MODULE - 1 GLIMPS - Phases of Software project - Quality Assurance, Quality control - Testing, Verification and Validation - Process Model to represent Different Phases - Life Cycle models. White-Box Testing: Static Testing - Structural Testing Challenges in White-Box Testing.

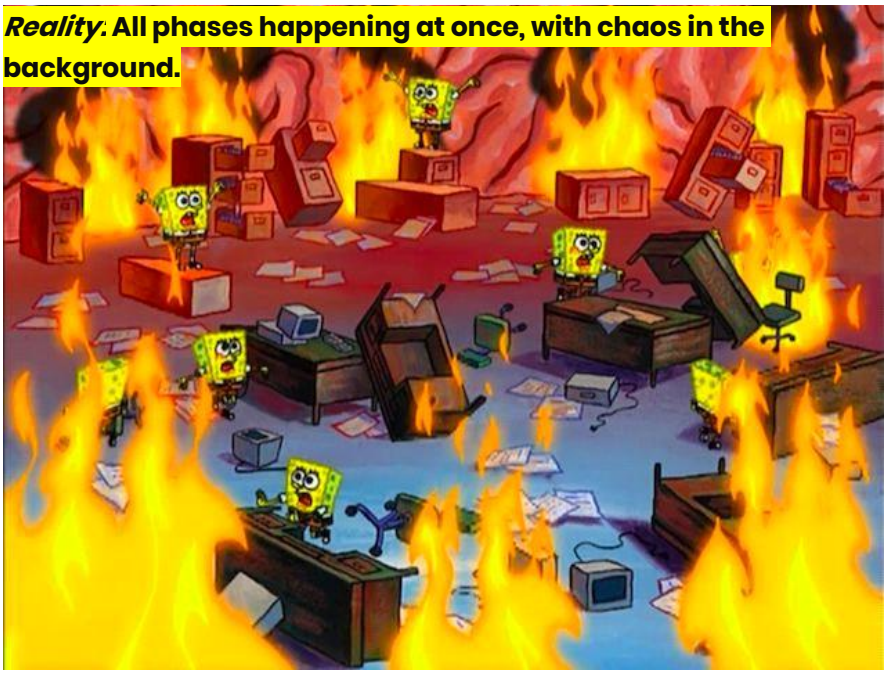
12 February 2025 21:00

https://github.com/akashksankar/SOFTWARETESTING

1. 🧱 Phases of a Software Project

Expectation: **Clean, sequential phases: Requirements → Design → Coding → Testing → Deployment.**

Reality: **All phases happening at once, with chaos in the background.**



Meme: *Expectation vs. Reality (SpongeBob in chaos)*

Study Explanation:
The software project phases are typically listed as sequential: Requirements, Design, Implementation, Testing, Deployment, and Maintenance. However, in real-world projects, these phases often overlap, loop back, or get rushed, leading to a chaotic workflow. This meme sets the tone for understanding that while we teach structured models, practical software development is often nonlinear.

Context

◇ 1. Phases of a Software Project

Q: Explain the different phases of a software project.
A:
A software project typically goes through the following phases:

1. **Requirement Analysis** – Understanding what the user needs
2. **Design** – Planning how the system will work

3. **Implementation (Coding)** – Writing the actual code
4. **Testing** – Checking the system for errors
5. **Deployment** – Delivering it to the user
6. **Maintenance** – Updating and fixing issues after release
   These phases often overlap and may repeat in real-world projects.

2. ☑ Quality Assurance (QA) vs Quality Control (QC)

Study Explanation:

Quality Control (QC) focuses on identifying defects in the final product, while Quality Assurance (QA) ensures processes are in place to prevent defects from occurring in the first place. QA is proactive, QC is reactive. The meme helps distinguish the two roles humorously but clearly.

