# GIT LAB TASK MANAGEMENT/DEFECT REPORTING

# Overview

- What is a Defect/Bug?
- What is a Defect Life Cycle?
- What status do we have for a task?
- How do we change the status?
- What is bug and bug reporting?
- What is a task ?
- Task Status
- Task Assigning Parameters
- Priority of  a task

# What is a Defect/ Bug?

- A **defect** is a variation or deviation from the original business requirements
- A **bug** is the consequence/outcome of a coding fault

  When a tester executes the test cases, he/she might come across the test result which is contradictory to expected result.

- This variation in the test result is referred as a **Software Defect**. These defects or variation are referred by different names in a different organization like **issues, problem, bug or incidents**.

# A DEFECT REPORT SHOULD:

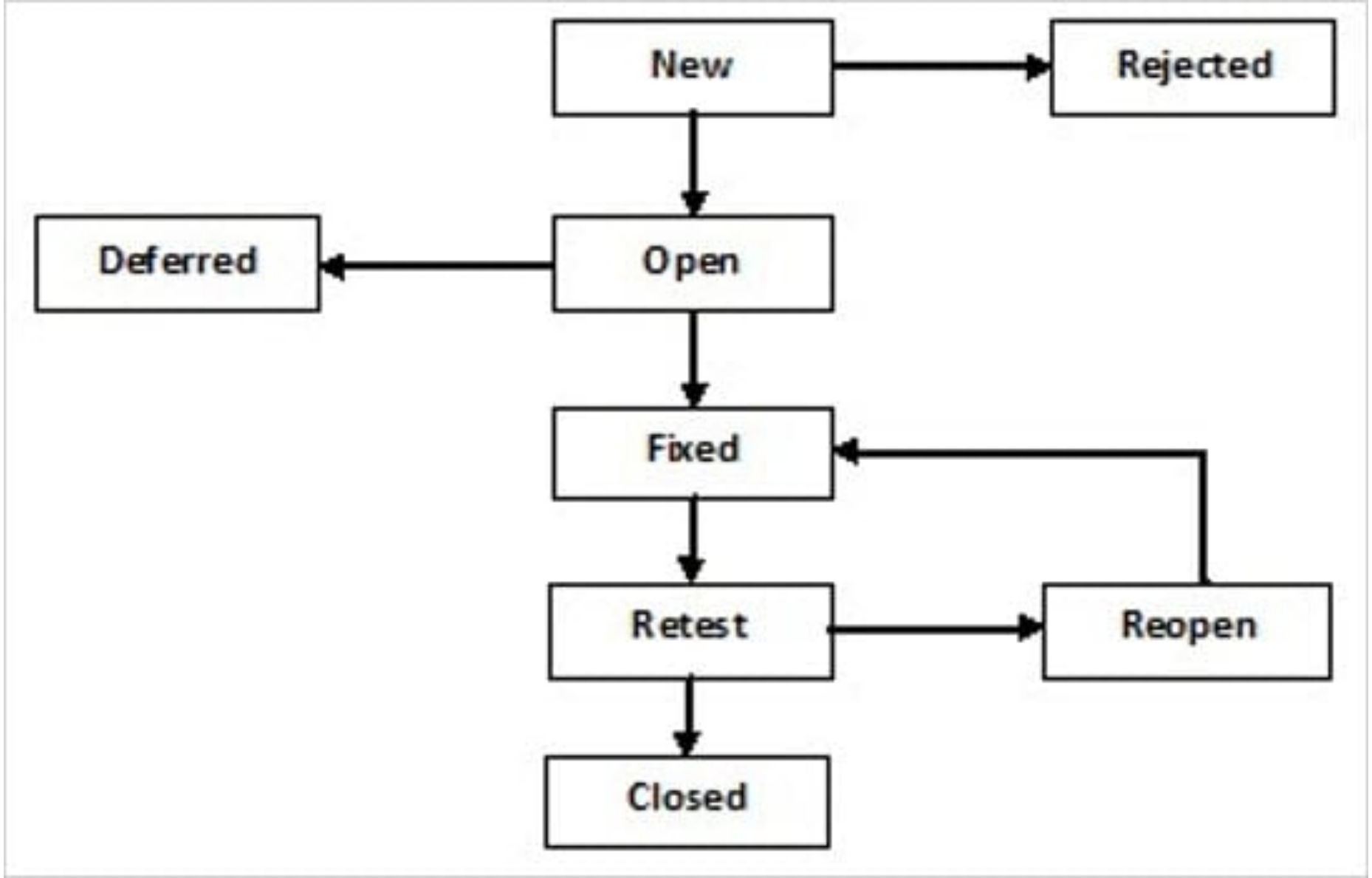
- Be well written.
- **Reproducible**: make it easy to reproduce the defect by developer
- **Specific**: be specific and to the point
- **Informational:** add more useful information
- Should have sufficient and high quality information to reproduce and fix defects in the software.
  - Note: The report should also enable stakeholders to make wise decisions about the defects that need to be fixed.

