Dhina karan

@dhinakaran

Regression testing

Dhina karan

@dhinakaran

Regression testing **checks if recent code changes maintain existing software functionality**. It's a **black-box technique ensuring new features or fixes don't introduce bugs**. Typically automated, it verifies that code still works after significant changes, saving time compared to manual testing. This **process is vital for software stability and reliability.**

Swipe next →

Dhina karan

@dhinakaran

Example of Regression testing

Adding a new payment method to an e-commerce site requires regression testing. This involves checking if basic functions like adding items to the cart and making a purchase still function correctly. Automated tests are often used to quickly ensure the changes don't impact existing features.

Swipe next →

Dhina karan

@dhinakaran

Performing regression testing involves:

1. **Selecting test cases:** Choose tests covering critical functions.
2. **Creating test suites**: Organize tests for efficiency.
3. **Automating where possible:** Speed up testing with automation.
4. **Executing tests:** Run both new and existing test cases.
5. **Comparing results:** Check for discrepancies from previous tests.
6. **Investigating failures:** Identify and address any issues.

Swipe next →

Dhina karan

@dhinakaran

7.**Updating test cases:** Modify tests to reflect changes.

8.**Re-running tests:** Confirm issue resolution and overall functionality.

9.**Repeat regularly**: Conduct regression testing after changes to ensure ongoing software stability.

Swipe next →

Dhina karan

@dhinakaran

Manual regression testing involves:

- **Selecting relevant test cases:** Pick tests affected by recent changes.
- **Organizing into test suites**: Group tests for efficient execution.
- **Executing tests manually**: Follow steps and verify outcomes.
- **Comparing results**: Check for differences from previous tests.
- **Investigating failures**: Identify and address issues.

Swipe next →

Dhina karan

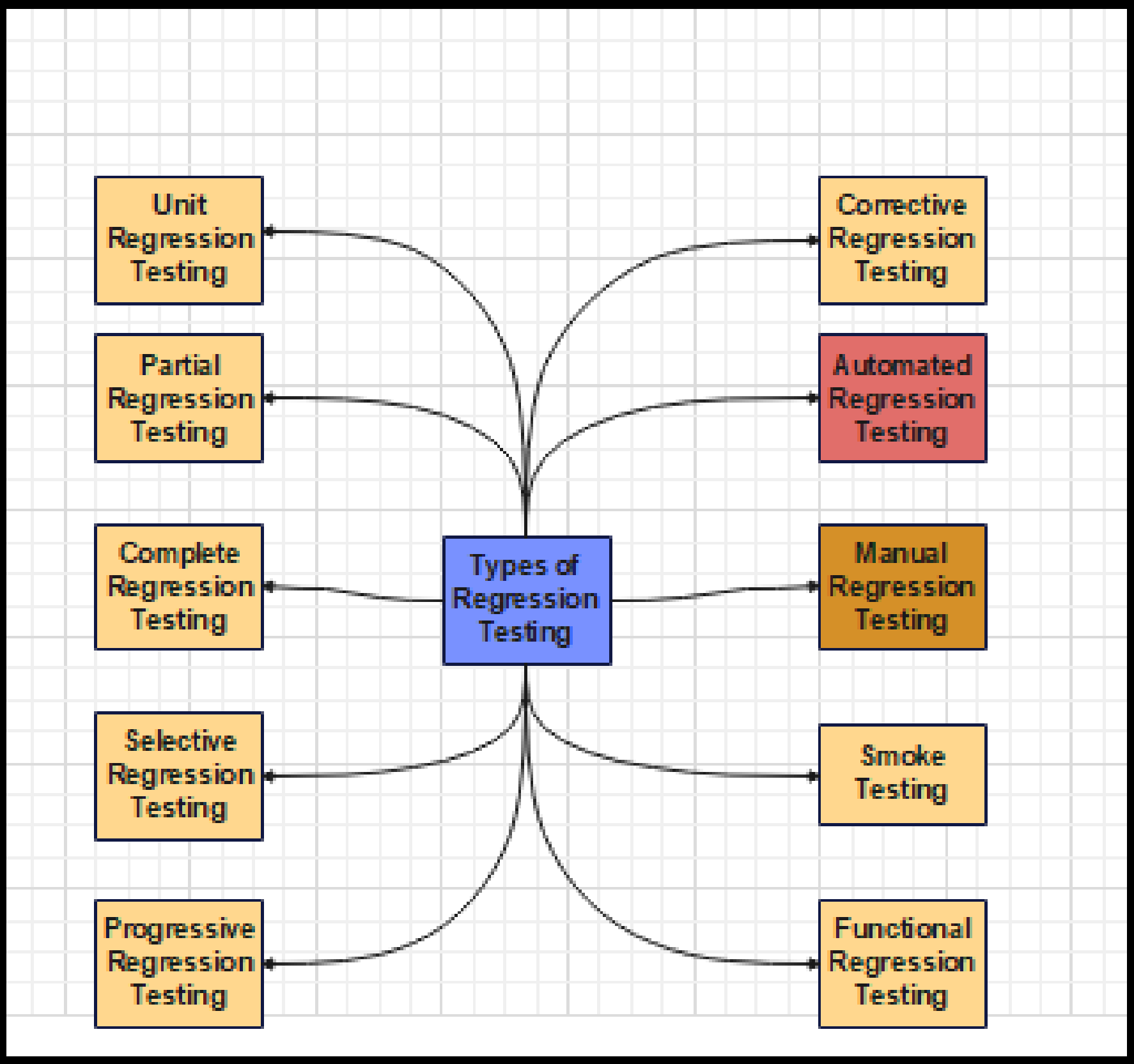
@dhinakaran

- **Updating test cases**: Modify as needed to reflect changes.
- **Re-running tests**: Confirm issue resolution and overall functionality.
- **Repeat regularly:** Perform manual regression testing after changes to maintain software stability

Swipe next →

Dhina karan

@dhinakaran



Swipe next ⟶

Dhina karan

@dhinakaran

- **Test Case Selection**: Identify and select test cases that cover critical functionalities and are likely to be affected by recent changes.
- **Test Suite Organization**: Organize the chosen test cases into logical test suites to facilitate efficient execution.
- **Automation (Optional)**: If possible, automate selected test cases to speed up testing and enhance repeatability, especially for repetitive scenarios.
- **Execution**: Run the selected test suites, either manually or using automated testing tools, to verify the functionality of the software.

Swipe next →

Dhina karan

@dhinakaran

- **Results Comparison**: Compare the current test results with baseline results from previous testing cycles to identify any discrepancies or failures.
- **Issue Investigation**: If there are failures, investigate the root cause to determine whether they result from recent changes or pre-existing issues.
- **Test Case Update (if necessary)**: Modify test cases to reflect changes in the application, such as updating test data, expected outcomes, or test steps.

Swipe next ⟶

Dhina karan

@dhinakaran

- **Re-execution:** After addressing any issues and updating test cases, re-run the test suites to confirm that problems have been resolved and that the overall application functionality remains intact.
- **Regular Iteration:** Perform regression testing regularly, especially after significant code changes, integrations, or before releasing a new version, to maintain ongoing software stability.

Swipe next →

FOLLOW ME !

FOR MORE

Dhina karan