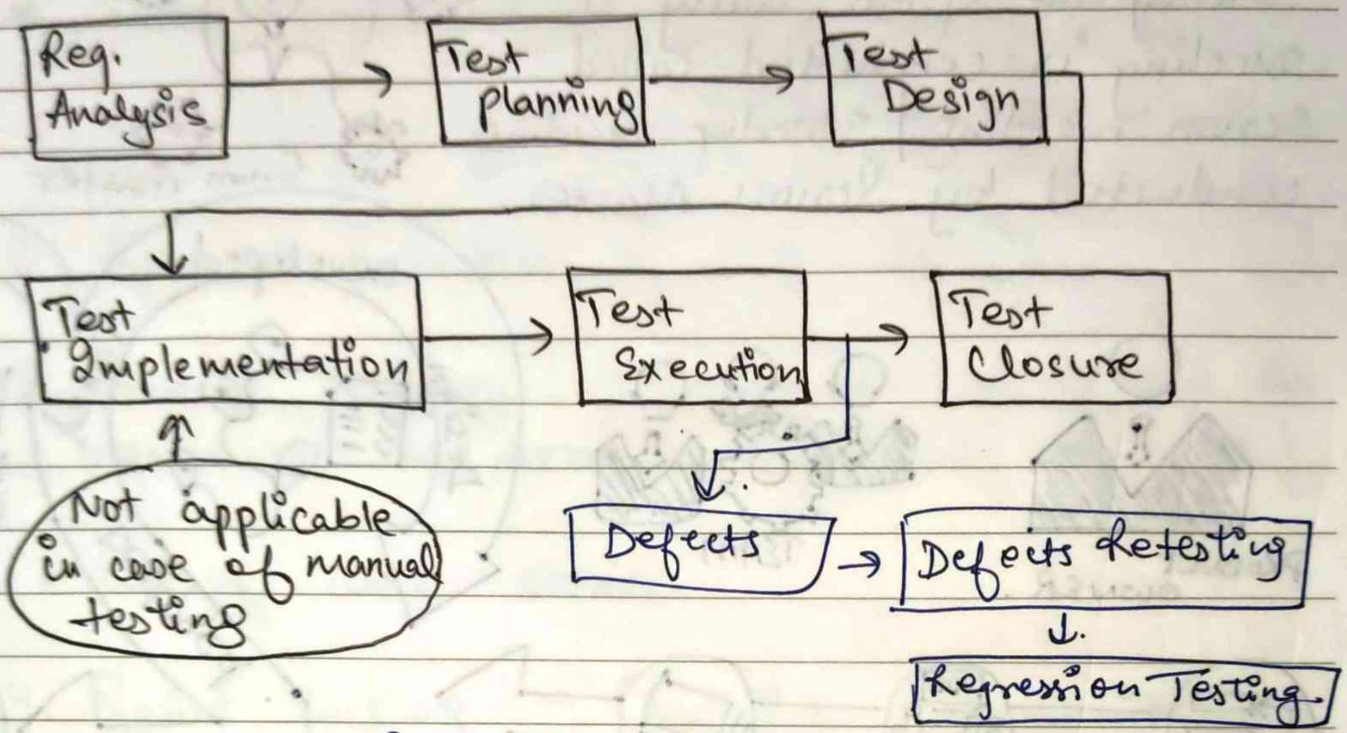
* STLC

Software Testing life Cycle



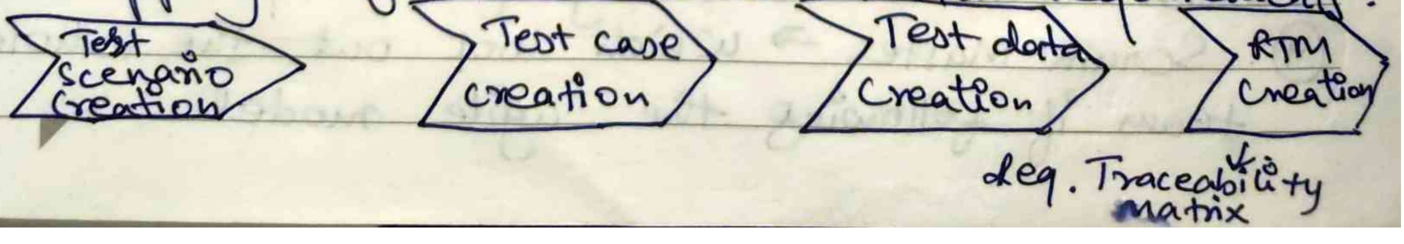
① Req. Analysis → Gathering details about focus area & identifying testable requirements.

