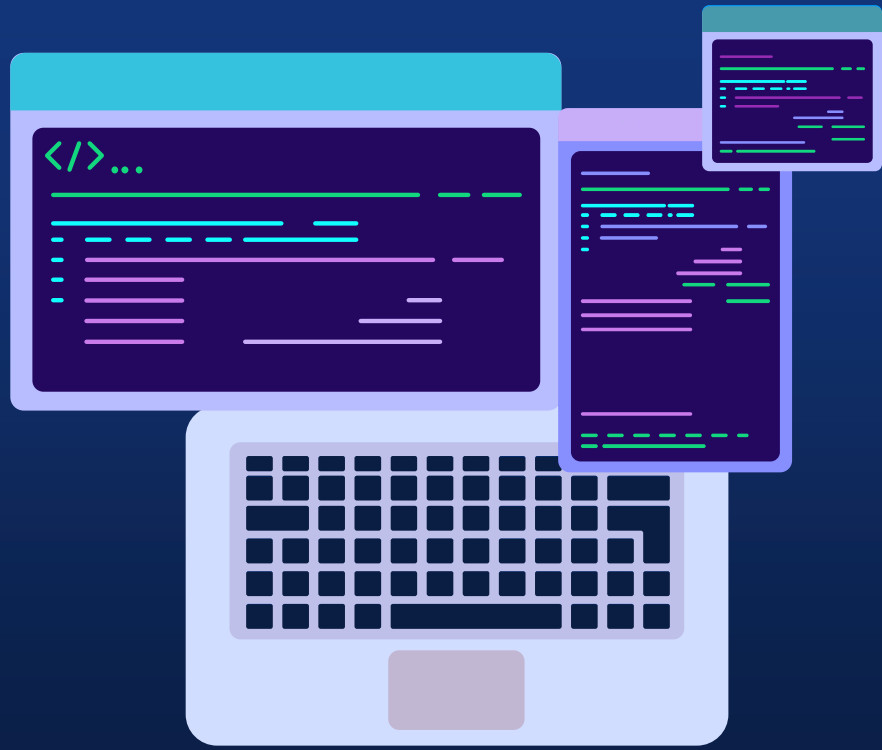
# Software Testing & It's Importance In any App

Prepared by:
Jawad Srour



# TABLE OF CONTENTS


01 Intro to Software Testing

02 Types of Software Tests

03 Dangers of not Testing

04 Best Practices

05 Tools & Technologies

06 Conclusion

“Quality is never an accident. It is always the result of intelligent effort..”

—John Ruskin

# 01

# Introduction to Software Testing



# Introduction to Software Testing

- It is the process used to identify the correctness, completeness and quality of developed software

- It is the process of executing a program/application under positive and negative conditions by manual or automated means. It checks for the:

1. Specification
2. Functionality
3. Performance



# Introduction to Software Testing

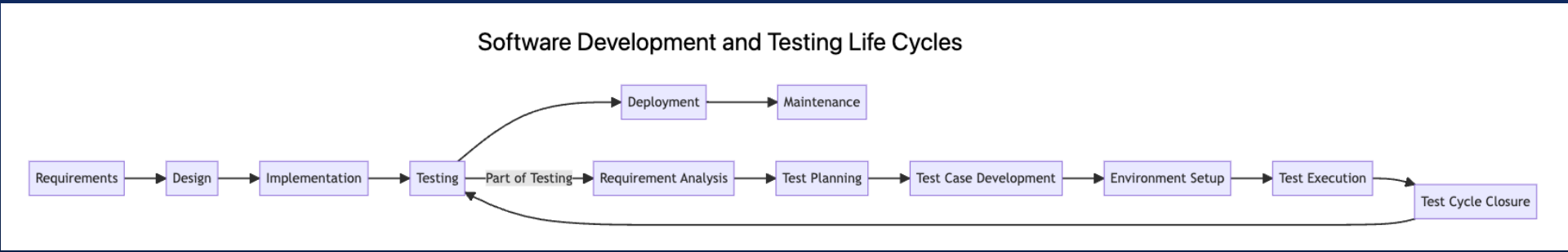
- Why do we Test?

1. Ensure Correctness
2. Prevent Future Regressions
3. Improve Code Quality
4. Facilitate Refactoring
5. Some sort of documentation
6. Detect integration issues
7. Build confidence

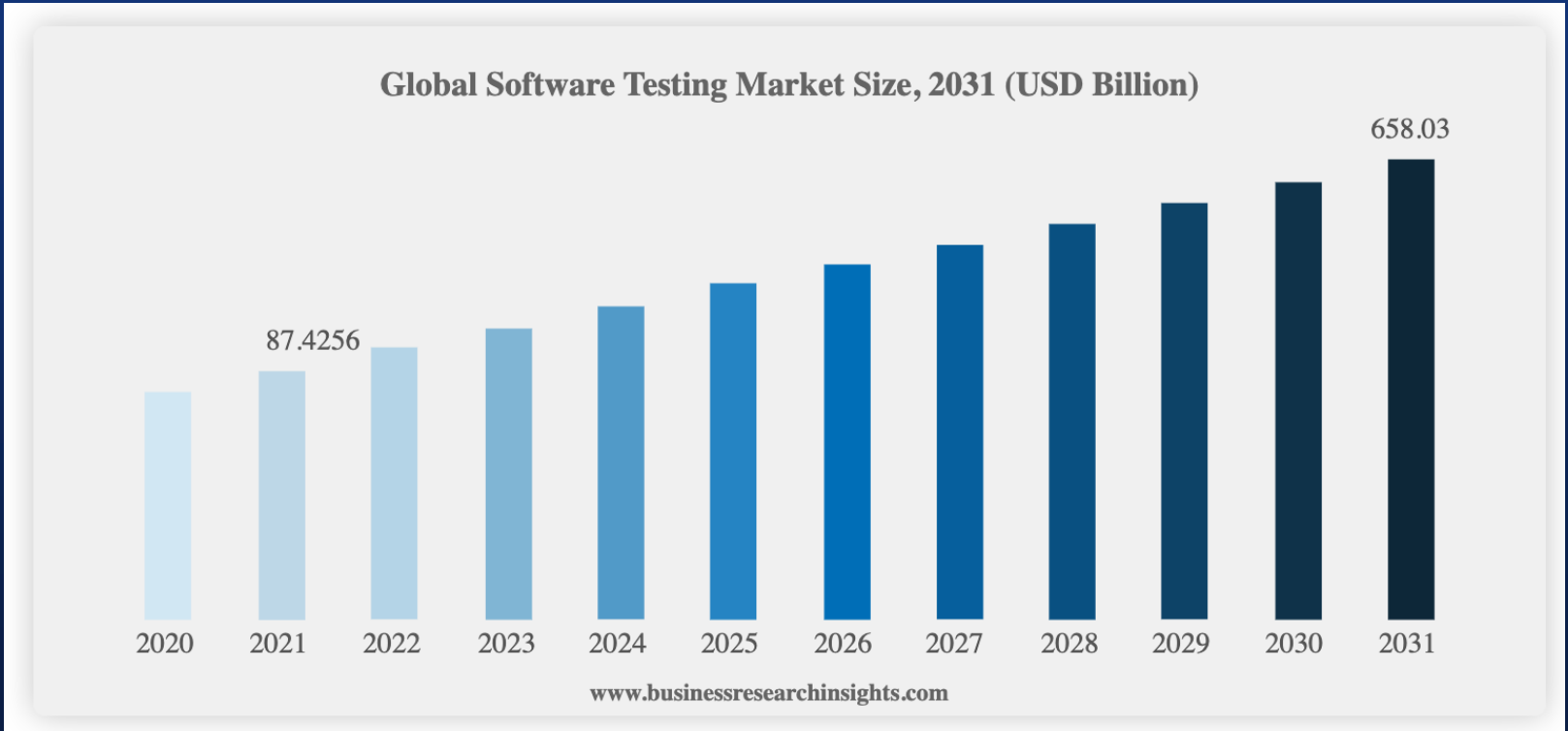


# Introduction to Software Testing (SDLC and Testing Life Cycle)

- Standard model used world wide to develop software.
- Necessary to ensure the quality of software.
- Logical steps taken to develop a software product



# Introduction to Software Testing (Market Growth)



# 02

# Types of Software Testing

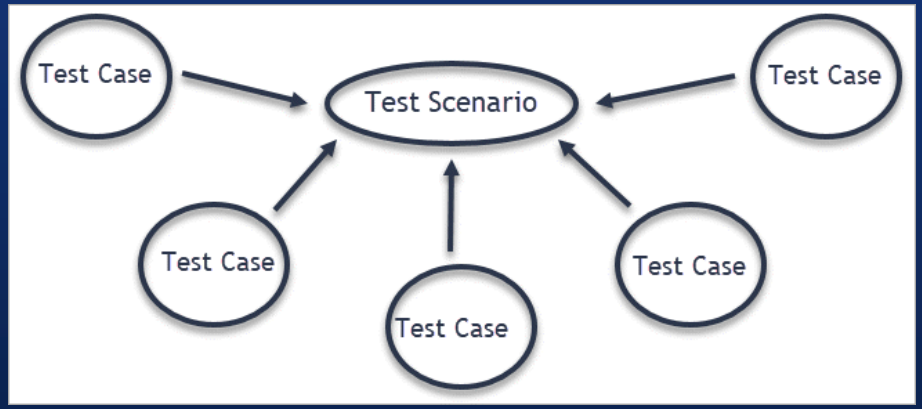


# Types of Software Testing (Test Case)

- Test case: is a specific procedure of testing a particular requirement

- It will include:

