AUTOMATED SOFTWARE TESTING

Automatic Testing

- Test automation is the practice of running tests automatically, managing test data, and utilizing results to improve software quality.
- From business analysts to developers and DevOps engineers, getting the most out of test automation takes the inclusion of everyone.

Automatic Testing

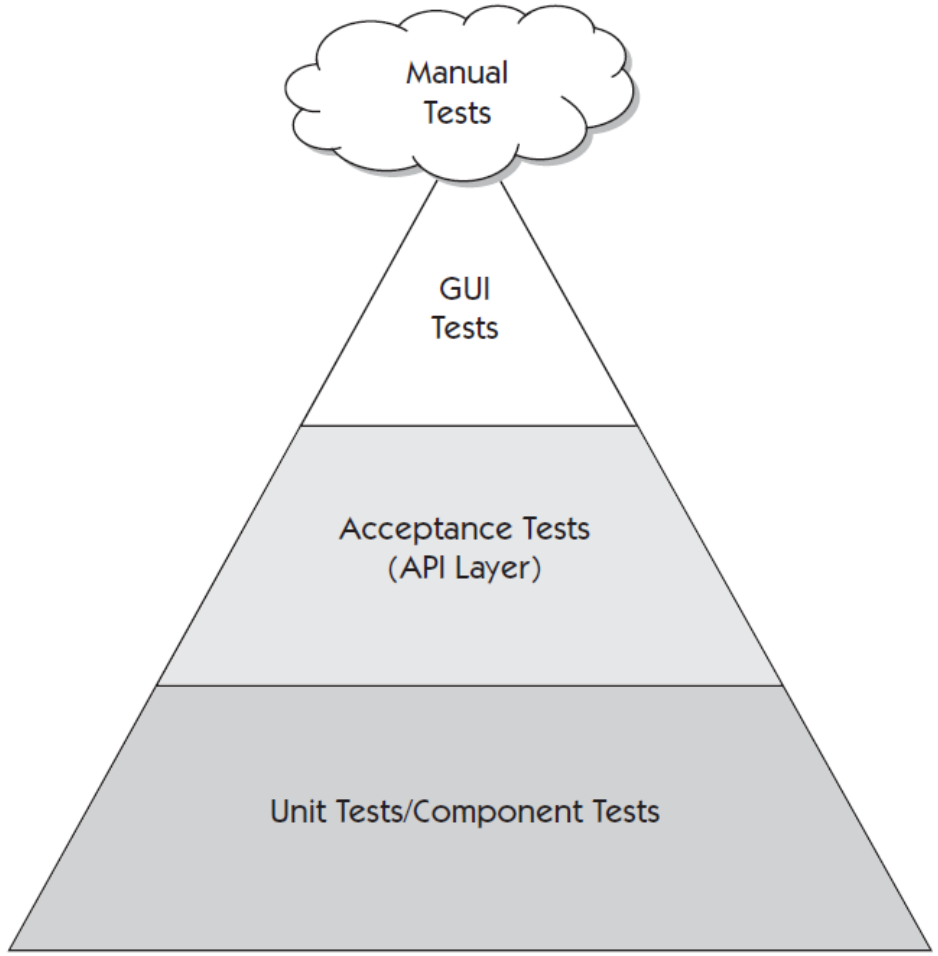
- Agile projects depend on automation.
- Good-enough automation frees the team to deliver high-quality code frequently.
- Automation is a vast topic:
  - writing simple shell scripts
  - setting up session properties
  - creating robust automated tests

Why Automate

- Manual testing takes too long.
- Manual processes are error prone.
- Automation frees people to do their best work.
- Automated tests give feedback early and often.
- Tests provide documentation.
- Automation can be a good return on investment.

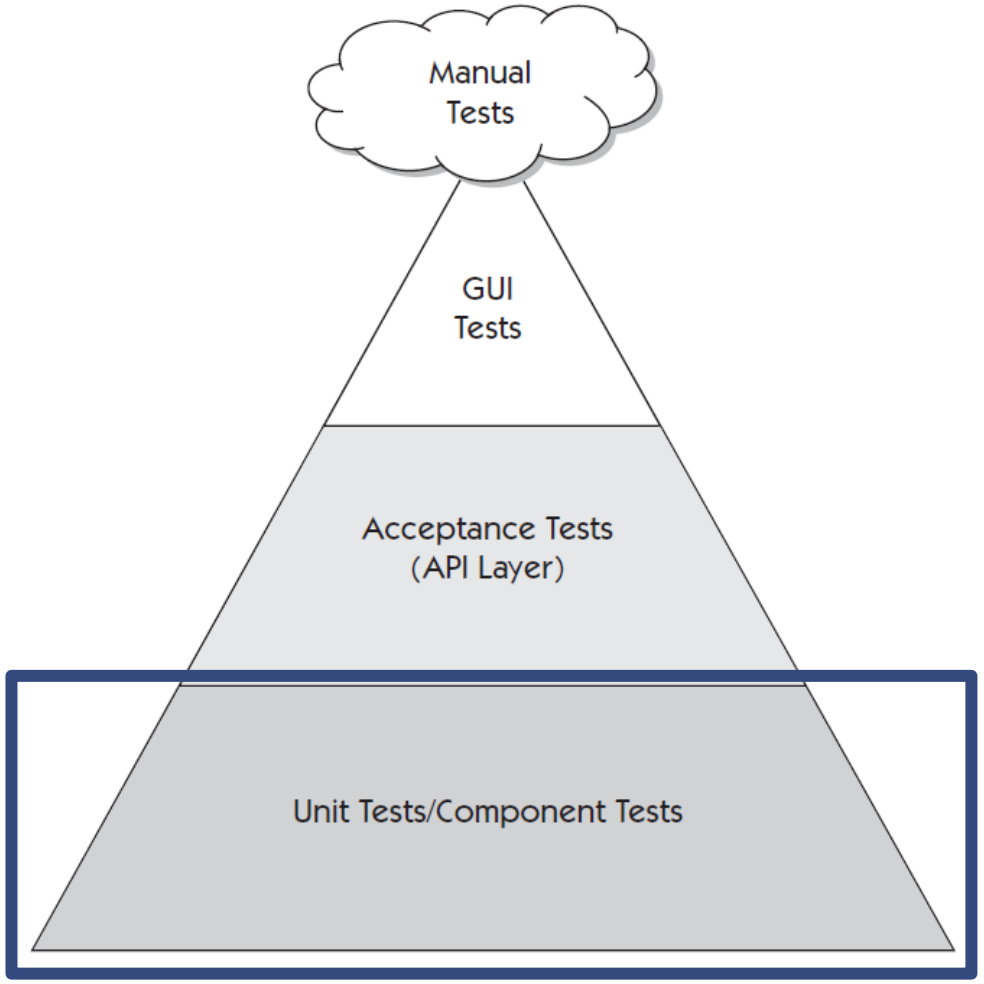
Test Automation Pyramid

- Shows three different layers of automated tests



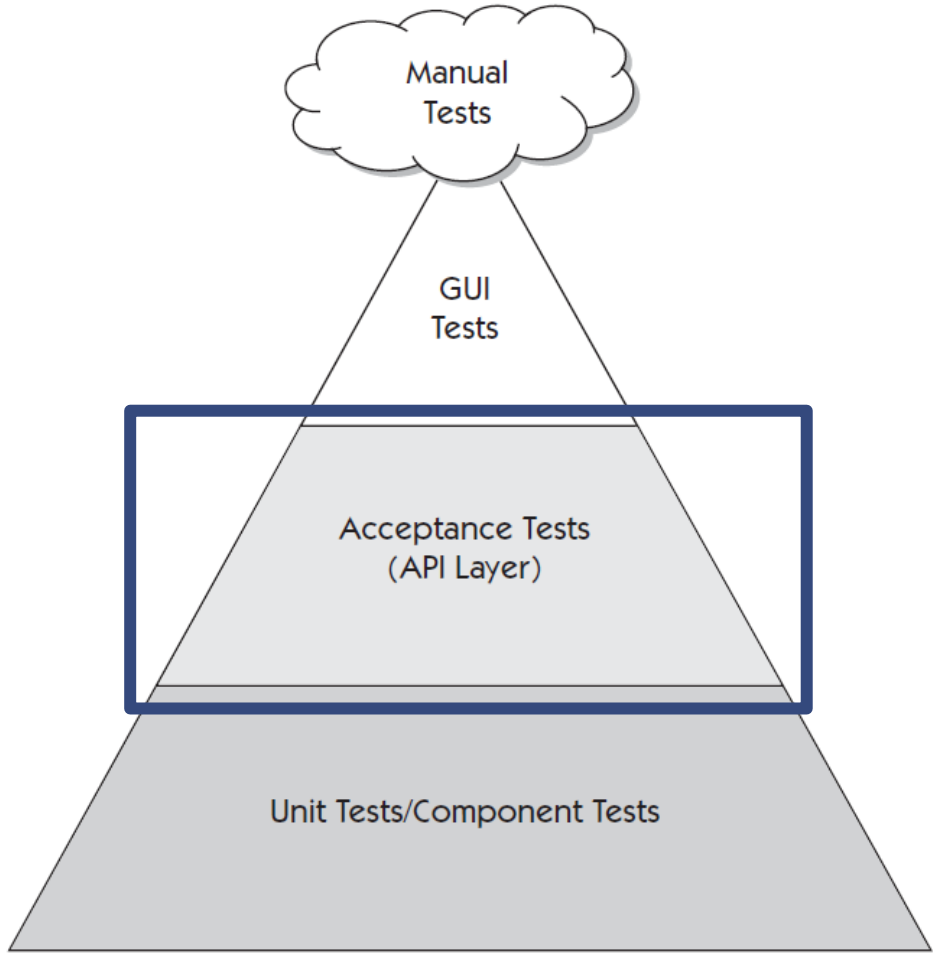
Test Automation Pyramid

- The lowest tier is the foundation of the tests, mainly made up of robust unit tests and component tests.
- These tests are generally written in the same language as the system under test



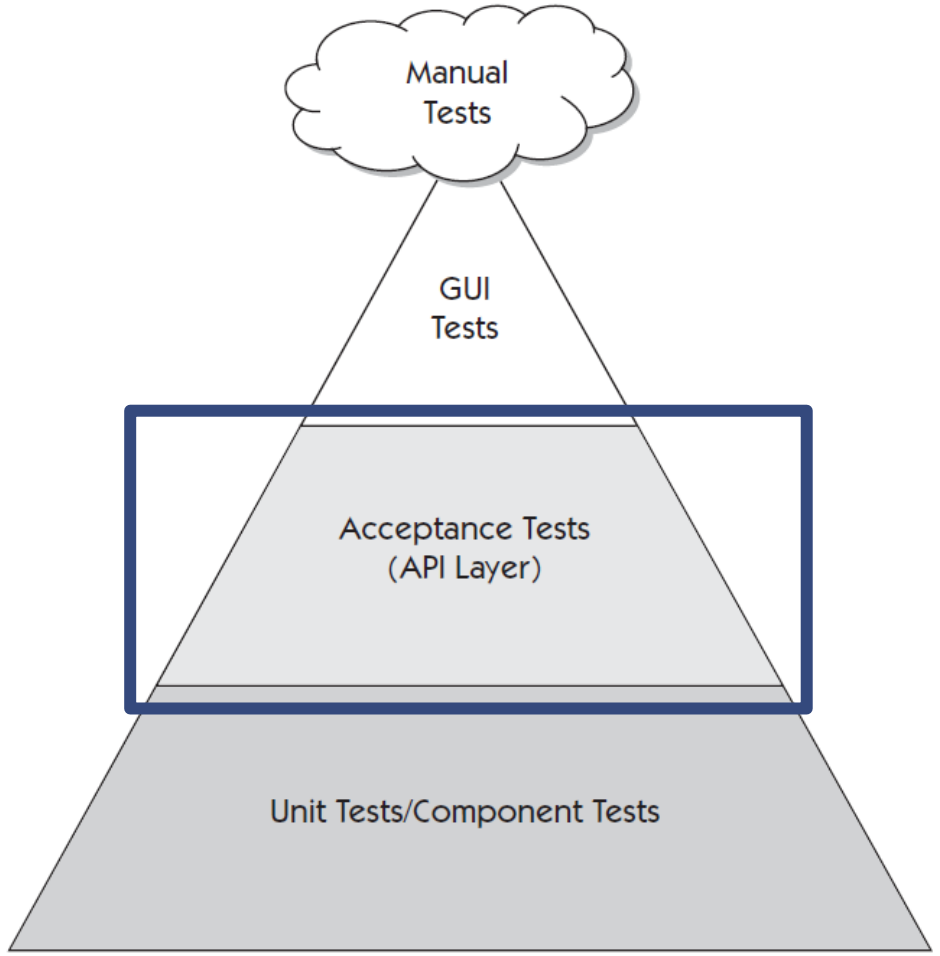
Test Automation Pyramid

- The middle tier includes most of the automated business-facing tests written to support the team.
- These are the functional tests that verify that we are "building the right thing".
- The tests in this layer may include "story" tests and "acceptance" tests.



Test Automation Pyramid

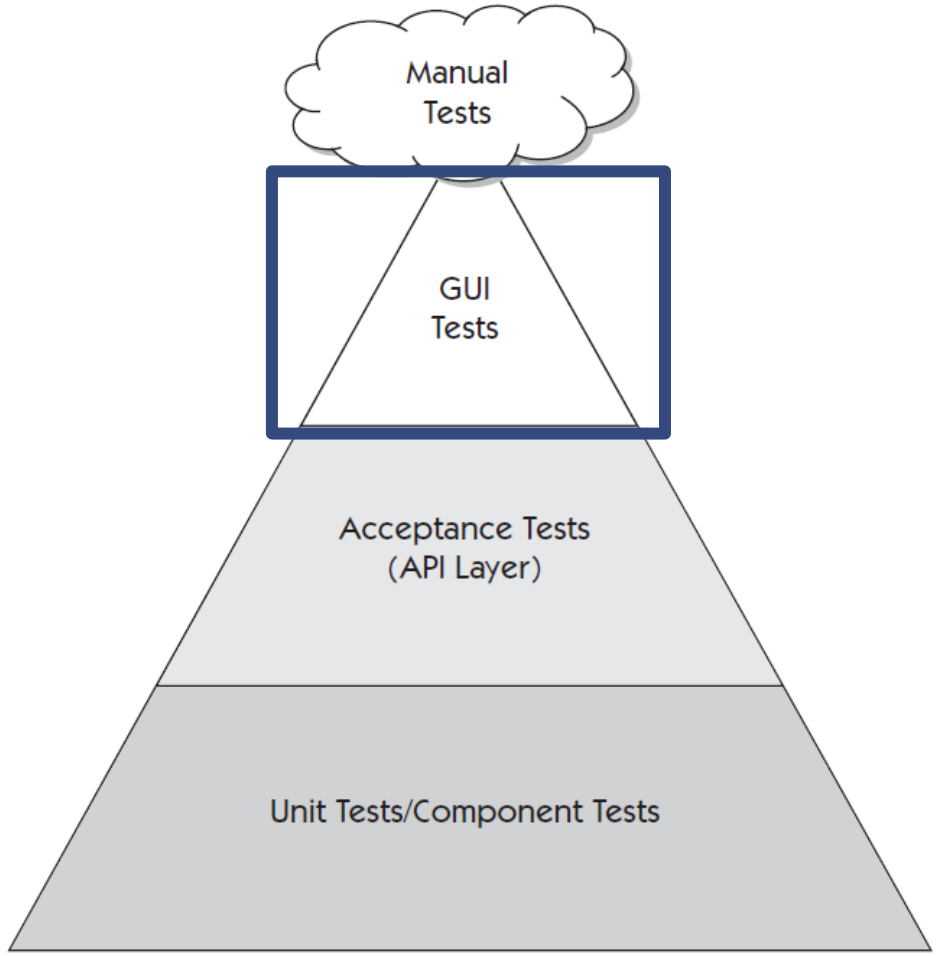
- These tests operate at the API level or "behind the GUI", testing the functionality directly without going through the GUI.



Test Automation Pyramid

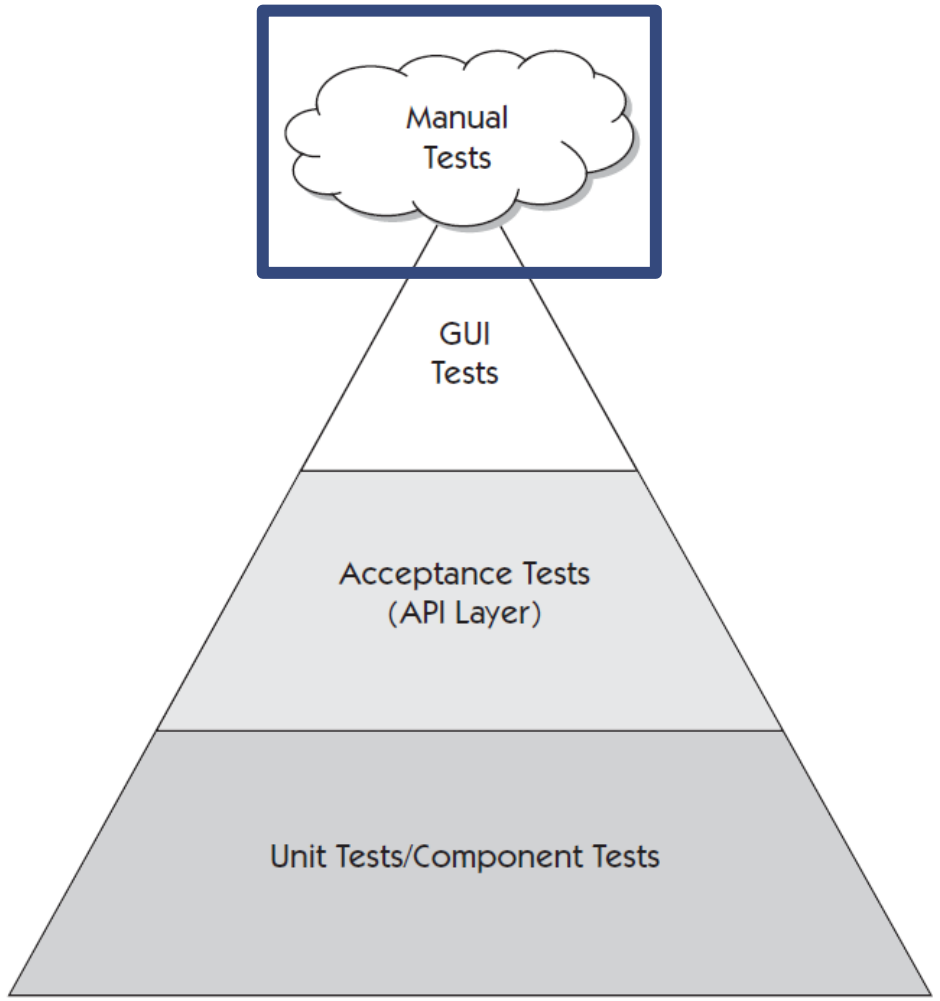
- The top tier represents what should be the smallest automation effort.
- These tests are the ones done through the GUI, the ones that actually operate and manipulate the presentation layer.

Because components of the user interface tend to be changed often, these tests are much more brittle. Ex: just renaming HTML elements could cause a test script to fail



Test Automation Pyramid

- No matter how many automated tests they have, most systems also need manual testing activities, such as exploratory testing and user acceptance testing.
- The bulk of the testing must be automated, or our manual testing won't give a good return on investment.



What Can We Automate?

- Any tedious or repetitive task involved in developing software is a candidate for automation.
- Unit tests and component tests.
- API or Web Services tests.
- Testing behind the GUI.
- Testing the GUI (Tool selection is key for successful GUI automation). An important note here is a stable GUI objects naming convention.
- Load Tests: Like generate a high-volume attack to verify whether a website can be hacked or can handle a large load without some tool framework.

What Shouldn't We Automate?

- Usability Testing: Observing users in action, debriefing them on their experiences, and judging the results, as well as logging user actions.
- Tests that Will Never Fail: As an example, making sure that the second address field in an HTML form is not required; Once the verification is done manually once the chance of someone changing it to be required are very slim.
- One-Off Tests

Appium: an Open Source tool for automating mobile applications

Specflow: is a testing framework supporting BDD practices in .NET framework.

Selenium: an open-source automated testing framework for web applications

Cucumber: A tool for testing Behavior-Driven Development (BDD)

Postman: a tool for testing calls to APIs