② Test Planning → The whole testing process will be detailed out.

- Identifying testing team size, roles & their responsibilities.
- Prepare schedule of testing.

③ Test Design → Identifying test scenarios.

- Create test case for each scenarios.
- Preparation of data for testing.
- Mapping of test cases with requirements.



Test Scenario → What to test in an application.

Facebook

- login
- logout
- Send req.
- Acc. req.
- Del. req.
- Mess.
- Post pic.
- Post vid.
- Write.

Total = 9 scenario

function or Scenarios.

RTM → Requirement Traceability Matrix
where we see, all the cases which were written, were executed or not.

STANDARD TEMPLATE FOR
TEST CASE. → step by step doc. which need to verify any test scenari

| Sr No. | Test Case ID | Objective | Pre-requisite | Test Steps | Test data | Exp. result | Actual result | Status Pass/fail |
|---|---|---|---|---|---|---|---|---|
| 1. | TC_001 | To check<br>---<br>---. |  | 1. Click On--<br>2. Enter ---<br>3. Click On "Submit" |  |  |  | Pass<br>fail |

Pre-condition

eg.

Test Scenario on Ceiling fan

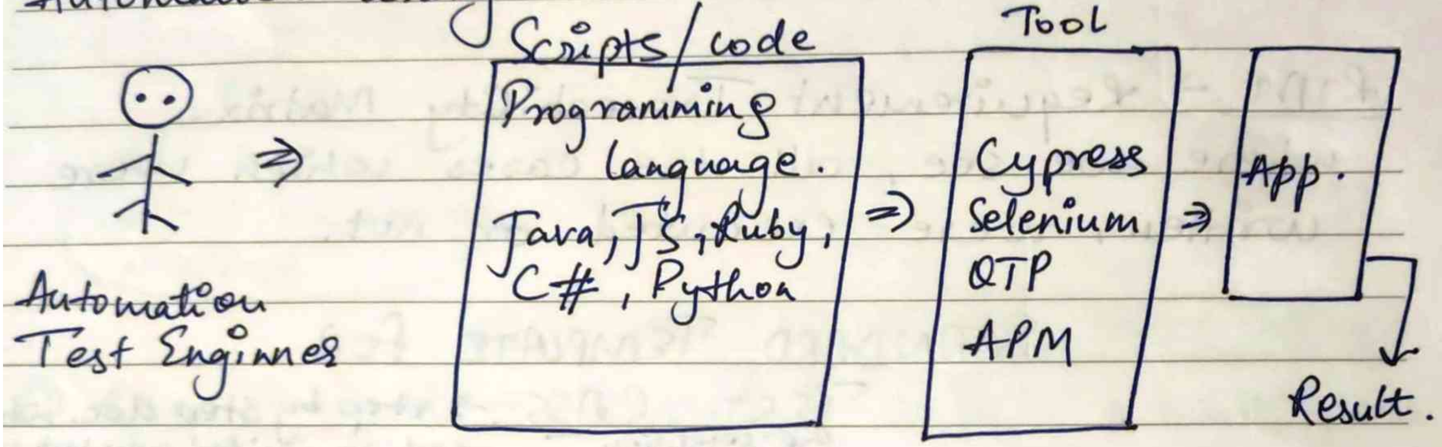
1. Verify the size of the fan.
2. Verify the weight, height, length, design of the fan.
3. verify the colour, brand, type, cost of the fan.
4. Verify the durability, material, quality of the fan.

* Introduction to Automation Testing.

Manual testing: Testing an application with reference requirements. manually.

- Tedious .
- It's time consuming
- We cannot test 24/7.

Automation testing :

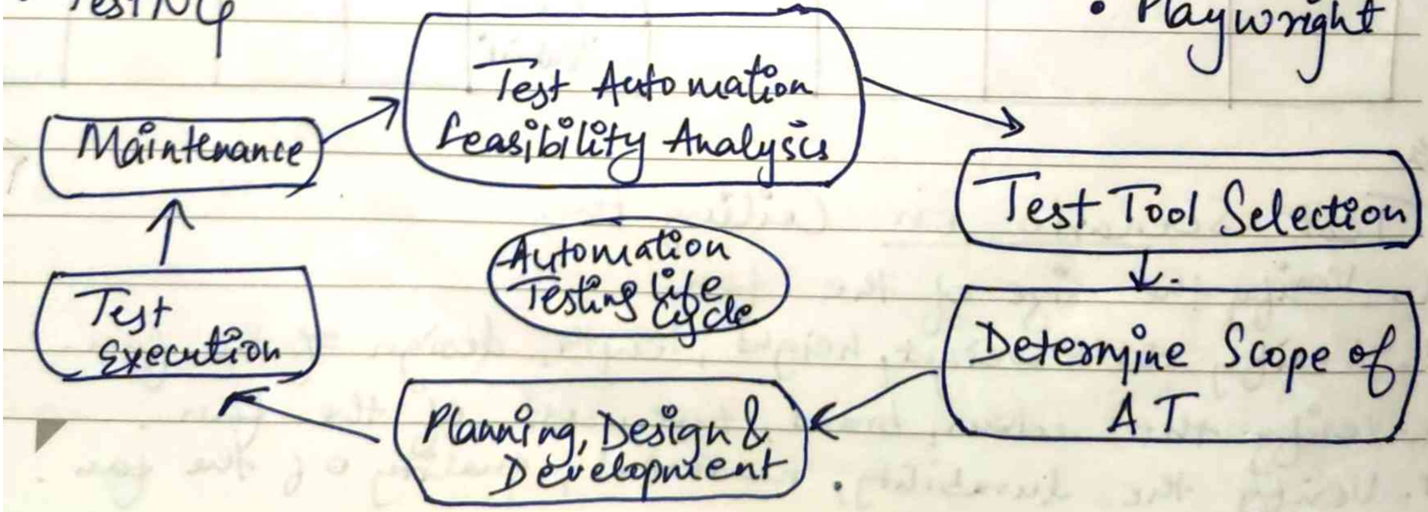


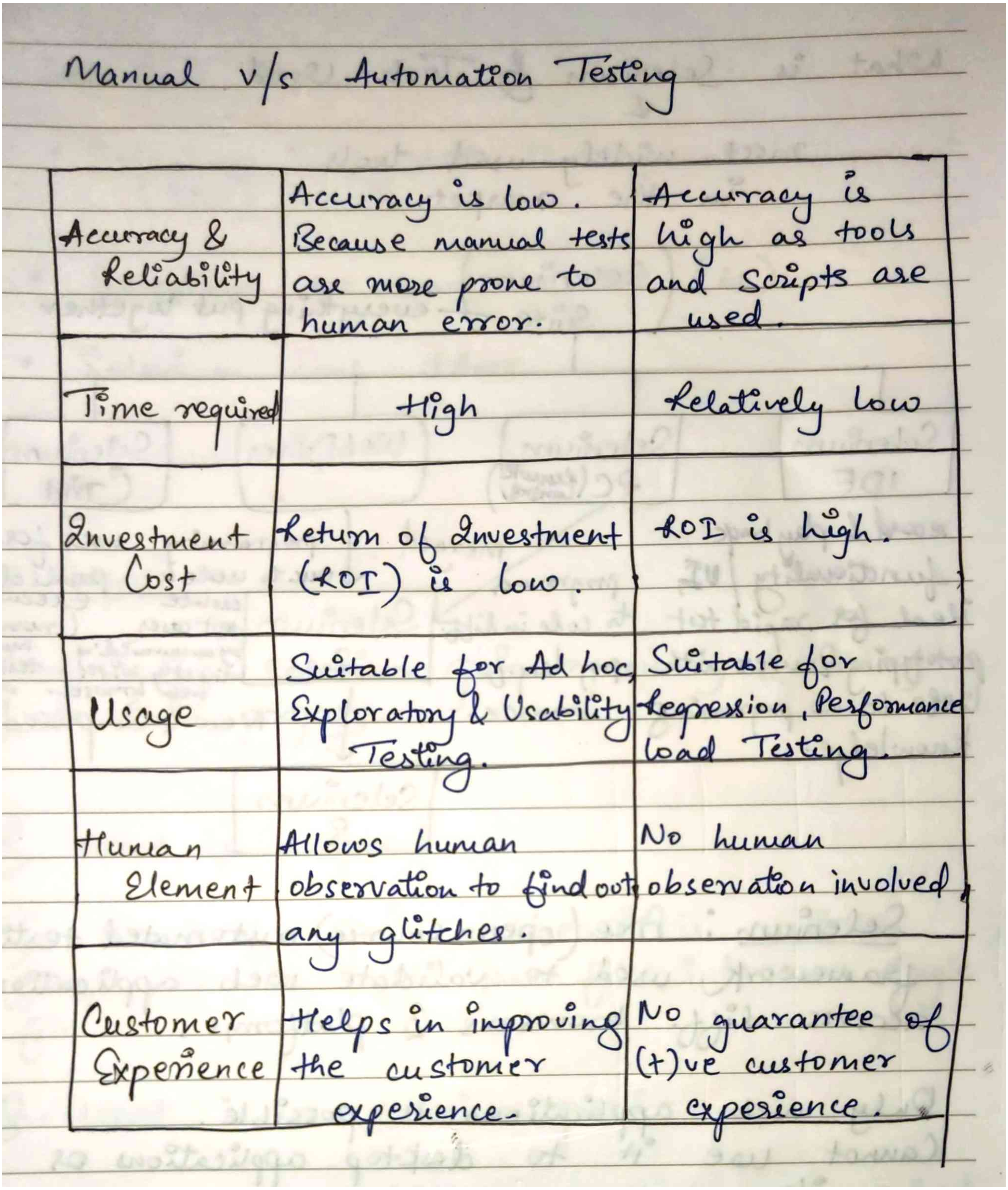
Test automation performs testing at 3 diff. levels:

Unit level
- Nunit
- Junit
- TestNG

API testing
- Rest assured
- Katalon

User Interface
- Selenium
- Cypress
- Playwright


Test Automation Feasibility Analysis
Test Tool Selection
Determine Scope of AT
Planning, Design & Development
Test Execution
Maintenance
Automation Testing Life Cycle



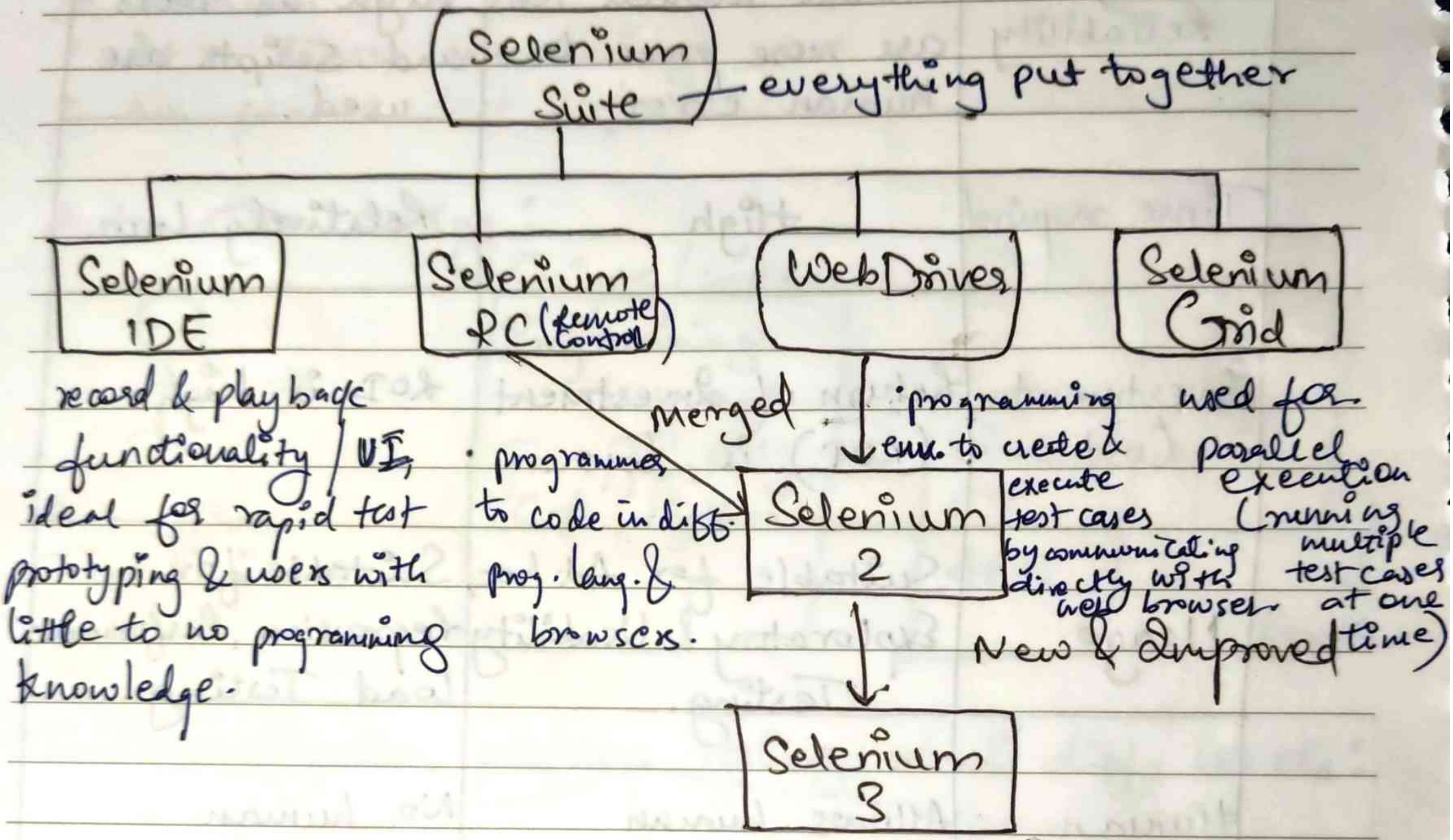
Manual v/s Automation Testing

| | | |
|---|---|---|
| Accuracy & Reliability | Accuracy is low. Because manual tests are more prone to human error. | Accuracy is high as tools and Scripts are used. |
| Time required | High | Relatively low |
| Investment Cost | Return of Investment (ROI) is low. | ROI is high. |
| Usage | Suitable for Ad hoc, Exploratory & Usability Testing. | Suitable for Regression, Performance load Testing. |
| Human Element | Allows human observation to find out any glitches. | No human observation involved. |
| Customer Experience | Helps in improving the customer experience. | No guarantee of (+)ve customer experience. |

What is Selenium & Tools Used.

↓

most widely used tools in the market.



record & playback functionality / UI, ideal for rapid test prototyping & users with little to no programming knowledge.

· programmers to code in diff. prog. lang. & browsers.

· programming env. to create & execute test cases by communicating directly with web browser.

used for parallel execution (running multiple test cases at one time)

<u>Selenium</u> : free (open-source) automated testing framework used to validate web applications accross diff. browsers & platforms.

Only web applications is possible.
Cannot use it to desktop applications or mobile applications.

<u>Driver in Selenium</u>: It is a s/w which enables interactions b/w java program & the browser.

Selenium tools.

- Selenium integrated development environment (IDE)
- Selenium remote control (RC)
- Selenium web driver.
- Selenium Grid.

# Projects.

*Downloading selenium Jars manually & adding to our projects.

Q. what is a Jar?

Jar stands for Java archive, It is used to group two or more Java files into one single file.

Q. what is a selenium Jar?

All the classes & interfaces which are required to automate the web application using Java is grouped together as selenium Jars.

## CSS selectors

1. tag id — tag#id
2. tag class — tag.classname
3. tag attribute — tag[attribute = "value"]
4. tag class attribute — tag.classname[attribute = "value"]

**Thread** : It is used for the delay.

```
psvm (String [] args) {                    setting up the browser
    try {
going to the Chrome browser
    WebDriverManager . chromedriver (). setup ();
    ChromeDriver  driver =  new  ChromeDriver ();
    Thread. sleep (10000);  // delay of 10 sec.
navigate to particular website
    driver . get (" www. google. com ") ;
    }
      Catch (Exception e) {
        sopln (e. getMessage ());
      }
   }
}.
```

* Cucumber BDD : Cucmber is a testing approach which supports(Behavior driven development) BDD. It explains the behavior of the application in a simple English using Gherkin language.
In BDD, test cases are written in simple language so that non technical people can read.

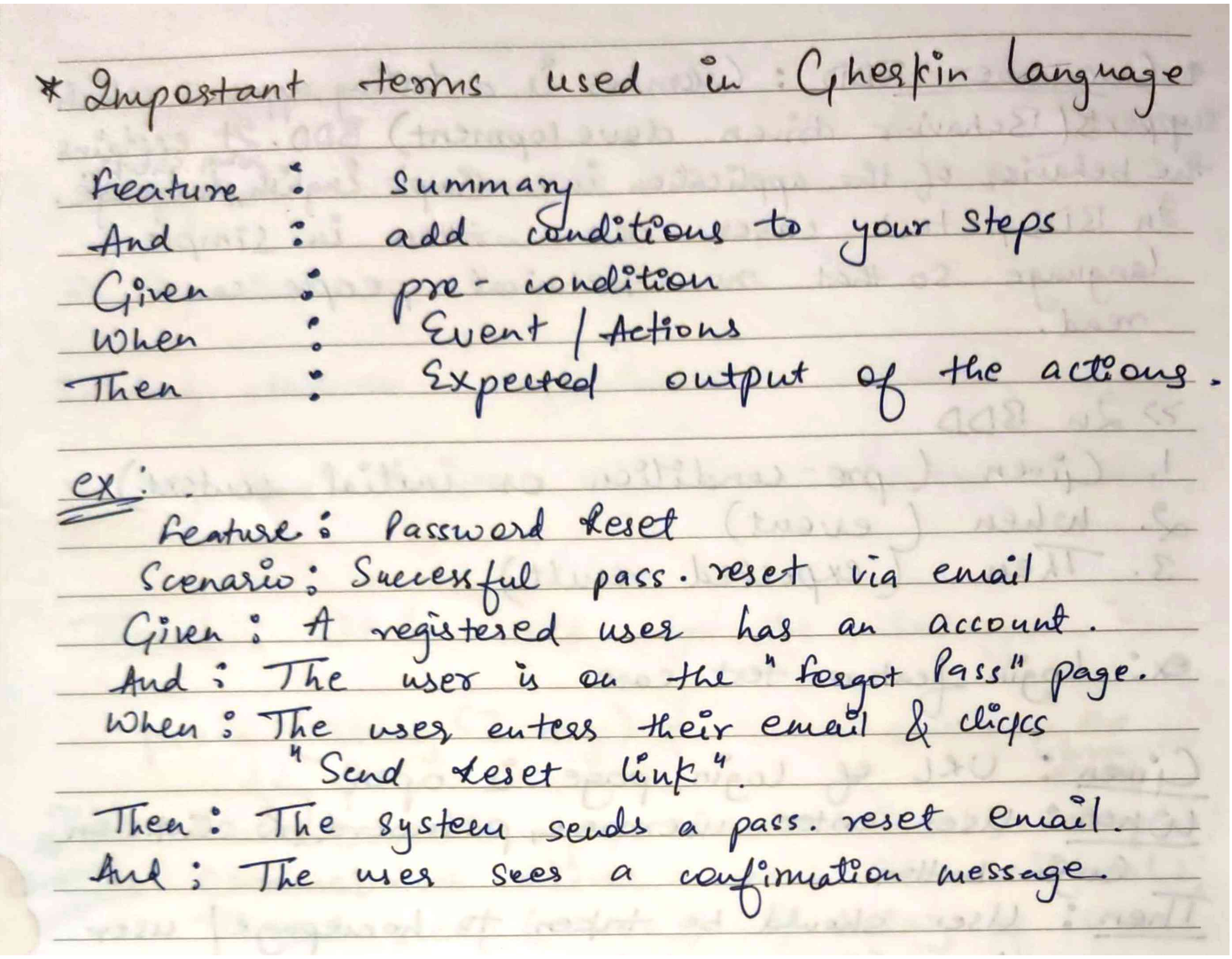
>> In BDD
1. Given ( pre-condition or initial context)
2. When ( event)
3. Then ( expected result)

ex: Login feature test case

Given : URL of login page is open.
When : User enters username, password, & clicks on login button.
Then : User should be taken to homepage / user should successfully login.

* TDD (Test Driven Development)

Test case are described in cucumber feature file. feature file should have extension .feature and each feature file should have only one feature.



* Important terms used in Gherkin language

Feature : summary
And : add conditions to your steps
Given : pre-condition
When : Event / Actions
Then : Expected output of the actions.

ex :

Feature : Password Reset
Scenario : Successful pass. reset via email
Given : A registered user has an account.
And : The user is on the "forgot Pass" page.
When : The user enters their email & clicks "Send reset link".
Then : The system sends a pass. reset email.
And ; The user sees a confirmation message.