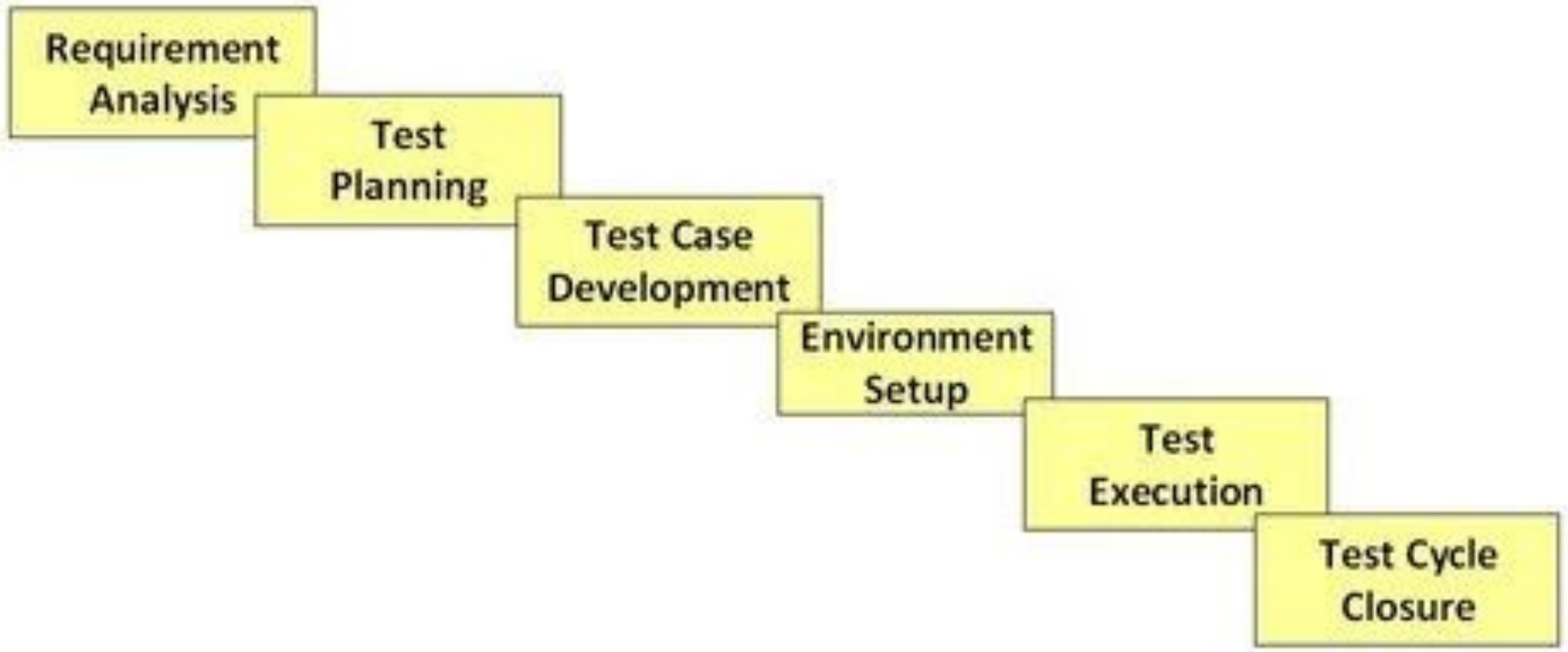
- To find defects
- To prevent defects
- To reduce the level of risk of insufficient software quality

STLC (Software Testing Life Cycle)

- **Software Testing Life Cycle (STLC):** It is a systematic process used in software testing to ensure the quality and reliability of a software product.

- The STLC defines specific stages that need to be followed during the testing process.

STLC (Software Testing Life Cycle)



Each of these stages has a definite Entry and Exit criteria, Activities & Deliverables associated with it.

STLC (Software Testing Life Cycle)

Requirement Phase Testing

- Requirement Analysis in which test team studies the requirements from a testing point of view to identify testable requirements and the QA team may interact with various stakeholders to understand requirements in detail.
- Requirements could be either functional or non-functional.
- Automation feasibility for the testing project is also done in this stage.

STLC (Software Testing Life Cycle)

Activities in Requirement Phase Testing

- Identify types of tests to be performed.
- Gather details about testing priorities and focus.
- Prepare Requirement Traceability Matrix (RTM).
- Identify test environment details where testing is supposed to be carried out.
- Automation feasibility analysis (if required).