1. Identification of specific requirement tested
2. Test case success/failure criteria
3. Specific steps to execute the test
4. Test data



# Types of Software Testing (Functional vs Non Functional Testing)



## Functional Testing

### User View

- Can the user do this ?
- Does the feature work?
- Does the software meets the users expectations ?



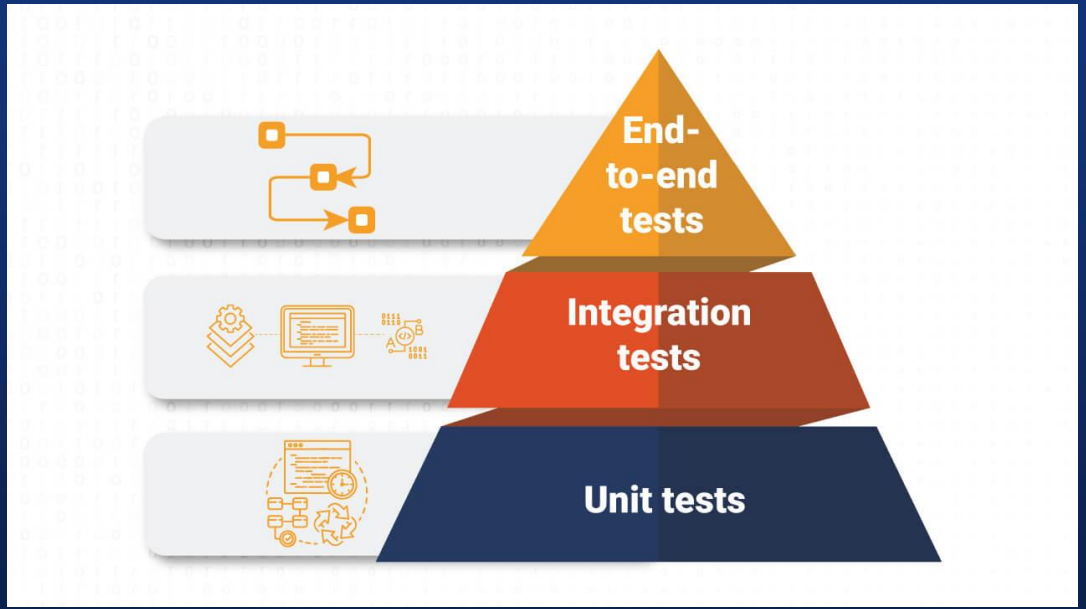
## Non Functional Testing

### Hacker View

- Are vulnerabilities in the network infrastructure , operating system or database system ?
- Can the client (browser etc.) be manipulated ?
- Is the server code robust?

