1: Can you please Introduce yourself:

2: What is the team size you work in?

In my current project, we work in an Agile Scrum environment with a 9-member team. We have 4 developers, 2 testers (including myself), a Test Lead, a Business Analyst, a Scrum Master, and a Product Owner. We all work together closely to make sure we deliver quality software in each sprint.

3: How do you report a defect?

When I find a bug, I retest it to make sure it's real. I also check with a fellow tester to see if they can find it in their environment. Before logging it, I talk with the developer to make sure everything is clear. Then, I document the bug with a title, detailed steps, expected vs. actual behavior, severity, priority, and a screenshot. We also have a bug triage process to prioritize and manage issues effectively.

4: When do you have your sprint planning meeting?

Before each sprint, we have a Sprint Planning meeting where we talk about what we can realistically get done in the next sprint. We go over and prioritize items from the Product Backlog, assign story points, and commit to the tasks we will complete in the next sprint.

5: Do you have sprint retrospective meeting and what do you do?

Yes, we have a sprint retrospective meeting. After the review meeting, we take some time to look back on how the last sprint went. We talk about what went well, what didn't go as planned, and where we can improve. We also discuss how we can make the next sprint better.

6: if you cannot finish a task in your sprint, when do you finish it?

We usually finish our tasks within the sprint. But if a task doesn't get done, it's moved to the product backlog. In the next sprint planning meeting, the team and the Product Owner look at its priority again. If it's still important, we include it in the upcoming sprint.

7: how are you assigned a task within a sprint?

In our daily scrum calls, we talk about our progress so everyone knows who's doing what and where the most value is being added. By the time we get to sprint planning, it's pretty

clear who should take on specific tasks. For example, if I've been working on automation testing for a particular module, I'll probably keep handling similar tasks in the next sprint.

8: Who do you report to in your scrum team?

In my current project, we don't formally report to anyone, but we work closely with the Scrum Master, the BA, and our team lead. Our team is self-managed, so collaboration and communication are really important. We make sure everyone is on the same page and doing their part to keep the project on track.

9: When do you conduct your daily stand-up meetings, and what do you discuss?

We have our daily stand-up meetings every morning. In these meetings, we talk about what we did the day before, what we plan to do that day, and if there are any blockers or Impediments that need to be solved.

10: How do you resolve ambiguous Requirement?

To resolve ambiguous requirements, we first discuss the issue as a team, usually during our daily stand-up or a quick meeting. We bring in the Business Analyst and Product Owner to clarify any details. If we still need more clarity, we might have a short workshop or chat with the stakeholders.

11: How do you prioritize tasks within a Sprint?

I don't prioritize tasks myself; that's up to the Product Owner. They decide what's most important based on the value to the product and business. During Sprint Planning, we talk about these priorities as a team and make sure the most important tasks are tackled first.

12: What is the bug life cycle?

The bug life cycle usually starts with the bug being marked as **New** when it's first discovered. Then it's Assigned to a developer to fix. Once the developer starts working on it, the status changes to **Open**. After the fix is applied, the bug is marked as **Fixed**. The tester then verifies the fix, and if everything is working as expected, the bug is marked as **Verified** and finally **Closed**.

In some cases, a bug might be **Deferred** if it's decided to fix it later, **Rejected** if it's not considered a bug, or marked as a **Duplicate** if it's already been reported.

13: What is reopening a bug?

A bug is reopened when it fails the retest after being marked as fixed. If the issue still exists, we change the status to "Reopened" so the developer can take another look at it.

14:. What is the difference between the severity and priority of a bug?

Severity means how much the bug affects the system or software—basically, how serious the issue is. Priority, on the other hand, is about how quickly the bug needs to be fixed—how urgent we need to deal with the bug.

15: What steps do you take when a critical bug is found during a Sprint?

When a critical bug is found during a Sprint, I log it right away with all the important details. Then, I immediately inform the team, especially if it could affect our Sprint goals. We usually prioritize fixing it right away. If needed, we might adjust the Sprint scope or timeline to make sure the issue is addressed. In these situations, quick action and teamwork are very important in this situation.

16: If you have a release tomorrow and haven't finished all the testing what will you do?

If I have a release tomorrow and haven't finished all the testing, the first thing I'd do is communicate the situation to the team and stakeholders. We'd conduct a risk assessment to identify the most critical areas that need testing. I'd prioritize running smoke tests to check the basic functionality, followed by regression and sanity tests to ensure that the recent changes haven't broken anything important. If there are still high-risk areas that haven't been fully tested, I'd recommend either postponing the release or doing a partial release. It is important to be upfront about the risks and ensure everyone knows before deciding.

17: what if your developer does not agree on a bug. What will you do?

If a developer doesn't agree with me about a bug, I'd first try to understand their perspective by discussing the issue in detail. I'd explain why I believe it's a bug, using specific examples and evidence from my testing. If we still don't agree, I'd involve the Business Analyst or Product Owner to get their input, as they can clarify the requirements or expected behavior. It's not about finding fault; it's about collaboration and clear communication to ensure the best outcome.

18: How would you test a door?

To test a door, I'd start by checking its basic functionality:

Open and Close: Ensure the door opens and closes smoothly without any issues.

Lock and Unlock: Test the locking mechanism to make sure it locks and unlocks properly.

Handle Operation: Verify that the handle works correctly and doesn't stick or jam.

Alignment: Check that the door is properly aligned with the frame and doesn't scrape the floor or get stuck.

Hinges: Inspect the hinges to make sure they support the door's weight and allow smooth movement.

Safety Features: If the door has safety features, like a peephole or auto-close mechanism, test those as well.

Overall, I'd focus on usability, safety, and functionality to make sure the door works as it should in various scenarios.

19: what is a build, a version, a release, and deployment

Build: A build is a specific instance of software that has been compiled and put together from the source code. It's essentially the result of the development team's work up to a certain point, often used for testing purposes.

Version: A version represents a specific state of the software, identified by a unique number (like 1.0, 2.1.3, etc.). Each version usually includes a set of new features, bug fixes, or improvements.

Release: A release is when a specific version of the software is made available to users. It's the official launch of that version, often after thorough testing to ensure it's ready for production.

Deployment: Deployment is the process of delivering the software to its intended environment, like a server or a user's device. This is when the software goes live and becomes usable in the real world.

20: Do you do regression testing? Which test cases do you keep in regression

Yes, I do regression testing, especially when there's a new enhancement or a bug fix. I focus on checking the existing functionality to make sure nothing is broken by the new changes. I include test cases that cover critical and core functionality, previously fixed defects, and any new changes or enhancements. We call these our regression checkpoints, making sure everything still works smoothly.

21: how do you know when to stop testing ?

We know when to stop testing by following specific exit criteria. First, we make sure all the planned test cases are run, and at least 95% of them pass. Any remaining test cases should be low-priority or have been deferred with proper approval.

Next, we check that all critical and high-priority bugs have been fixed, and there are no open issues that could impact the release. We also ensure that we've covered all the requirements and user stories, confirmed through our Requirement Traceability Matrix (RTM). Finally, we complete regression testing to make sure new changes haven't caused any new problems.

Once all these boxes are checked, we know it's safe to stop testing and move forward.

22: Can you write a test plan document?

Yes, I can write a test plan. In the test plan, I'd include the scope of testing, the objectives, the resources we have, and the schedule. I'd also outline the test environment, entrance and exit criteria, types of testing we'll perform, and a risk assessment. Finally, I'd list the deliverables we need to produce. The goal is to have a clear roadmap for the testing process so everyone knows what to expect and how we'll ensure the quality of the product.

23: How do you make sure your test has a very good coverage

I make sure my tests have good coverage by using a Requirement Traceability Matrix (RTM), where I map each requirement to its corresponding test case. This way, I can see that every requirement is tested. I also use different testing techniques like positive and negative testing, boundary value analysis, and decision table testing. Plus, I prepare a variety of test data to cover different scenarios, making sure all parts of the application are thoroughly tested.

24: When do you do your smoke test?

We do a smoke test whenever a new build is deployed. It's a quick check to make sure the core critical functions are working before we dive into deeper testing. Especially when a build is new and might be unstable, smoke testing helps us find major issues in early stage.

25: Have you ever done any testing without documentation?

Yes, I've done testing without documentation before. In those cases, I rely on my understanding of the application, discussions with the team, and any available requirements or user stories. I also focus on exploratory testing to find issues based on how users might interact with the software.