1. What is priority?

- Priority is Relative and Business-Focused. Priority defines the order in which we should resolve a defect. Should we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mentioned then the developer has to fix it at the earliest. The priority status is set based on the customer requirements.

For example:

  - If the company name is misspelled in the home page of the website, then the priority is high and severity is low to fix it.

Priority can be of following types:

  - **Low**:-
    - The defect is an irritant which should be repaired, but repair can be deferred until after more serious defect has been fixed.
  - **Medium**::-
    - The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created.
  - **High:-**
    - The defect must be resolved as soon as possible because the defect is affecting the application or the product severely. The system cannot be used until the repair has been done.
    - **Critical:** Extremely urgent, resolve immediately

2. What is severity?

- Severity is absolute and Customer-Focused. It is the extent to which the defect can affect the software. In other words it defines the impact that a given defect has on the system.

For example:

- If an application or web page crashes when a remote link is clicked, in this case clicking the remote link by an user is rare but the impact of application crashing is severe. So the severity is high but priority is low.

Critical: The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable and there is no acceptable alternative method to achieve the required results then the severity will be stated as critical.

Severity can be of following types:

- **Major (High):**
  - The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable but there exists an acceptable alternative method to achieve the required results then the severity will be stated as major.
- **Moderate (Medium):**
  - The defect that does not result in the termination, but causes the system to produce incorrect, incomplete or inconsistent results then the severity will be stated as moderate.
- **Minor (Low):**
  - The defect that does not result in the termination and does not damage the usability of the system and the desired results can be easily obtained by working around the defects then the severity is stated as minor.

Cosmetic: The defect that is related to the enhancement of the system where the changes are related to the look and field of the application then the severity is stated as cosmetic.

3. Bug categories are…

- **Data Quality/Database Bugs:**
  - Deals with improper handling of data in the database.

  Example :-

  - Values not deleted/inserted into the database properly
  - Improper/wrong/null values inserted in place of the actual values

- **Critical Functionality Bugs:**
  - The occurrence of these bugs hampers the crucial Functionality of the application.

  Example :-

  - Exceptions

- **Functionality Bugs:**
  - These Bugs affect the functionality of the application.

  Examples :-

  - All JavaScript errors
  - Buttons like Save, Delete, Cancel not performing their intended functions
  - A missing functionality (or) a feature not functioning the way it is intended to
  - Continuous execution of loops

- **Security Bugs:**
  - Application security Bugs generally involve improper handling of data sent from the user to the application. These Bugs are the most severe and given highest priority for a fix.

  Examples :-

  - Authentication: Accepting an invalid username/password
  - Authorization: Accessibility to pages though permission not given

- **User Interface Bugs:**
  - As the name suggests, the bugs deal with problems related to U are usually considered less severe.

  Examples:

  - Improper error/warning/UI messages
  - Spelling mistakes
  - Alignment problems

4. Advantage of Bugzila.

- Bugzilla is also a bug tracking tool developed by the Mozilla Foundation in 1998. It is a quite popular reporting tool which is having a simple user interface. Due to its simple interface, beginners take no time to understand its workflow.

- Bugzilla is also a bug tracking tool developed by the Mozilla Foundation in 1998. It is a quite popular reporting tool which is having a simple user interface. Due to its simple interface, beginners take no time to understand its workflow.

Advantages of Bugzilla

- It improves the quality of the product.
- It enhances the communication between the developing team and the testing team.
- It has the capability to adapt to multiple situations.
- Key features of Bugzilla include
- Advanced search capabilities
- E-mail Notifications
- Modify/file Bugs by e-mail
- Time tracking
- Strong security
- Customization
- Localization

5. Difference between priority and severity

Priority	Severity
The sequence in which the developer should resolve defects is specified by Defect Priority.	The defect severity of a fault is defined as the influence it has on the product's operation.
Priority is divided into three categories. • Low • Medium • High	There are five levels of severity. • Critical • Major • Moderate • Minor • Cosmetic
Priority has to do with scheduling.	The term "severity" refers to the degree to which something is functional or adheres to a set of standards.
The priority of a bug determines how quickly it should be repaired.	The severity of a problem on a product's functionality is indicated by its severity.
In consultation with the manager/client, the priority of faults is determined.	The defect's severity level is determined by the QA engineer.
The business value determines priority.	The severity of a situation is determined by its functioning.
When a problem has a high priority and low severity, it means it has to be corrected right away but isn't affecting the application.	When a fault has a high severity and a low priority, it means it has to be corrected, but not right now.
The priority status is determined by the needs of the consumer.	The product's technical aspect determines the severity level.
During UAT, the development team prioritizes faults and fixes them.	During SIT, the development team will prioritize and resolve bugs based on their severity.