# Types of Software Testing (the Pyramid)

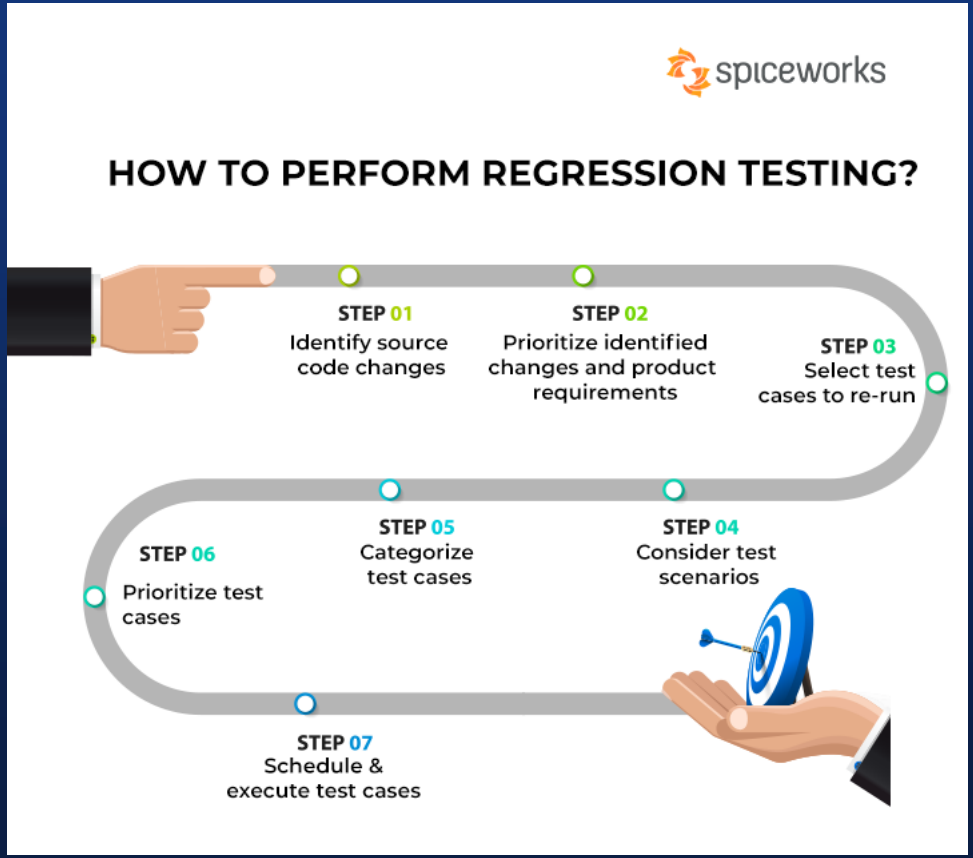
1. Unit Tests: Testing individual components of code in isolation
2. Integration Tests: Testing the combination of two or more units or components
3. End-to-End Tests: Testing the complete flow of an application from start to finish.



# Types of Software Testing (Regression Testing)

- Regression Testing: Ensuring new code changes don’t negatively impact previously tested components.
- New requirements implemented = Regression testing required.

- Sometimes a test failure upon changes might be a positive impact, and not a regression.

