ASMT:

1. Why you want to go for assessment.

* As a key Developer in my current project I am already doing the Lead activities from the fast 2 years,
* I took the responsibility of one of the scrum team (portal concurs) acting as a scrum master for this team, working with PO to pull the user stories to sprint backlogs on sprint grooming, estimating the use stories and doing negotiation with PO on story points.
* Setting up the priorities to the pulled user stories. Raising ambiguities on sprint backlog stories. Conducting sprint ceremonies (sprint planning, daily standup, sprint review and sprint retro spective).
* I am a responsible to verify code quality, by putting check points on peer-reviews, encouraging the team to do the proper analysis with respective team members, security violations, code coverage by writing unit test cases,
* To mitigate the unplanned leave vs Deliverables as a team we committed to push the code to repository every day to his own branch, so we don’t have any dependency wit

<https://grow.epam.com/skillMatrices/31006?competency=DotNET-Web&level=4>

2) Long term plan for the future

3) Performance issues in the current project

Using pagination no need to get all the records to local, based on the page number records will be pulled. in Offset(R2) will help to pull the records from SQL.

**in UI:** you can use Chrome ySlow tool to get the where exactly its lagging.

<http://yslow.org/>

**in Server** side apply the Caching techniques to hold the master data, instead of hitting the db again and again for the same.

4) How do you involve in code reviews.

Peer Code reviews and By Reviewer

5) Software Quality attributes

Coding stand, Unit testing, analysis, .

General software quality attributes include scalability, security, performance and reliability.

<https://pdfs.semanticscholar.org/2b1e/3ff0100c830853c9d9a30bf8989d4520436d.pdf>

6) Sonar Cube in your project

Sonar cube vs Sonar Link

[**SonarLint**](http://www.sonarlint.org/) lives only in the IDE (IntelliJ, Eclipse and Visual Studio). Its purpose is to give instantaneous feedback as you type your code. For this, it concentrates on what code you are adding or updating.

[**SonarQube**](http://www.sonarqube.org/) is a central server that processes full analyses (triggered by the various SonarQube Scanners). Its purpose is to give a 360° vision of the quality of your code base. For this, it analyzes all the source lines of your project on a regular basis.

<https://stackoverflow.com/questions/39828609/sonarqube-and-sonarlint-difference>

Sonar Cube vs Resharper

VeraCode (server) Local verification tool is GreenLight:

Veracode is a Central server that process full analysis,

7) Prefer executable specification vs Static document.

**Sandcastle XML Comments Guide** will take the XML object and prepare the HTML help content file.

<https://ewsoftware.github.io/XMLCommentsGuide/html/4268757F-CE8D-4E6D-8502-4F7F2E22DDA3.htm>

web api documentaion -- swagger

8) What are different kinds of documents present in your project?

Architecture diagram, Flow diagram, sequence diagram, mapping document,

9) UML diagrams and its basics and purpose (Unified Modeling Language)

The purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes in order to better understand, alter, maintain or document information about the system.

10) Difference between software architecture and software design

Software architecture is the process of converting software characteristics such as flexibility, scalability, reusability and security into a structured solution that meets the technical and the business expectations.

While software architecture is responsible for the skeleton and the high-level infrastructure of a software, the software design is responsible for the code level design such as, what each module is doing, the classes scope and the functions purposes, etc.

11) Performances fixes will comes under architectural or design changes

Performances fixes comes under design changes

12) Why do we need design patterns?

Design patterns are proven solutions which can help us to solve the day to day occurring issues in software development,

13) How to improve maintainability using design patterning name some of the DP you used

14) Anti patterns and how to avoid them

15) Estimation technique. Which one you used what other techniques you know

<https://www.sitepoint.com/3-powerful-estimation-techniques-for-agile-teams/>

Planning Poker: 0,1,2,3,5,8,13,21

T-Shirt Sizes: Extra small, small, Medium, Large, Extra Large, Double Extra Large

XS, S, M, L, XL, XXL.

Relative Mass Valuation: To Estimation larger number of backlog of stories that need to estimate all at once. One of the biggest advantages of agile estimation is that stories are estimated relative to each other, not on the basis of hourly or daily effort.

In initial agile process we used to use the Planning Poker, now team is well matured we are estimating the based on the fast experience (getting accurate results).