# A DEFECT LIFE CYCLE

**Defect Life Cycle** or Bug Life Cycle in software testing is the specific set of states that defect or bug goes through in its entire life.

The **purpose** of Defect life cycle is to easily coordinate and communicate current status of defect which changes to various assignees and make the defect fixing process systematic and efficient.

# THE DEFECT STATES/CYCLE



# A DEFECT CYCLES

- **New:** When a new defect is logged and posted for the first time. It is assigned a status as NEW.
- **Assigned:** Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team
- **Open/Active**: The developer starts analyzing and works on the defect fix
- **Fixed**: When a developer makes a necessary code change and verifies the change, he or she can make bug status as “Fixed.”
- **Verified**: The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified.”
- **Retest**: Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to “Re-test.”

# A DEFECT CYCLES cntd…

**Reopen**: If the bug persists even after the developer has fixed the bug, the tester changes the status to “reopened”. Once again the bug goes through the life cycle.

**Closed**: If the bug is no longer exists then tester assigns the status “Closed.”

**Duplicate**: If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate.”

**Rejected**: If the developer feels the defect is not a genuine defect then it changes the defect to “rejected.”.A defect can be rejected for any of the 3 reasons; viz - duplicate defect, NOT a Defect, Non Reproducible.

**Deferred**: If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status “Deferred” is assigned to such bugs

# Defect Report Sample with data

- **Example: Defect Report for  Employee Login Page**
- **Defect id:** D001
- **Project Name:** MyASP
- **Module Name:** Login
- **Sub Module Name:** Employee Login
- **Type of Defect:** Missing
- **Status:** New
- **Priority :** High
- **Summary:** Employee Login Page Not Opening
- **Description :**
  Enter Url
  Click On the Employee Login Option
- **Expected Result:** Employee login page should get open
- **Actual Results:** Employee login page does not get open
- Reported By: ABC Tester
- Assign To: XYZ Developer
- Date & Time: 12/28/2019

# TASKS

- Basically, finalized project requirements divided into individual sections.
- Tasks are created and assigned using different Task management tools such as
- Gitlab
- Other Defect tracking tools

# TASK STATUS

- **Backlog/ Open** : Task is created but not assigned.
- **To Do** : Task has been assigned.
- **In progress/ Doing** : Started working on the task.
- **Done/ Fixed** : Task done by dev
- **Review**: Ready to be tested by QA
- **Closed** : Task Tested and Verified By QA

# TASK ASSIGNING PARAMETERS

1. Subject
2. Description
3. Attachments
4. Assigning

# 1. Subject..

- Should cover THE MODULE NAME
- Should not include:
  - Long Explanations and unnecessary Information

Example:

[ Profile Module] – Implement Personal Information Section

# 2. Description..

- Should Contain:

  A very brief explanation about the actual requirements of the tasks.

  Steps to do the tasks.

  Expected Result

- Should Not Contain:

    Informal Languages and texts

    Same content as in Subjects

    Deviation from subjects and unrelated issues

## Example:

**Subject:**

[ Profile Module] – Implement Personal Information Section

**Description:**

This module should allow the users to enter the user details like First name, Last name, Permanent Address, Temporary Address, Gender…etc.

Steps:

a. Click on the Profile icon.
b. Enter the First name.
c. Enter the Last name.
d. Upload the User Image
e. Click on the Submit Button.

Expected Result: The user should be able to fill in the details and submit the form.

# 3. Attachments..

Can be :

- Documents (SRS portions)
- Design Links
- Understandable by novice (new user)

# 4. Assigning..

- Developer/programmer - particular project for particular purpose
- Yourself

# PRIORITY

Priority: How important it is to be done now.

Priority: High/ Medium /Low

# Assignment

1. Create a Gitlab account->Project(public) and setup an issue board and create few issues/tasks with proper data. (gitlab.com)
2. Class Assignment: Class groupism

# REFERENCE LINKS

1. Scrum VS Kanban board: https://www.lucidchart.com/blog/kanban-vs-scrum

1. Scrum Basics

   What Is Scrum Methodology? & Scrum Project Management

*Any queries ?*

**THANK YOU !**