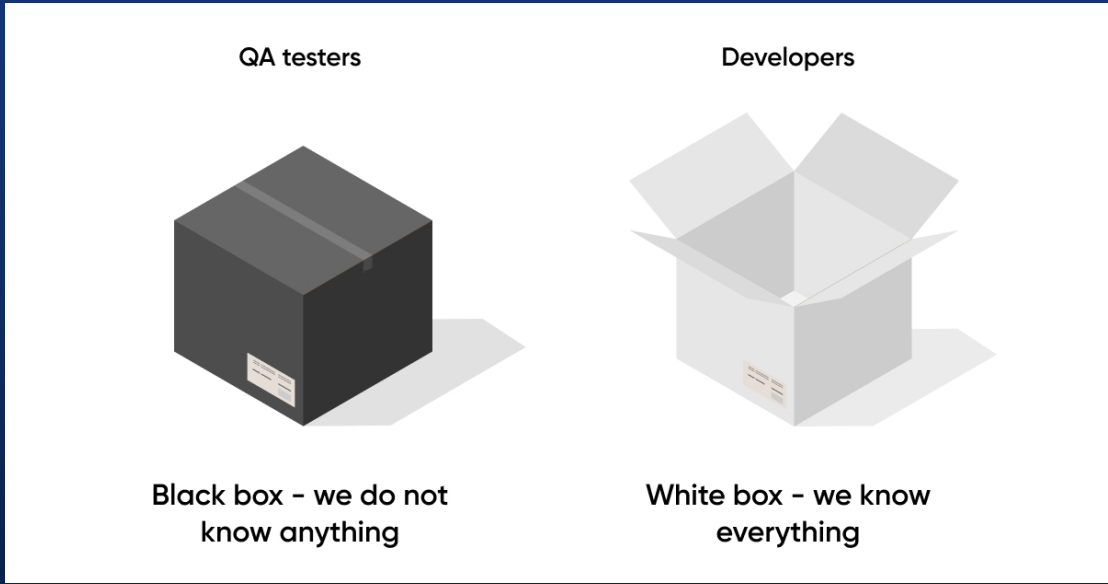


# Types of Software Testing (Whitebox vs Blackbox)

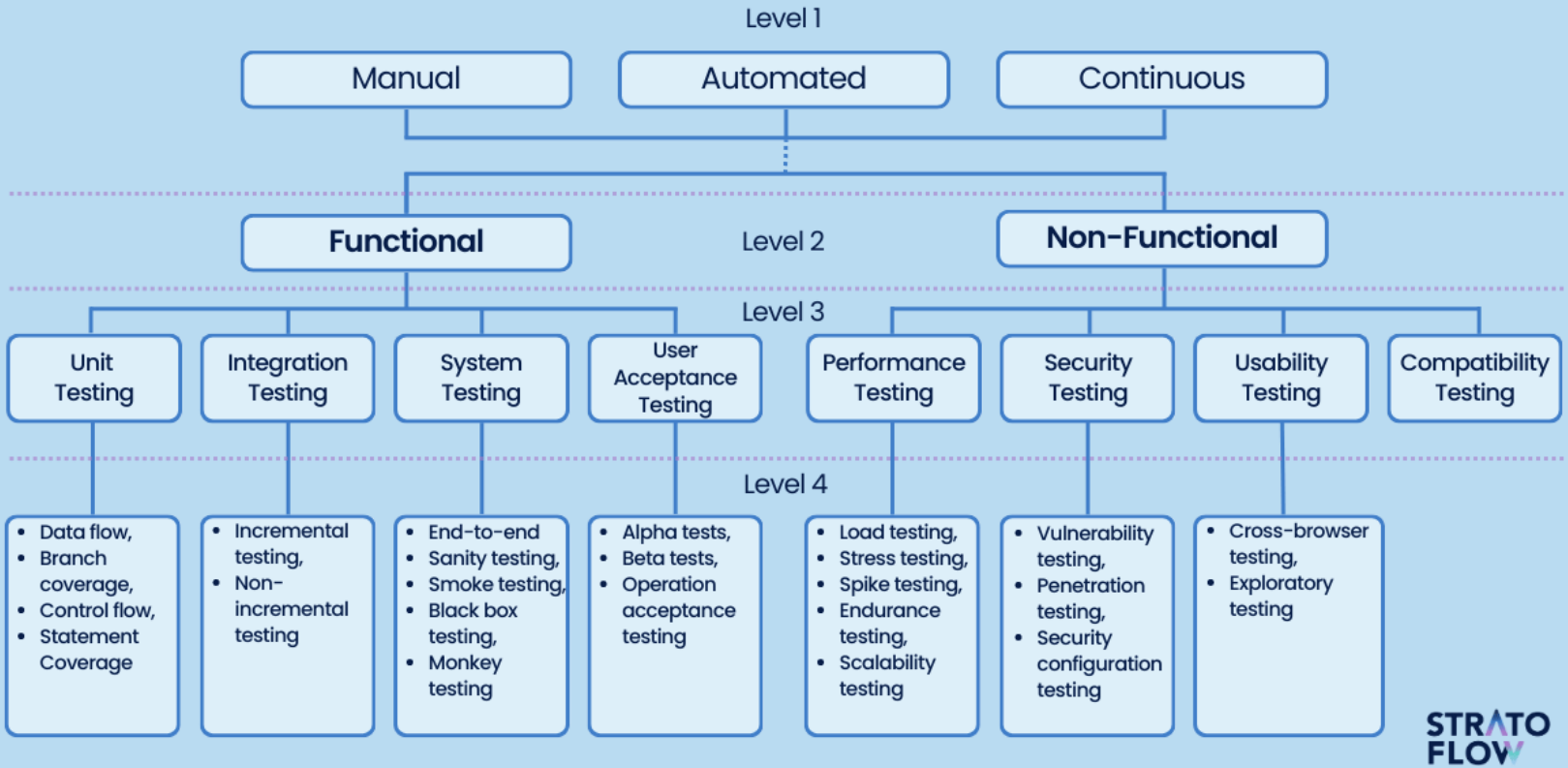
1. Black-box testing: focuses on the functionality of the app. (I/O)
2. White-box testing: testing the internal structures of the application.

Who does what?

- Black-box testing is typically done by QA engineers.
- White-box testing is done by software engineers.