16) Why Fibonacci series why can't other techniques

<https://pm.stackexchange.com/questions/4251/why-would-teams-use-the-fibonacci-sequence-for-story-points>

Some teams also use powers of two, or have a scale like 1, 2, 5, 8, 20. The idea is that the larger the story is, the more uncertainty around it and the less accurate the estimate will be. Using the Fibonacci sequence helps teams to recognize this uncertainty, deliberately creating a lack of precision instead of wasting time trying to produce estimates that might also carry a false degree of confidence.

17) What is Technical debt? How do you remove it?

**Technical debt (**also known as design Debt or code Debt**)** reflects the implied (indirect) cost of additional rework caused by choosing an easy solution now instead of using the better approach that would take longer. If technical debt is not repaid, it can accumulate ‘interest’, making it harder to implement changes later on. Technical debt is not necessarily a bad thing, and sometimes technical debt is required to move projects forward.

<https://en.wikipedia.org/wiki/Technical_debt>

18) Customer insist to decrease the technical debt. How can you manage?

<https://dzone.com/articles/how-to-reduce-tech-debt-a-practical-experience-gui-1>

19) How do you introduce a junior to the project? If a new comer comes to the project how you knowledge transfer to that guy. What kind of tasks you will assign to the person.

Small features or bug fixes or all over the project you assign the tasks.

20) How can you guarantee the code quality in you project

We are doing code reviews, Peer reviews, we have a check point like DOD (Definition of done) we follow SOLID principals to increase the code quality. We are writing the unit test cases to increase the code coverage. Using tools like Sonar Qube, Re-sharper, Vera-Code,

By using the SonarQube

<https://simpleprogrammer.com/5-tips-code-quality/>

21) All team members will participate in the reviews. Whom you send your code to review.

No, Only selected people will do the code review. As a team monthly once we will discuss consolidated review comments to prevent in upcoming development same scenario’s.

22) Difference b/w continuous integrations vs continuous delivery

CI: Each merge request triggers an automated build and testing sequence for the given project.

Continuous Delivery: The continuous delivery process typically includes at least one manual step of approving and initiating a deploy to production.

23) Difference b/w continuous delivery vs continuous deployment

Continuous delivery required a manual intervension to deploy the code to Productions or any desired environment where as in Continuous Deployment is fully automated no manual intervention required.

<https://semaphoreci.com/blog/2017/07/27/what-is-the-difference-between-continuous-integration-continuous-deployment-and-continuous-delivery.html>

24) What are the bad unit tests can you tell some points

[Test passes but not testing the actual feature](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#skip-correct-feature)

[Testing irrelevant things](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#test-irrelevant)

[Testing multiple things in assertions](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#test-multiples)

[Test accessing the testee using reflection](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#reflection)

[Tests swallowing exceptions](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#swallow-exception)

[Test which depends on excessive setup](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#excess-setup-dependent)

[Test compatible to only developer's machine](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#machine-compatible)

[Test filling log files with load of texts](https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/#log-emission)

<https://howtodoinjava.com/best-practices/8-signs-of-bad-unit-test-cases/>

25) Diff b/w Integration test and unit test

|  |  |
| --- | --- |
| **Unit Testing** | **Integration Testing** |
| Unit testing is a type of testing to check if the small piece of code is doing what it is supposed to do. | Integration testing is a type of testing to check if different pieces of the modules are working together. |
| Unit testing checks a single component of an application. | The behavior of integration modules is considered in the Integration testing. |
| The scope of Unit testing is narrow, it covers the Unit or small piece of code under test. Therefore while writing a unit test shorter codes are used that target just a single class. | The scope of Integration testing is wide, it covers the whole application under test and it requires much more effort to put together. |
| Unit tests should have no dependencies on code outside the unit tested. | Integration testing is dependent on other outside systems like databases, hardware allocated for them etc. |
| This is first type of testing is to be carried out in Software testing life cycle and generally executed by developer. | This type of testing is carried out after Unit testing and before System testing and executed by the testing team. |
| Unit testing is not further sub divided into different types. | Integration testing is further divided into different types as follows:  Top-down Integration, Bottom-Up Integration and so on. |
| Unit testing is starts with the module specification. | Integration testing is starts with the interface specification. |
| The detailed visibility of the code is comes under Unit testing. | The visibility of the integration structure is comes under Integration testing. |
| Unit testing mainly focus on the testing the functionality of individual units only and does not uncover the issues arises when different modules are interacting with each other. | Integration testing is to be carried out to discover the the issues arise when different modules are interacting with each other to build overall system. |
| The goal of Unit testing is to test the each unit separately and ensure that each unit is working as expected. | The goal of Integration testing is to test the combined modules together and ensure that every combined modules are working as expected. |
| Unit testing comes under White box testing type. | Integration testing is comes under both Black box and White box type of testing. |

26) Diff b/w Mock and stub

A **Mock** is just testing behavior, making sure certain methods are called.

A **Stub** is a testable version of a particular object

27) Diff b/w scrum and Kanban

<https://www.guru99.com/scrum-vs-kanban.html>

28) Scrum master roles and responsibilities

<https://www.yodiz.com/blog/scrum-master-job-descriptions-and-responsibilities-in-agile-methodology/>

29) Scrum ceremonies

The scrum suggests three roles

1. Product Owner 2. Scrum Master 3. Development team (who executed the story Dev, QA, BA)

Four Ceremonies :

1. Sprint planning meeting 2. Daily Scrum 3. Sprint review meeting and 4. Sprint Retrospective meeting.

The three artifacts:

The product increment, Product backlog and sprint backlog.

30) Waterfall process where it is applicable.

31) How you are handling Nonfunctional requirements

Nonfunctional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability.

In every sprint we are taking the 80% work and 20% of buffer, If you don’t get any haddock works in this sprint we will consume these Non Functional activities like Unit testing, reducing code smiles, Performance issues, knows tech debts if any.   
32) How do you participate in communication with customer?

33) You got a direct call from the customer to change some requirements how do you deal in that situation

34) A task is assigned to you are not expert in that technology how do you overcome this situation

35) You gave a comments to your college but he is not agree on those comments what you do.

36) How do you get the information about new trends in the market?

37) New technologies to learn how much time you spend

38) Proud about your project

Need to check with mani & ravi on this.

39) Do you learn any new language or technology which is not related to your project in recent days?

40) How do you do refactoring the code where do you start. How you improve the design of existing code

41) What is clean code?

We need to follow the coding standards, code should be more readable it helps to developer to find the fix quickly, use verbs as function names, use nouns as variable names.

Code should follow the SOLID principles

Reduce the code smiles

If the other person affired to touch your code or review your code, that means it’s not a clean code.

42) Art of unit testing

Writing the testable code, the code should be unit testable,

43) Design patterns elements of reusable object oriented software

44) Pros and cons of a stored procedure

45) Synchronization primitives

46) .Net collections

<https://www.c-sharpcorner.com/article/collections-in-net/>

47) JS function invocation.

48) Objects to store data in C#

49) DB performance optimization and indexes

50) Aspect oriented programming

51) Clean code and code smells

52) ASP.NET MVC pipeline

53) Dependency injection vs service locator

54) What is a good unit and integration tests means

1. How do you manage if a resorce is on sick leave, but the next day he has to deliver story ?

2. you are about to go on leave, you got a priority 1 defect how you are going to handle ?

3. how you are estimate the task.

4. Client is keep on changing the requirements, how you are going to manage? which methodology you are going to use.---- Kanban

5. what are the project artifacts, where will we store these artifacts ?

6. what are all the graphs used in Agile ? Burndown chat, Effort variance

7. Generics

<http://www.tutorialsteacher.com/csharp/csharp-generics>

Generics introduced in C# 2.0. Generics allow you to define a class with placeholders for the type of its fields, methods, parameters, etc. Generics replace these placeholders with some specific type at compile time.

A generic class can be defined using angle brackets <>. For example, the following is a simple generic class with a generic member variable, generic method and property.

8. unit testing tools to get the coverage

Ncover vs opencover

Ncover – is commercial tool not a open source from Re-Sharp

OpenCover : is a open source, compatable to

9. Filters in webapi projects.

10. Serilog vs Log4net vs NLog

11. Coverlet, OpenCover and ReportGenerator

To get the code coverage in Visual studio.

12. 100% code coverage guarantees effective testing

13. LINQ queries.

14. MVC vs MVP

15. How you are giving the feedback to one user who is performing low.

16. SOAP vs Rest full services