Deliverables of Requirement Phase Testing

- RTM
- Automation feasibility report. (if applicable)

STLC (Software Testing Life Cycle)

Test Planning in STLC is a phase in which a Senior QA management determines the test plan strategy along with efforts and cost estimates for the project.

Moreover, the resources, test environment, test limitations and the testing schedule are also determined.

The Test Plan gets prepared and finalized in the same phase.

Test Planning Activities

- Preparation of test plan/strategy document for various types of testing
- Test tool selection
- Test effort estimation
- Resource planning and determining roles and responsibilities.
- What will be tested/ What will not be tested
- Training requirement

Deliverables of Test Planning

- Test plan/strategy document.
- Effort estimation document.

STLC (Software Testing Life Cycle)

Test Case Development Phase

- It involves the creation, verification and modify test cases/test scripts after the test plan is ready.
- Then the QA team starts the development process of test cases for individual units/ modules / requirements/ user stories

Test Case Development Activities

- Create test cases, automation scripts (if applicable)
- Review and baseline test cases and scripts
- Create test data (If Test Environment is available)

Deliverables of Test Case Development

- Test cases/scripts
- Test data

STLC (Software Testing Life Cycle)

Test Environment Setup

- It decides the software and hardware conditions under which a work product is tested. It is one of the critical aspects of the testing process and can be done in parallel with the Test Case Development Phase.
- Test team may not be involved in this activity if the development team provides the test environment.
- The test team is required to do a readiness check (smoke testing) of the given environment.

Test Environment Setup Activities

- Understand the required architecture, environment set-up and prepare hardware and software requirement list for the Test Environment.
- Setup test Environment and test data
- Perform smoke test on the build

Deliverables of Test Environment Setup

- Environment ready with test data set up
- Smoke Test Results.

STLC (Software Testing Life Cycle)

Test Execution Phase

- It is carried out by the testers in which testing of the software build is done based on test plans and test cases prepared.
- The process consists of test script execution, test script maintenance and bug reporting.
- If bugs are reported, then it is reverted back to development team for correction and retesting will be performed.

Test Execution Activities

- Execute tests as per plan
- Document test results, and log defects for failed cases
- Map defects to test cases in RTM
- Retest the Defect fixes
- Track the defects to closure

Deliverables of Test Execution

- Completed RTM with the execution status
- Test cases updated with results
- Defect reports

STLC (Software Testing Life Cycle)

Test Closure:

- It is the final stage of the STLC where all testing-related activities are completed and documented.
- The main objective of the test closure stage is to ensure that all testing-related activities have been completed and that the software is ready for release.
- At the end of the test closure stage, the testing team should have a clear understanding of the software's quality and reliability, and any defects or issues that were identified during testing should have been resolved.
- The test closure stage also includes documenting the testing process and any lessons learned so that they can be used to improve future testing processes

Test summary report: A report is created that summarizes the overall testing process, including the number of test cases executed, the number of defects found, and the overall pass/fail rate.

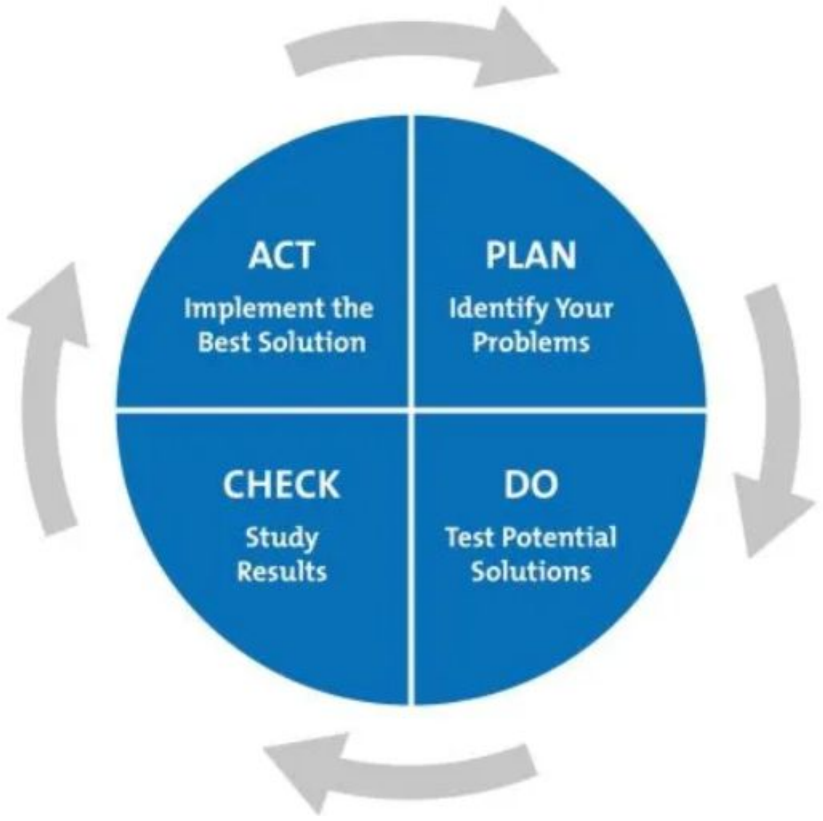
Defect tracking: All defects that were identified during testing are tracked and managed until they are resolved.

Knowledge transfer: Knowledge about the software and testing process is shared with the rest of the team and any stakeholders who may need to maintain or support the software in the future.

Feedback and improvements: Feedback from the testing process is collected and used to improve future testing processes

Test Planning, Monitoring & Controlling

✔ **Test Process:** The set of interrelated activities comprising of test planning, test monitoring and control, test analysis, test design, test implementation, test execution, and test completion.



Planning

1. Risk Analysis
2. Test Estimation
3. Test Planning
4. Test Organization

Execution

5. Test Monitoring and Control
6. Issue Management
7. Test Report and Evaluation

Test Planning, Monitoring & Controlling

Test Planning is the process of defining the **scope, approach, resources, and schedule** of intended testing activities.

It sets the foundation for successful test execution and helps ensure that testing is conducted effectively and efficiently.

Key Activities:

- **Define the Testing Scope**: Clearly identify what is to be tested (features, functionalities, interfaces, etc.) and what is not.
- **Determine Testing Objectives**: Specify the goals of testing, such as finding defects, verifying functionality, validating performance, etc.
- **Select Test Strategy**: Decide on the types of testing to be performed (manual vs. automated, functional, non-functional, performance, security, etc.).

Test Planning, Monitoring & Controlling

Key Activities:

- **Resource Planning**: Identify human resources (testers), tools, and environments needed for testing.
- **Test Environment Setup**: Plan for the configuration of test environments, including hardware, software, and network settings.
- **Risk Analysis and Mitigation**: Identify potential risks (such as lack of skilled resources, tight deadlines, etc.) and plan mitigation strategies.
- **Effort Estimation**: Estimate the time and resources needed for each testing phase.
- **Define Test Deliverables**: List all the documents and reports that need to be delivered, such as test cases, test scripts, test reports, and defect logs.

Test Risks

Test Planning Risks

Schedule Risk

- Time is not estimated perfectly
- Improper resource allocation
- Frequent project scope expansion

Budget Risk

- Wrong/Improper budget estimation
- Unexpected Project Scope expansion
- Mismanagement in budget handling
- Cost overruns

Technical Risks

- Frequent changes in requirement
- Less use of future technologies
- Less number of skilled employee
- High complexity in implementation
- Improper integration of modules

Testing Monitoring

Monitoring is the ongoing process of tracking and assessing testing activities to ensure they are aligned with the plan and objectives. It involves keeping an eye on the progress of testing efforts and making necessary adjustments.

Key Activities:

- **Track Test Progress**: Monitor the execution of test cases against the planned schedule.
  Defect Tracking: Keep track of defects identified during testing, including their status (open, in progress, closed, etc.).
  Metrics Collection: Collect testing metrics such as the number of test cases executed, passed, failed, blocked, defect density, and defect leakage.
  Regular Reporting: Provide regular status updates to stakeholders (daily, weekly, or bi-weekly reports) regarding test progress, issues, and risks.
- **Test Coverage Analysis**: Assess the coverage of test cases against requirements to ensure all functionalities are tested.

Testing Controlling

Controlling is the process of managing changes to the test plan and test activities to ensure testing stays on track and meets its objectives. This involves making adjustments based on the monitoring data.

Key Activities:

- **Analyze Deviations**: Identify reasons for deviations from the plan (e.g., delays, unexpected defects, resource constraints).
- **Implement Corrective Actions**: Take steps to address any deviations or issues (e.g., reallocating resources, adjusting schedules, or adding more testers).
- **Change Control**: Manage changes to the testing process, scope, or requirements. Ensure any changes are documented, reviewed, and approved by stakeholders.
- **Risk Management**: Continuously monitor risks and take steps to mitigate them as testing progresses.
- **Review and Update Test Plan**: Adjust the test plan and strategy based on ongoing findings and feedback.

Test Process

Indicators of Test Success

Test Pass Criteria:

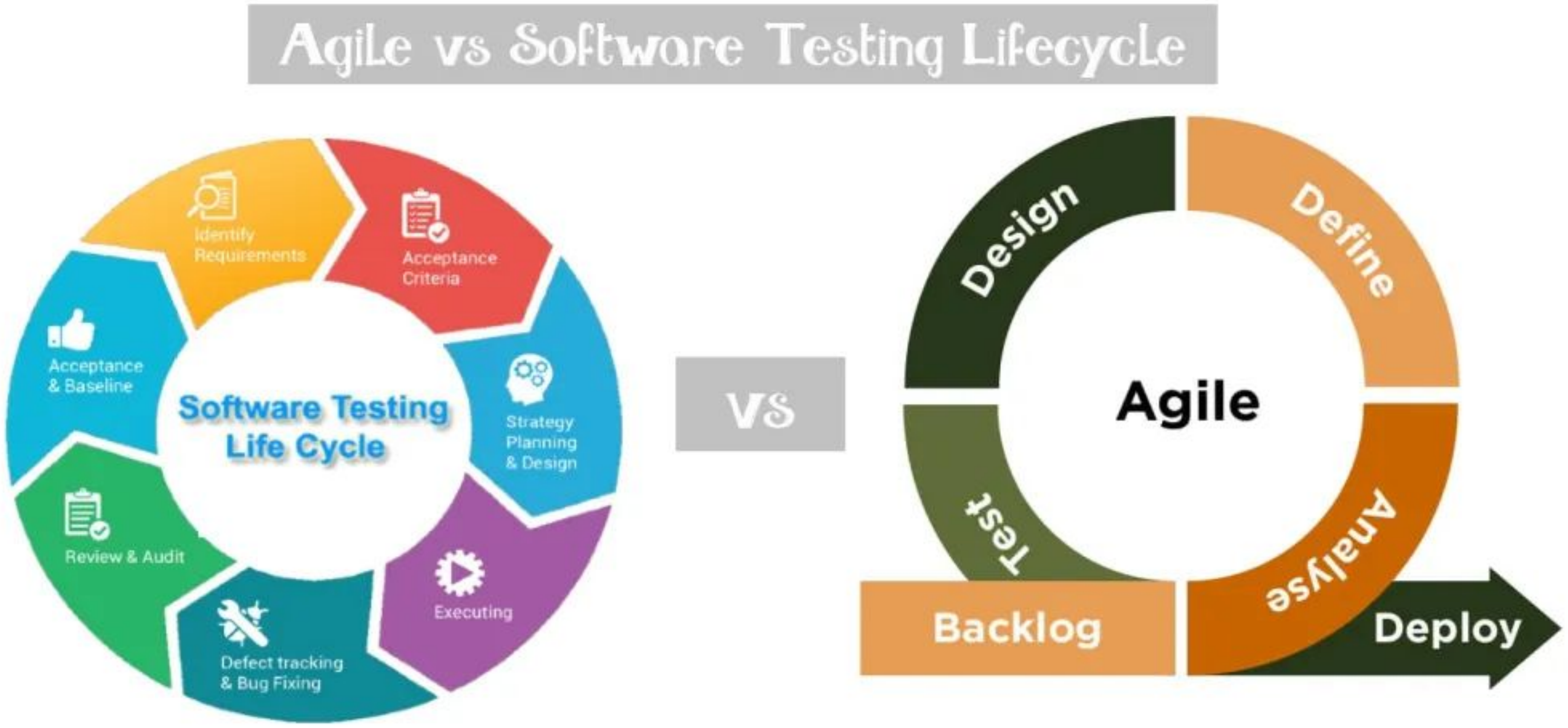
- 100 percent of test cases have been executed.
- Product/ Project meets customer requirements with respect to functionality, performance, and convergence around failures.
- No Severity 1 or 2 defects
- All Severity 3 defects (if any) have been documented/filed and workarounds have been provided.

Test Fail Criteria:

- Severity 1 defects are found with no workaround in an area critical to the solution.
- Severity 2 defects are found that can put in jeopardy the on-time deployments of the solution.
- One or more "key" features are missing, are not operational, or don't meet customer-specific requirements.

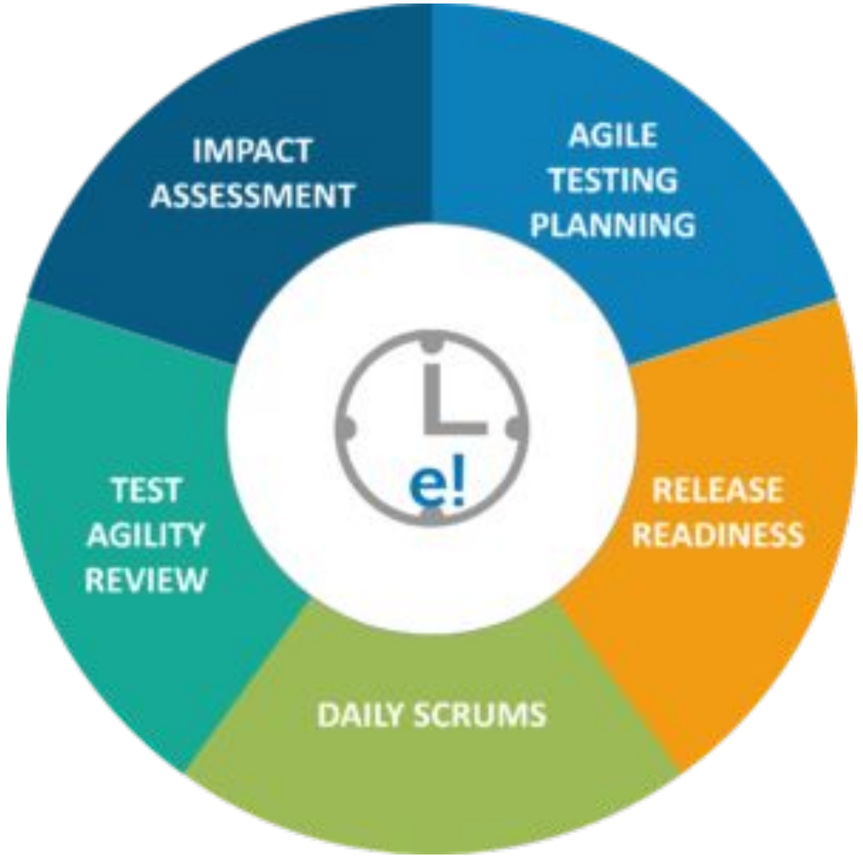
Test Process

Agile Testing Life Cycle Vs Software Testing Life Cycle



Test Process

Agile Testing Life Cycle



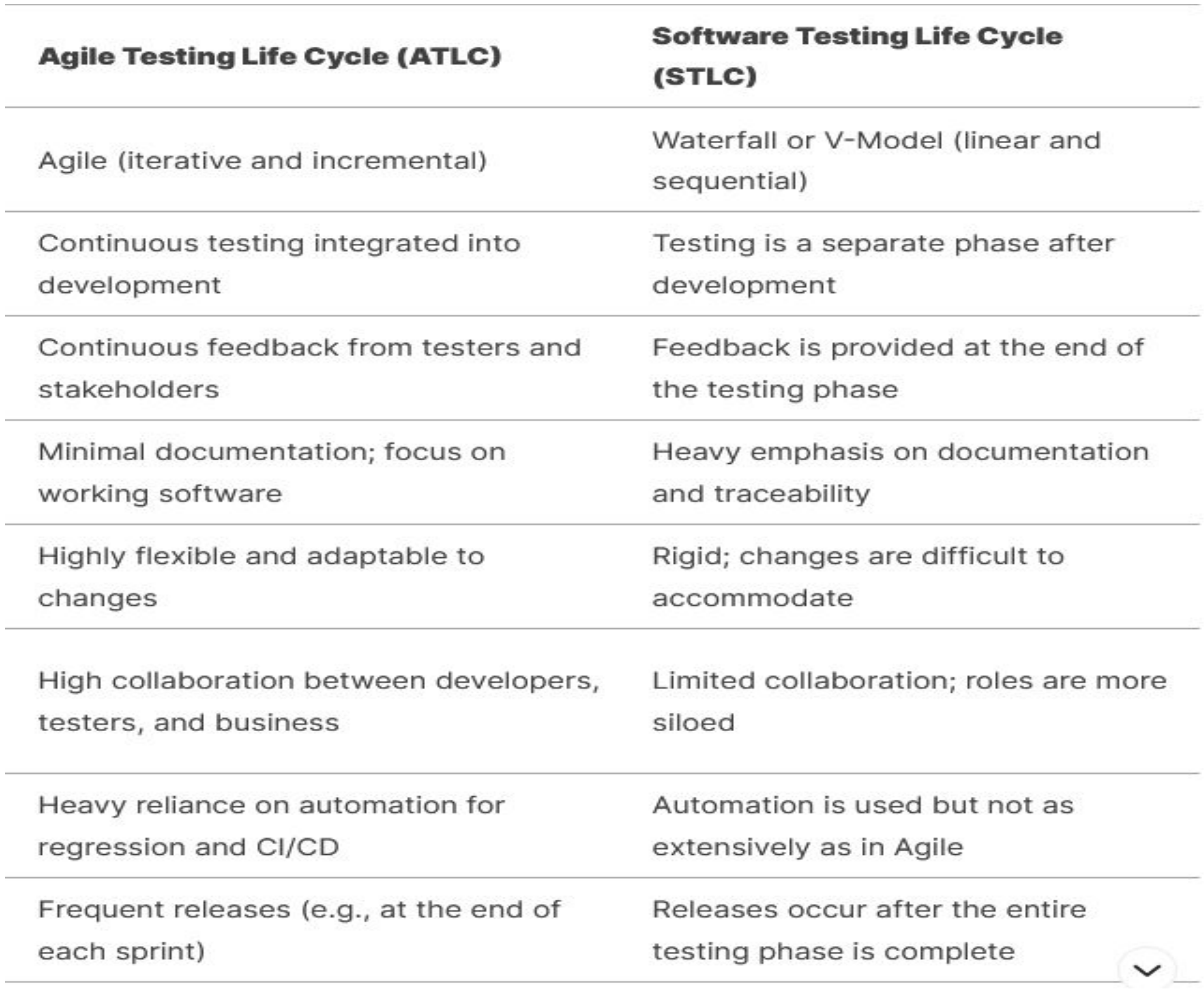
Agile Testing Life Cycle (ATLC)

- **Test Planning**:
  - Occurs at the beginning of each **iteration (sprint).**
  - Focuses on the user stories and features to be delivered in the current sprint.
- **Test Design**:
  - Test cases are created based on **user stories** and **acceptance criteria**.
  - Emphasis on collaboration between developers, testers, and business stakeholders.
- **Test Execution**:
  - Testing is performed continuously throughout the sprint.
  - Includes unit testing, integration testing, and acceptance testing.
- **Test Feedback**:
  - Continuous feedback is provided to the development team.
  - Defects are fixed quickly, and retesting is done in the same iteration.

Agile Testing Life Cycle (ATLC)

- **Regression Testing**:
  - Ensures that new changes do not break existing functionality.
  - Automated regression tests are often used to save time.
- **Release**:
  - The tested features are delivered to the customer at the end of the sprint.
- **Retrospective**:
  - The team reflects on the testing process and identifies areas for improvement.

Test Process



Agile Testing Life Cycle (ATLC)	Software Testing Life Cycle (STLC)
Agile (iterative and incremental)	Waterfall or V-Model (linear and sequential)
Continuous testing integrated into development	Testing is a separate phase after development
Continuous feedback from testers and stakeholders	Feedback is provided at the end of the testing phase
Minimal documentation; focus on working software	Heavy emphasis on documentation and traceability
Highly flexible and adaptable to changes	Rigid; changes are difficult to accommodate
High collaboration between developers, testers, and business	Limited collaboration; roles are more siloed
Heavy reliance on automation for regression and CI/CD	Automation is used but not as extensively as in Agile
Frequent releases (e.g., at the end of each sprint)	Releases occur after the entire testing phase is complete