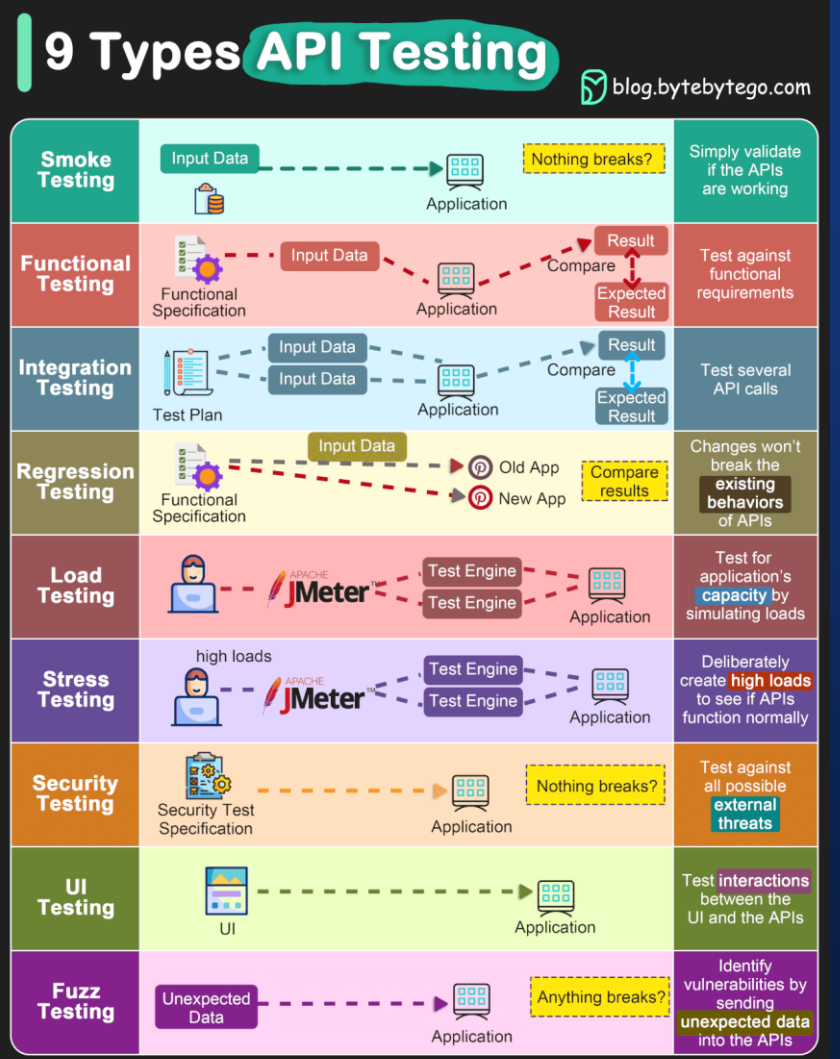


# Types of Software Testing



# Types of Software Testing

## API Testing



# 03

## Dangers of Not Testing Code



# Dangers of Not Testing Code (Historical Events)

- American Airlines FNS, 1995 crash
- 151 passengers killed out of 155 total and the entire crew.
- Proper software testing could’ve saved the pilots catching their mistakes.



# Dangers of Not Testing Code (Historical Events)

- Six people killed by radiation overdose.
- Old software was deployed on new hardware without testing…
- A flag overflow problem caused reset of some variables to zero, bypassing safety measures.



# Dangers of Not Testing Code (Historical Events)

- Ariane 5 rocket failure.
- Loss of guidance and attitude information on launch.
- Inertial Reference System software bug.
- Casting a 64 bit floating point and passing it to a 16 bit integer.



# Dangers of Not Testing Code (Historical Events)

- Ariane 5 rocket failure.
- Loss of guidance and attitude information on launch.
- Inertial Reference System software bug.
- Casting a 64 bit floating point and passing it to a 16 bit integer.



# 370,000,000$

Loss due to Ariane-5 Crash in June 4, 1996

# 04

## Best Practices



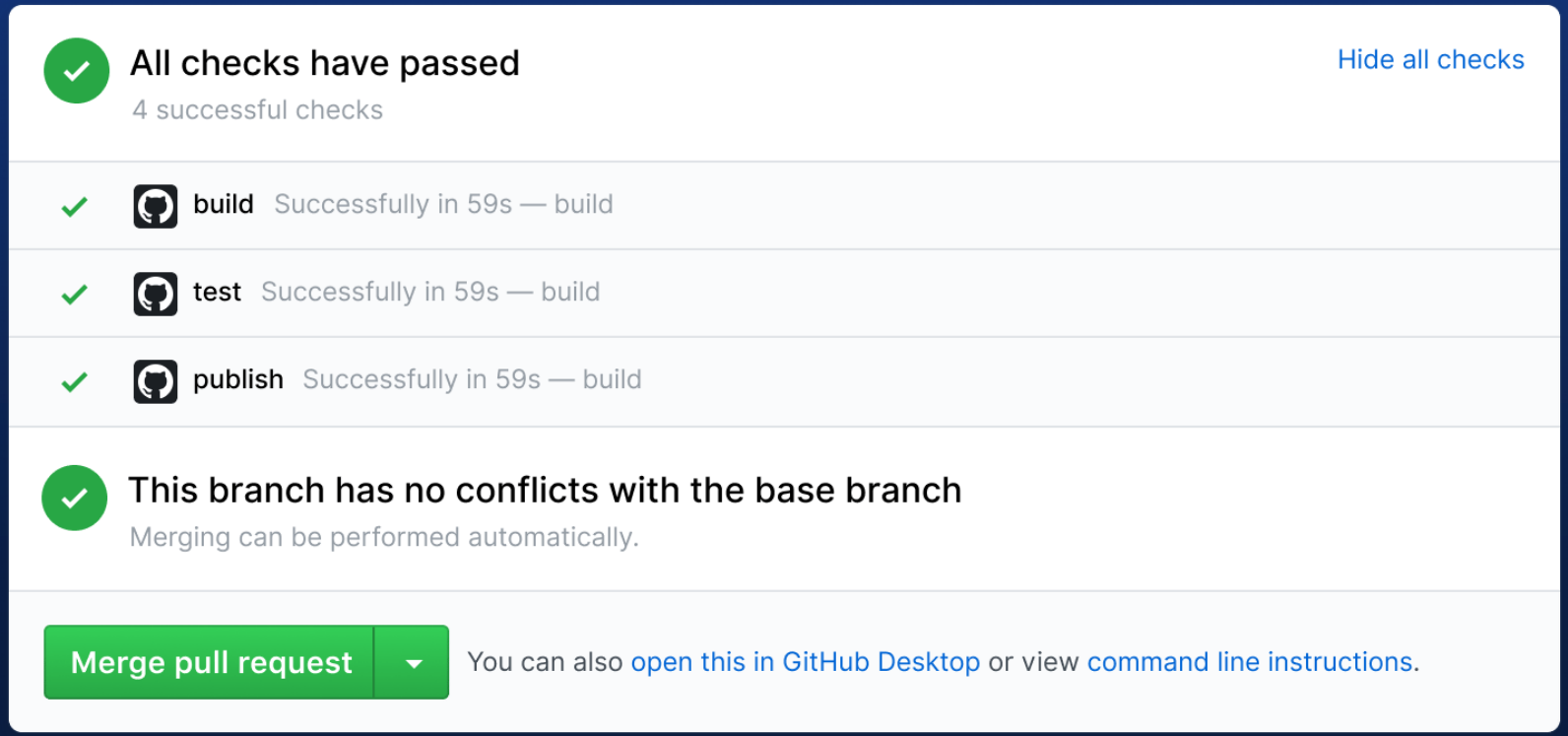
# Best Practices

1. Aim for High Percentage of Test Coverage

| | |
|---|---|
| **Covered lines:** | 171 |
| **Uncovered lines:** | 1 |
| **Coverable lines:** | 172 |
| **Total lines:** | 738 |
| **Line coverage:** | 99.4% (171 of 172) |
| **Covered branches:** | 22 |
| **Total branches:** | 22 |
| **Branch coverage:** | 100% (22 of 22) |

