Software Testing

Nicholas Mendez

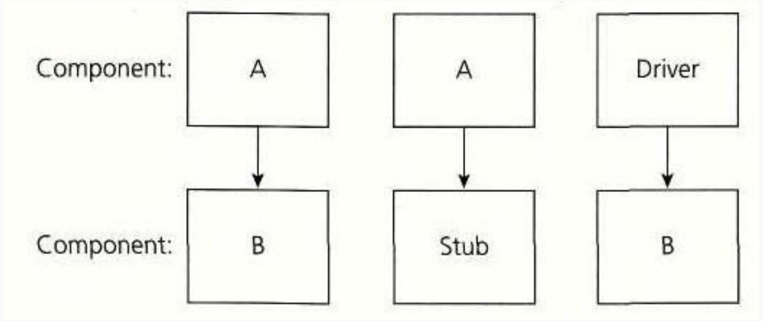
Outline

Introduction

- Testing shows that a program does what it is indeed to do
- Can reveal errors but is not proof of their absence
- Testing is part of the Verification and Validation process
    - Verification: Are we building the product right, conforms to spec. Performed via defect testing.
    - Validation: Are we building the right product, meets the user's needs. Performed via validation testing.
- Validation Testing
    - to demonstrate that the software meetings its requirements
- Defect Testing
    - to discover faults or defects in the system where its behaviour violates the specification

Stubs & Drivers

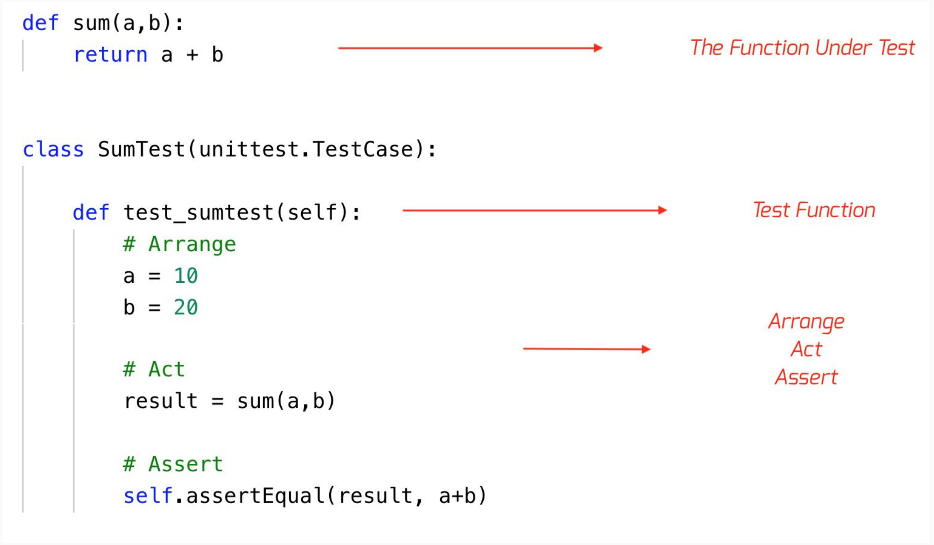
- Often components need to be tested in isolation of the system. Stubs and Drivers are simply abstractions that let us mock external components that are connected to the component to be tested
- Stub: A mock that is called by the tested component
- Driver: A mock that calls the tested component



DriverStub Example

Unit Testing

- Testing individual components of software
- All components other interdependent components are mocked
- Test Cases simply assert if a given input provides the expected output
- Text fixtures setup of requires state for a test. Eg setting up account balance before testing withdrawal



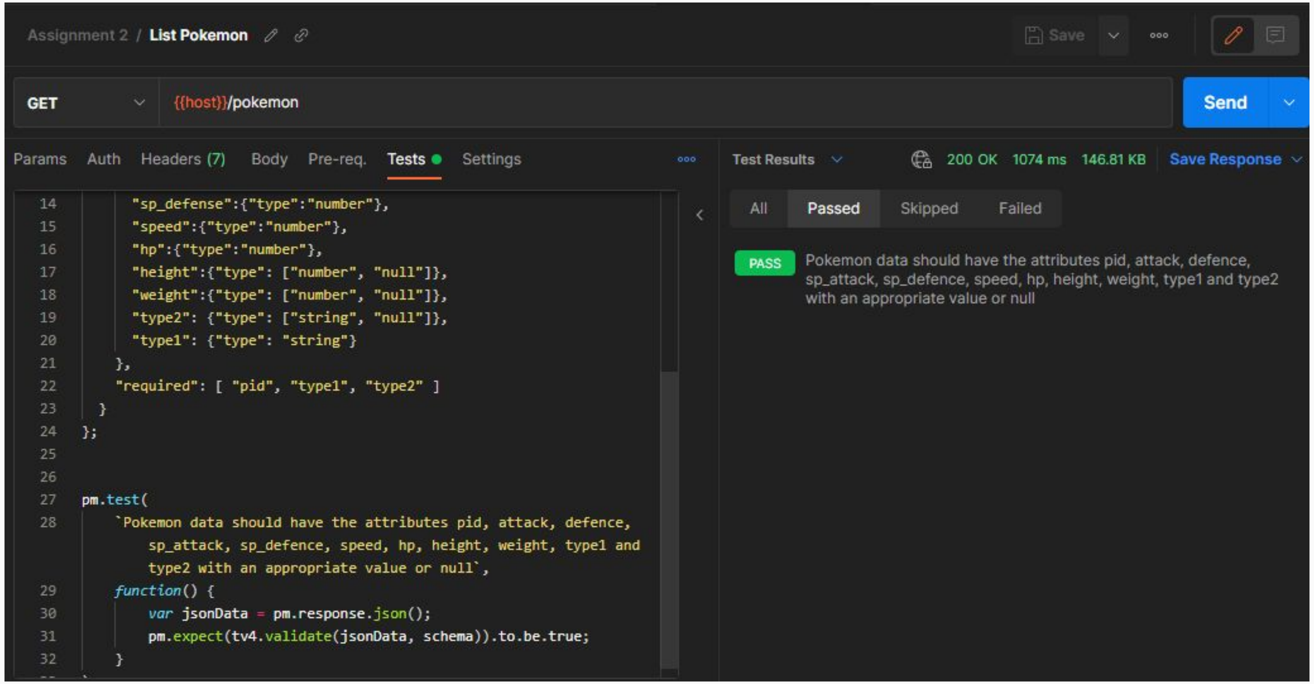
```
def sum(a,b):
    return a + b


class SumTest(unittest.TestCase):

    def test_sumtest(self):
        # Arrange
        a = 10
        b = 20

        # Act
        result = sum(a,b)

        # Assert
        self.assertEqual(result, a+b)
```

Integration Testing

- Validates system requirements
- They test multiple system components or integrations to external systems
- Eg test that information is saved to the database.
- Instead of mocks, a real test environment must be setup. Eg test databases

API Testing

- Testing a REST API calls,
- Validates
  - Request body
  - Response body
  - Status code

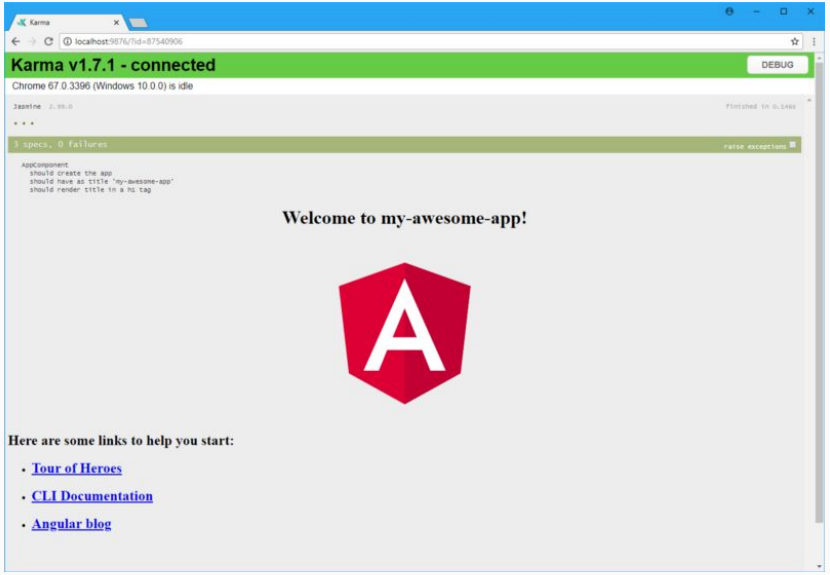


```
        "sp_defense":{"type":"number"},
        "speed":{"type":"number"},
        "hp":{"type":"number"},
        "height":{"type": ["number", "null"]},
        "weight":{"type": ["number", "null"]},
        "type2": {"type": ["string", "null"]},
        "type1": {"type": "string"}
      },
      "required": [ "pid", "type1", "type2" ]
    }
};


pm.test(
    `Pokemon data should have the attributes pid, attack, defence,
        sp_attack, sp_defence, speed, hp, height, weight, type1 and
        type2 with an appropriate value or null`,
    function() {
        var jsonData = pm.response.json();
        pm.expect(tv4.validate(jsonData, schema)).to.be.true;
    }
```

PASS Pokemon data should have the attributes pid, attack, defence, sp_attack, sp_defence, speed, hp, height, weight, type1 and type2 with an appropriate value or null

End to End Testing

- Validates User Requirements
- Ensures that user flows work
    - Eg Login -> add to cart -> checkout
- Uses browser automation tools for testing web applications
- Requires a staging environment which should match the production environment



Karma v1.7.1 - connected

Chrome 67.0.3396 (Windows 10.0.0) is idle

```
Jasmine 2.99.0                                   finished in 0.146s
• • •
3 specs, 0 failures                              raise exceptions
  AppComponent
    should create the app
    should have as title 'my-awesome-app'
    should render title in a h1 tag
```

Welcome to my-awesome-app!

Here are some links to help you start:

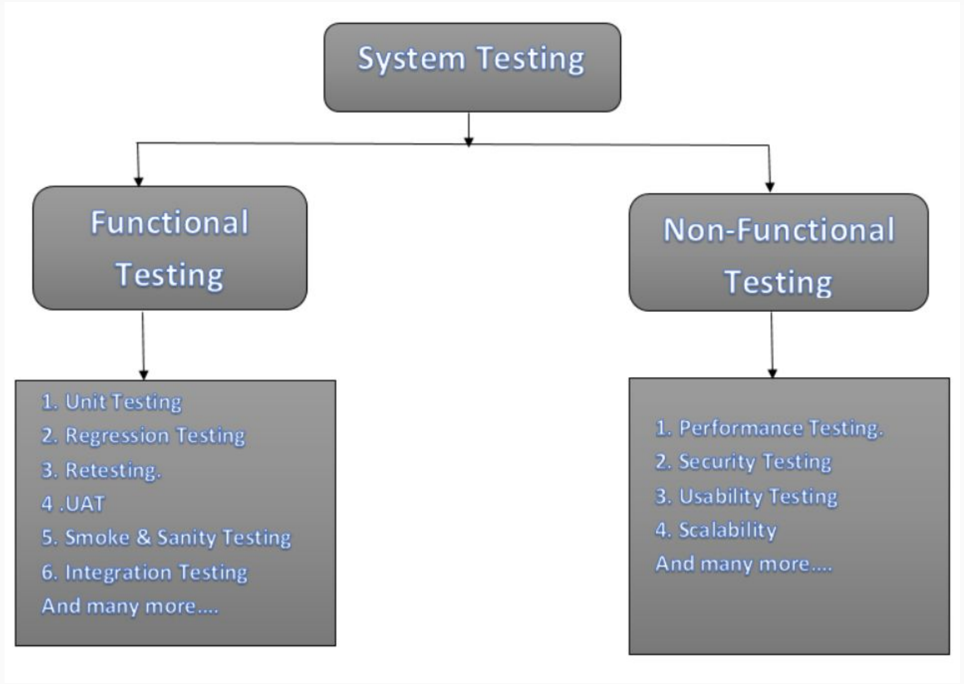
- Tour of Heroes
- CLI Documentation
- Angular blog

Acceptance Testing

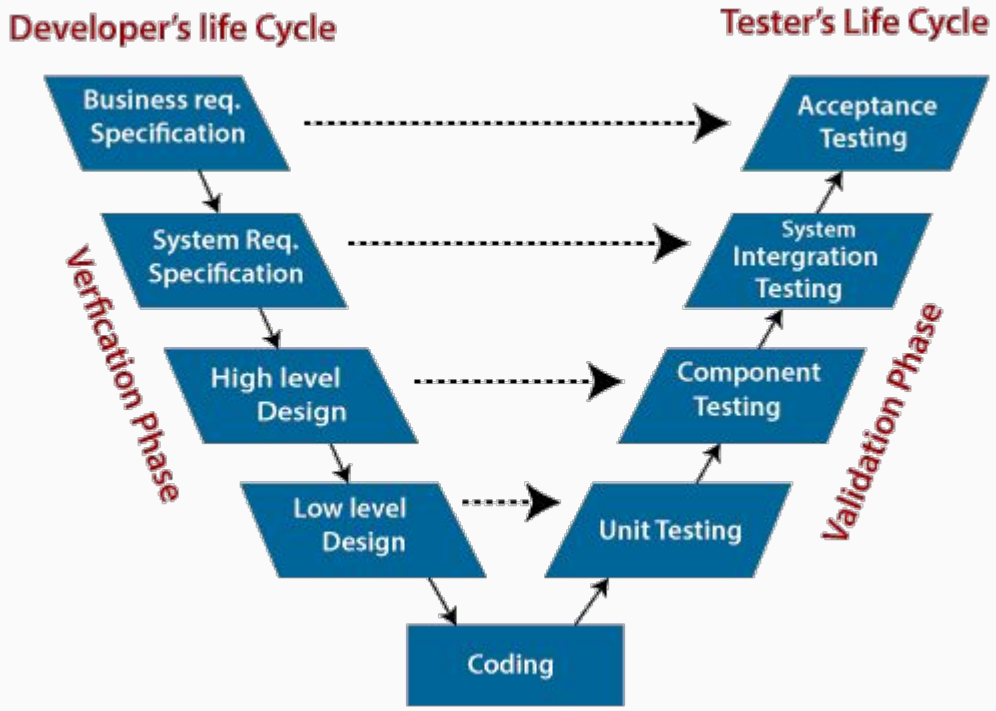
- Manual tests done by a user to validate user requirements
- User perform various tasks on the application and confirm that they conform to user's expectations
- Performed by Product Owner that represents the user base

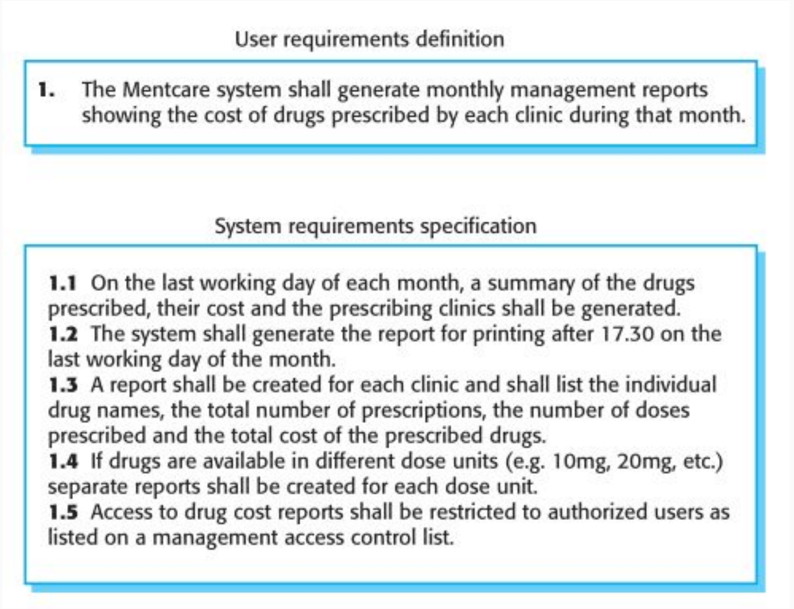
System Testing

- Holistic test of an entire system to validate functional and nonfunctional requirements
- Test plan includes a combination of tests
- Can identify system level errors



Testing V Model





User requirements definition

1. The Mentcare system shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.

System requirements specification

1.1 On the last working day of each month, a summary of the drugs prescribed, their cost and the prescribing clinics shall be generated.
1.2 The system shall generate the report for printing after 17.30 on the last working day of the month.
1.3 A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed and the total cost of the prescribed drugs.
1.4 If drugs are available in different dose units (e.g. 10mg, 20mg, etc.) separate reports shall be created for each dose unit.
1.5 Access to drug cost reports shall be restricted to authorized users as listed on a management access control list.

Testing Presentation

Join us for the first ever Virtual DevFest Caribbean in
Google Developer Groups largest developer event for the year
OCTOBER 16,17 18
GDG
devfest
[Caribbean]2020
#devfest
TO REGISTER VISIT:
bit.ly/register_devfestcarib

Resources

- Writing Good Unit Tests
- System Testing
- Testing Flask Applications
- System VS End-to-End Testing
- Pass The Test Akeem Philbert