Q: Differentiate between Quality Assurance and Quality Control.
A:

- **Quality Assurance (QA):** Ensures the **process** used to develop the product is effective to prevent defects. It's proactive.
- **Quality Control (QC):** Involves checking the **final product** to identify defects. It's reactive.
  QA focuses on building quality into the process, while QC finds issues in

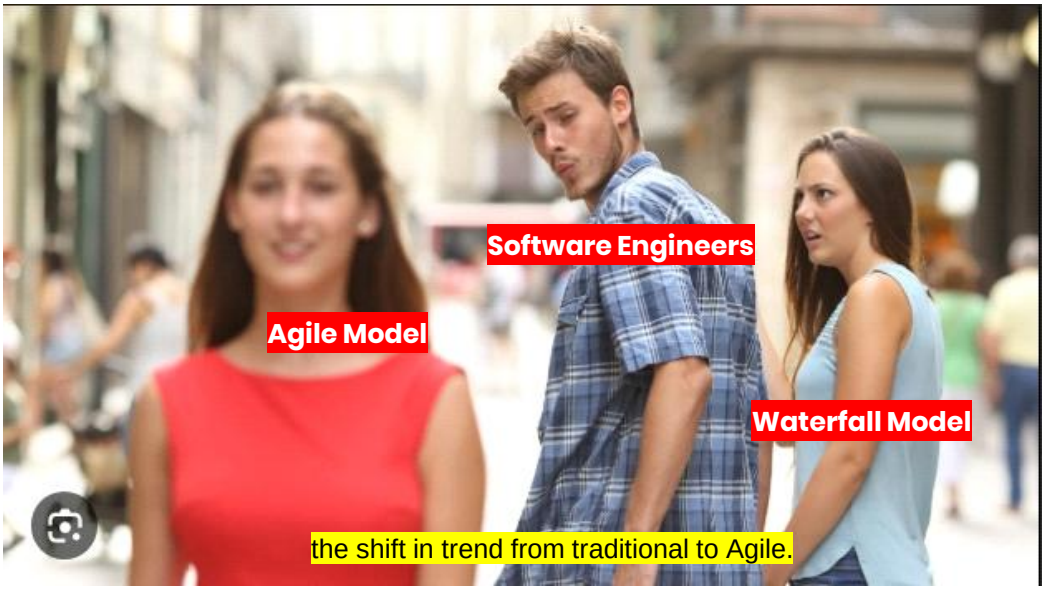
the finished product.

3. ◇ Verification, Validation, and Testing

Q: Define and differentiate Verification, Validation, and Testing.
A:

- **Verification:** Are we building the product **right**? It checks if the design and requirements are properly implemented.
- **Validation:** Are we building the **right** product? It ensures the final product meets the user's needs.
- **Testing:** It is the actual process of executing the program to find bugs or errors.

4. 🌀 Process Model to Represent Different Phases



Process models help plan software development. Traditional models like Waterfall follow a linear path, while Agile is iterative, focusing on collaboration and customer feedback. The meme reflects the shift in industry preference toward Agile methodologies for flexibility and responsiveness.

Q: What are software development process models? Explain with examples.
A:
Process models are structured approaches that guide how software is developed. Common models include:

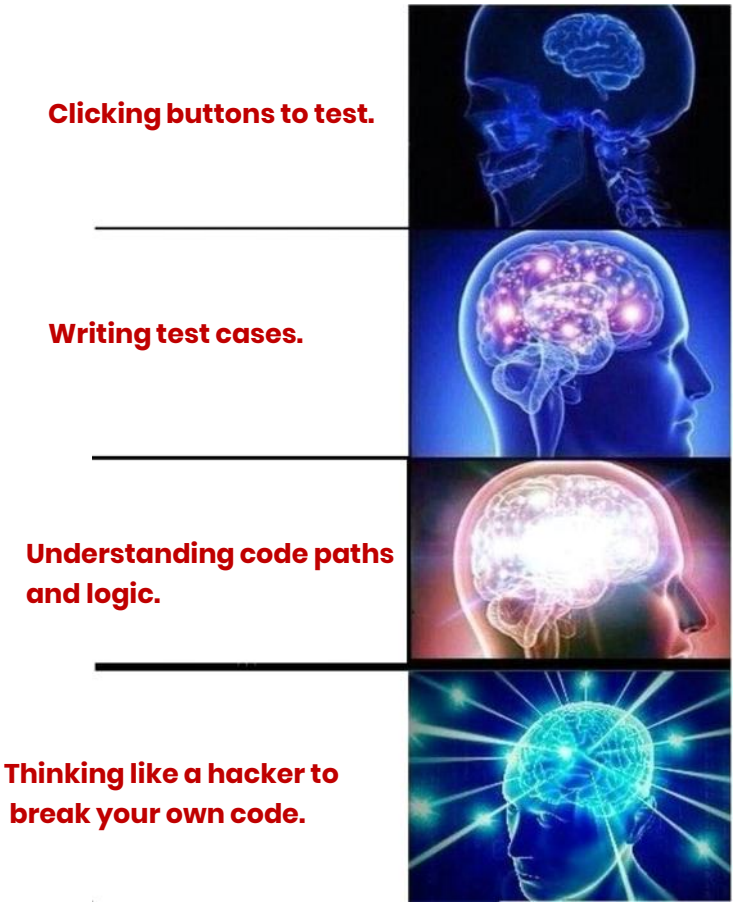
- **Waterfall Model:** Linear and sequential phases
- **Agile Model:** Iterative and customer-feedback-based development
- **Spiral Model:** Emphasizes risk analysis and iterative progress
- **V-Model:** Verification and validation go hand-in-hand
  Each model fits different types of projects based on complexity and client needs.

5 . ♻ Life Cycle Models

Q: What are Software Life Cycle Models?
A:
Software Life Cycle Models describe the stages a software goes through from conception to retirement.
Examples include:

- **Waterfall Model** – Simple, phase-based
- **Spiral Model** – Iterative, with risk handling
- **Agile Model** – Fast and adaptive
- **V-Model** – Every development phase is tested
  These models help teams plan and manage the development effectively.

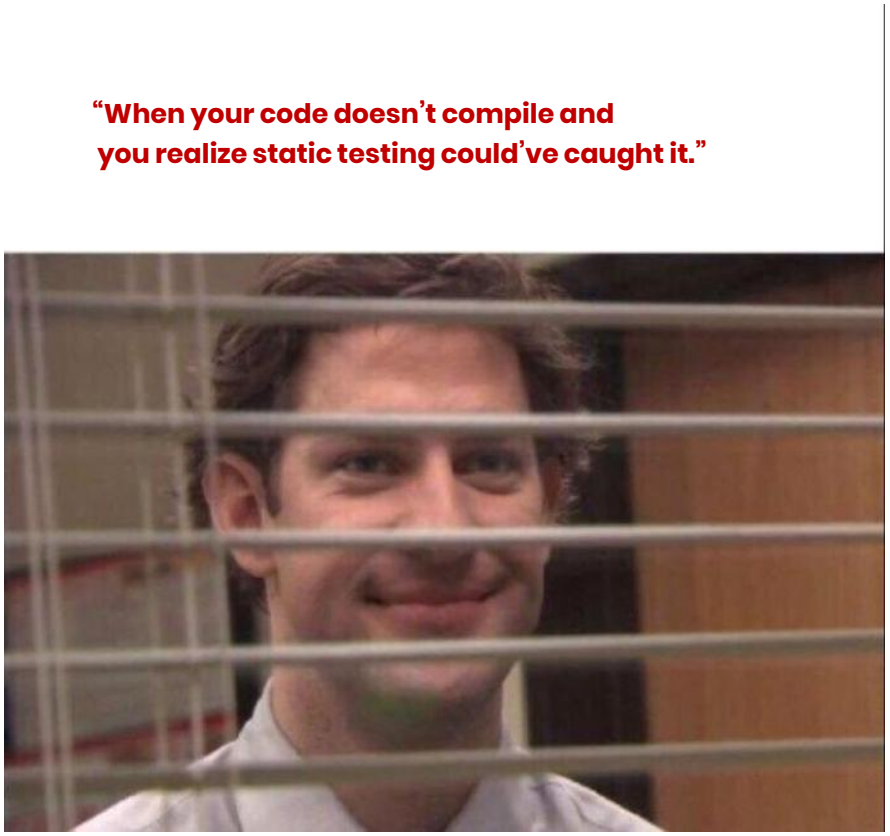
6◯ White-Box Testing



Q: What is White-box Testing?
A:
White-box testing is a testing method where the internal structure or logic of the code is known to the tester. The tester examines the **flow of inputs and outputs** and **logic paths**. Techniques include:

- Path testing
- Loop testing
- Condition testing
  This method helps to optimize the code and catch hidden bugs.

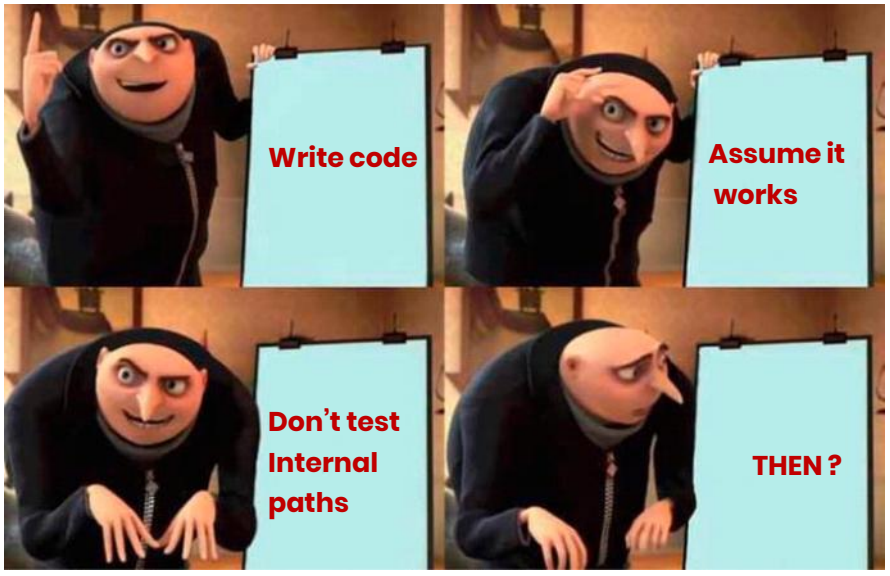
7. 🔍 Static Testing



Q: What is Static Testing? Give examples.
A:
Static Testing is done **without executing** the code. It involves reviewing the code, documents, and design to detect errors early.
Examples include:

- **Code Reviews**
- **Walkthroughs**
- **Inspections**
  It helps in catching bugs before the software even runs, saving time and cost.

8. 🔬 Structural Testing



Q: Explain Structural Testing.
A:
Structural Testing is another name for **White-box Testing**. It tests the structure of the software code by analyzing paths, decisions, and loops.
It ensures that **every part of the code** is tested thoroughly. This requires knowledge of programming and internal logic.

9. 😵 Challenges in White-Box Testing

"When you have 100% code coverage but still miss edge cases."

Q: What are the major challenges in White-box Testing?
A:

- High **code complexity** makes full coverage difficult
- Requires deep **programming knowledge**
- Can miss **external behavior issues**
- Difficult to cover **all logical paths**
- Takes more **time and effort**
  Despite its depth, white-box testing may not always detect all runtime or UI issues.

https://github.com/akashksankar/SOFTWARETESTING