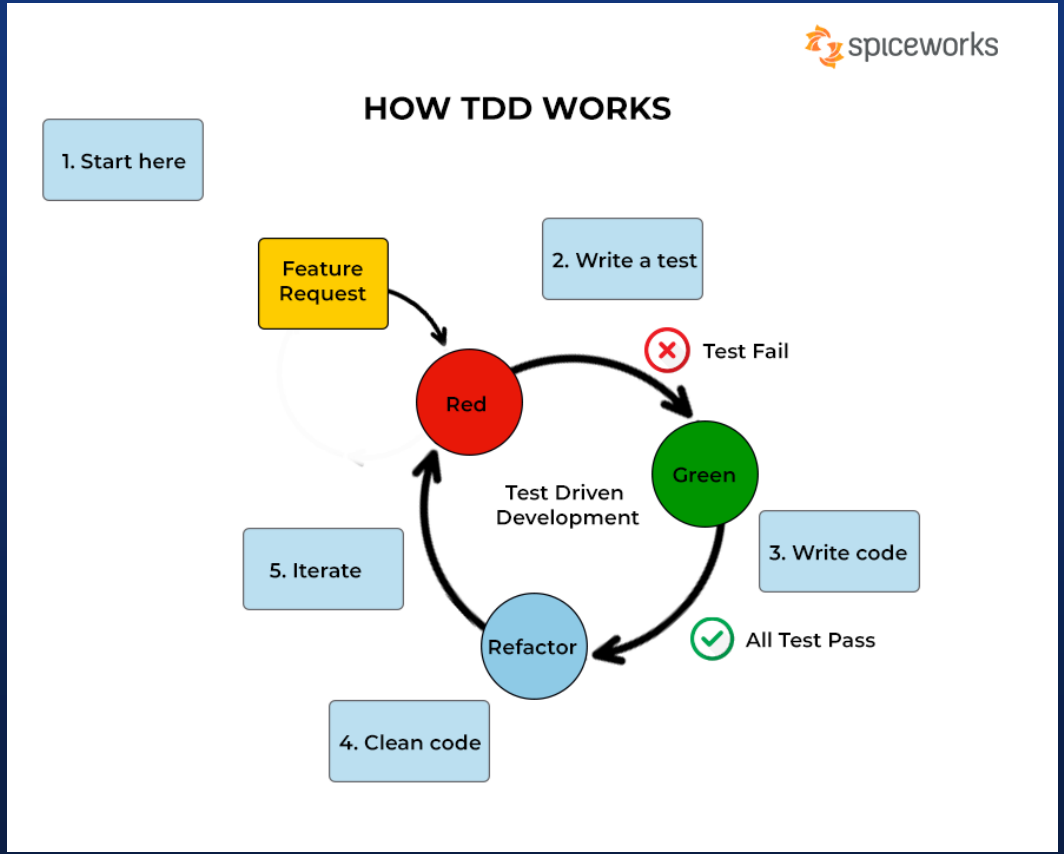
# Best Practices

2. Automate tests in CI/CD pipelines



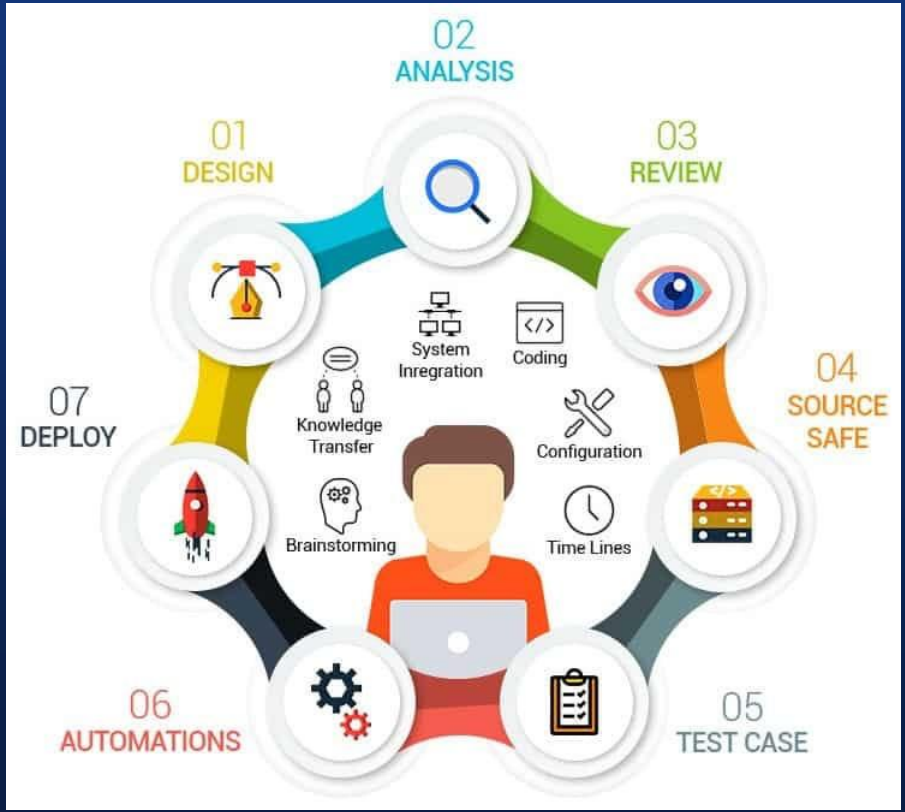
# Best Practices

3. Test Driven Development (TDD)



# Best Practices

4. Prioritize Tests
5. Start testing early on.
6. Developers must avoid “Wishful Testing”.



# 05

# Tools & Technologies



# Tools and Technologies

1. Selenium: automated testing framework for web apps.
2. Git-hub actions: create your CI pipelines and add tests to them!
3. Mocha: JavaScript test framework running on Node.JS
4. Jest: Popular option for testing JavaScript and React apps
5. Cypress: End-to-End testing framework in JavaScript
6. PyTest: Python framework for writing unit and integration tests.
7. Junit: Good choice for unit tests in Java
8. Nunit: popular option for unit testing .NET apps.

# Getting our Hands Dirty

# Q&A

Thanks!

Linked-in Blog:

