AGILE ASSIGNMENT

Q1. what is agile methodology?

ANS:

The Agile software development methodology is one of the simplest and effective processes to turn a vision for a business need into software solutions. Agile is a term used to describe software development approaches that employ continual planning, learning, improvement, team collaboration, evolutionary development, and early delivery. It encourages flexible responses to change. It involves constant collaboration with stakeholders and continuous improvement at every stage. These methodologies use adaptive approaches and teamwork to focus on continuous improvement. Usually, agile software development consists of small, self-organizing teams of software developers and business representatives regularly meeting in-person throughout the software development life cycle. Agile favors a lightweight approach to software documentation and embraces—rather than resists—changes at any stage of the life cycle.

Q2. State Values and principles of Agile.

ANS:

The Agile Manifesto is comprised of four foundational values and 12 supporting principles which lead the Agile approach to software development. Each Agile methodology applies the four values in different ways, but all of them rely on them to guide the development and delivery of high-quality, working software.

1. Individuals and Interactions Over Processes and Tools

Valuing people more highly than processes or tools is easy to understand because it is the people who respond to business needs and drive the development process. If the process or the tools drive development, the team is less responsive to change and less likely to meet customer needs. Communication is an example of the difference between valuing individuals versus process. In the case of individuals, communication is fluid and happens when a need arises. In the case of process, communication is scheduled and requires specific content.

2. Working Software Over Comprehensive Documentation

Agile does not eliminate the lengthy documentation, but it streamlines it in a form that gives the developer what is needed to do the work without getting bogged down in minutiae. Agile documents requirements as user stories, which are sufficient for a software developer to begin the task of building a new function. The Agile Manifesto values documentation, but it values working software more.

3. Customer Collaboration Over Contract Negotiation

The Agile Manifesto describes a customer who is engaged and collaborates throughout the development process, making. This makes it far easier for development to meet their needs of the customer. Agile methods may include the customer at intervals for periodic demos, but a project could just as easily have an end-user as a daily part of the team and attending all meetings, ensuring the product meets the business needs of the customer.

4. