**Summary SoftwareTesting**

NB: input =>software=> output

**Introduction to SDLC(software development life cycle) models:**

traditional:

Waterfall model:change is not easy ,costs, outdated.

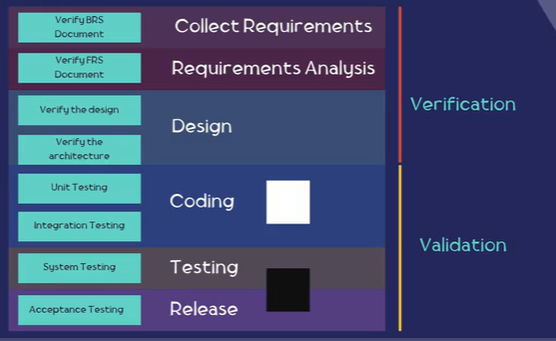
Incremental model:every list has its own waterfall cycle.

V-model:double v (verification,validation):implement test in every step ,we need the software to be ready to start performing Validation

but we don't need it ready to perform verification

**Test steps**

Requirement Collection > Requirement analysis > Design > Coding > Testing > Release

****

**1**-Test user requirements (test sur les documents) : static test(peer review,walkthrough,inspection)

**2-** test the application :dynamic test

--------quiz1-----

-the main document created from the collect requirements step=The Business requirement specifications (BRS)is the output from the collect the requirement phase

-the main document created from the Requirement analysis step=System requirements specification document

-The business analyst or system analyst is the person who creates the Functional requirement specifications (FRS)

---NB:

requirements :functional tests

expectation:non functional tests(rapidité..)

---

**Test levels in SDLC:**

**unit test:**test one module all tests means it cover all statements



**integration test:**

using two manners:

1-bigbang:

wait all modules to be ready before starting any integration test

2-incremental:

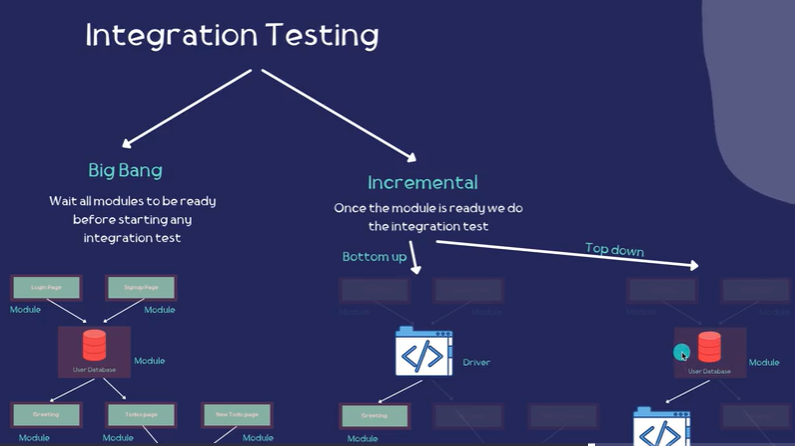
Once the module is ready we do the integration test

top-down :main modules to sub-modules remplacer par stub.

Bottom-up :modules to main module ,remplacer par driver

system integration test:

relier avec autres systèmes d'intégration



------

**System testing :**(functional,and non functional-test)

already unit,integration test are done

test are manual or automatic(automation test (we write code))

-------

**acceptance test**: essentially with customer

alpha testing

Beta testing

—---

**Test techniques**

positive testing (test to pass) vs negative testing (test to fail)

Equivalence partitioning:minimize test cases

Boundary value analysis

**Decision tree** :miro(tool)

**state transition**:visualiser les scénarios possibles

**White box testing:**

statement testing and coverage (every single line)

Decision Testing and coverage if (condition) else ;Flow chart

**Test Process**

depends on:

SDLC type,

test types(security/nonfunctional(rapidité..),

business domain and the product,

budget,

Time,

complexity,

Resources

1-test planning:done by test manager

plan to do well the test ,defining the scope (the one should be testing)

defining the out of scope(the one doesn't need test for this step),

timeline

plan for ressources, for the test team,

2-test Monitoring and control:

-on going process

-compare the plan with the actual progress

-take actions to fix any difference in the plan

-update the test plan

3-test Analysis and Design:

test basis analysis

what to test

what do we need for testing

test conditions

scénarios

test cases

test data (queries)

4-Test Execution and Completion:

test implementation and execution

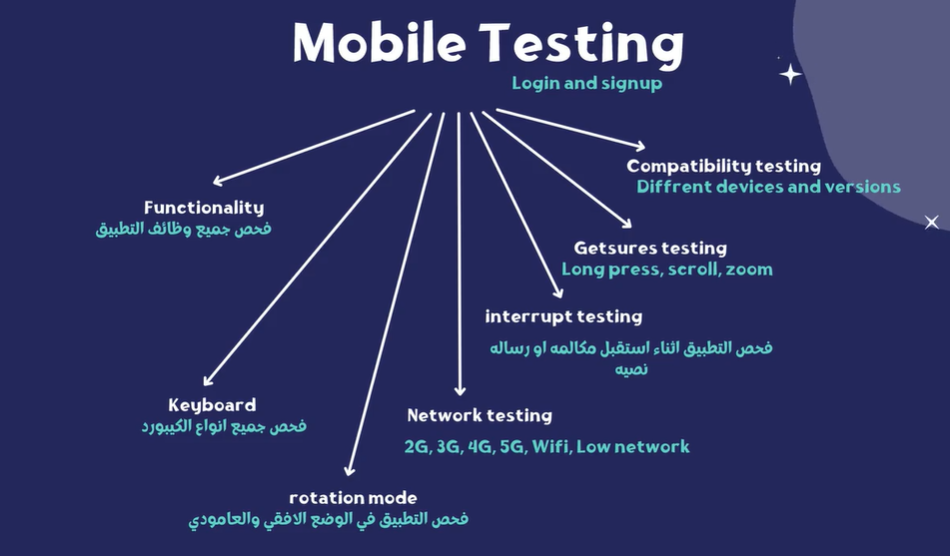
exécute test cases

bug/defect reporting

progress report

-test completion : final progress report by tester,test manager report,lesson and learn

---------

**mobile testing**

**Web testing:**

understand your website

\*access to database

\*access to api document

\*network and api calls

\*download important browsers

\*access to code and run the app locally

Test on non functional testing :

different browsers/different screen size:compatibility test

usability testing

performance testing

security testing

content

accessibility testing

------------

**WebTesting** :

cookies save on hardware laptop,cookie related on specific page web

,local storage:save data on browser akber mel cookie

,session storage: lorsque on ferme browser kol chay yetfasakh

request are stateless

-----

**Developer Tools : Network tab:**

network calls fetch/XHR preview to see api calls

**Developer Tools : application** to see cookies and storage

---

**mock ;changement data api(**tweak )

----

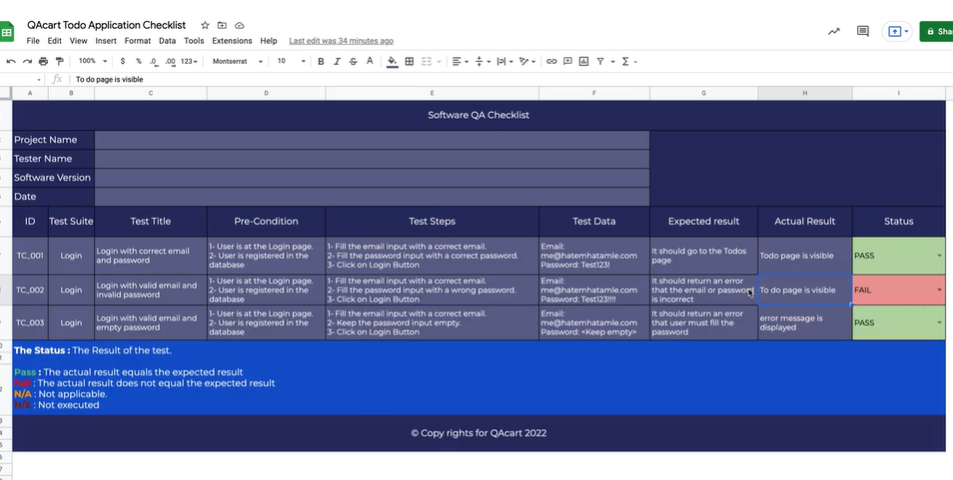
browserStack : on different browsers/operating systems

-----

lighthouse:generate report ;performance/accessibility/seo

**writing testCase**

ID,suite, title ,precondition,test steps,test Data,expected results,actual results,status

**Building checklist:**

-----

FalsePositive:defect in functional or nonfunctional test

false negative:problem in test cases

**Why Agile :**

**Why do we need agile? :** changement , itérative, implication des clients**,**

**focus on individuals and interactions** than processes and tools

**customer collaboration** more important than contract negotiation

**software working** more important than comprehensive documentation

**Responding to change** more important than following a plan