6. Mention what are the categories of defects

- **Data Quality/Database Bugs:**
  - Deals with improper handling of data in the database.

  Example :-

    - Values not deleted/inserted into the database properly
    - Improper/wrong/null values inserted in place of the actual values

- **Critical Functionality Bugs:**
  - The occurrence of these bugs hampers the crucial Functionality of the application.

  Example :-

    - Exceptions

- **Functionality Bugs:**
  - These Bugs affect the functionality of the application.

  Examples :-

    - All JavaScript errors
    - Buttons like Save, Delete, Cancel not performing their intended functions
    - A missing functionality (or) a feature not functioning the way it is intended to
    - Continuous execution of loops

- **Security Bugs:**
  - Application security Bugs generally involve improper handling of data sent from the user to the application. These Bugs are the most severe and given highest priority for a fix.

  Examples :-

    - Authentication: Accepting an invalid username/password
    - Authorization: Accessibility to pages though permission not given

- **User Interface Bugs:**
  - As the name suggests, the bugs deal with problems related to U are usually considered less severe.

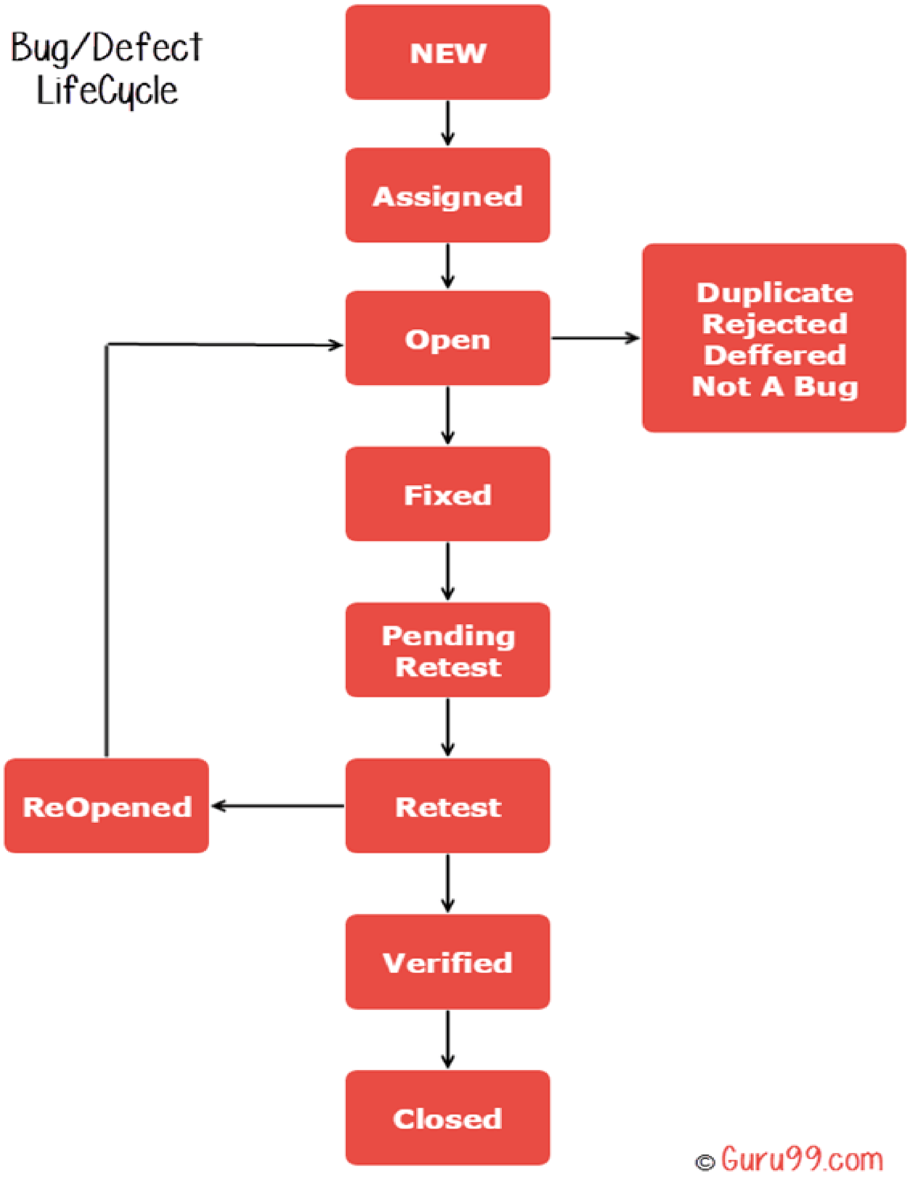
  Examples:

    - Improper error/warning/UI messages
    - Spelling mistakes
    - Alignment problems

7. What is Bug Life Cycle ?

- The duration or time span between the first time defects is found and the time that it is closed successfully, rejected, postponed or deferred is called as ‘Defect Life Cycle’**.**

- The standard bug life cycle is fully established with multiple states which helps to cover as much as possible all the stages during the entire life of a bug. This full version of bug life cycle is usually adopted by large organizations or enterprises where the development process is complicated, or using a traditional process such as waterfall approach, or the main channel of communication between developer and tester is via bug life cycle.



- **New:**
  - When a new bug gets discovered the first time, it is marked as New.
- **Assigned:**
  - After a QA Lead or Product Manager approves the bug, it is then assigned to a developer who will take responsibility to fix it.
- **Open:**
  - The developer will investigate and start working to fix this bug.
- **Fixed:**
  - The bug is changed to Fixed when a developer makes necessary code changes and confirms the change.
- **Pending Retest:**
  - The bug is marked as Pending Retest after the developer conducted a code change, and this bug is waiting for the tester to test again.
- **Retest:**
  - Tester will be performing retesting on this bug to determine that the bug is fixed or not.
- **Verified:**
  - If the bug does not happen in the application anymore, the tester will approve that this bug is fixed entirely and change the status to Verified.
- **Reopen:**
  - If the bug still occurs when the tester retests it, its status is changed to Reopen and pushed back to the life cycle.
- **Closed:**
  - Tester will change the status to Closed if this bug no longer exists in the application.

- **Duplicate:**
    - When two bugs refer to the same error, one of them will be marked as Duplicate.
- **Rejected:**
    - This Rejected status is marked for a bug if the developer confirms that this bug is not new.
- **Deferred:**
    - If a bug is changed to Deferred, that means the team will address this bug in the next release. Many factors contribute to a bug being Deferred, such as the bug being low priority, low severity, or limited time for release.
- **Not a bug:**
    - If there is no functionality affected by the bug. Example: a request to change text or color in UI. Or a bug is only introduced when testing on a wrong environment/test data. Then the bug is marked as Not a bug.

8. Difference between authorization and authentication.

Authentication	Authorization
Authentication is the process of identifying a user to provide access to a system.	Authorization is the process of giving permission to access the resources.
In this, the user or client and server are verified.	In this, it is verified that if the user is allowed through the defined policies and rules.
It is usually performed before the authorization.	It is usually done once the user is successfully authenticated.
It requires the login details of the user, such as user name & password, etc.	It requires the user's privilege or security level.
Data is provided through the Token Ids.	Data is provided through the access tokens.
Example: Entering Login details is necessary for the employees to authenticate themselves to access the organizational emails or software.	**Example:** After employees successfully authenticate themselves, they can access and work on certain functions only as per their roles and profiles.
Authentication credentials can be partially changed by the user as per the requirement.	Authorization permissions cannot be changed by the user. The permissions are given to a user by the owner/manager of the system, and he can only change it.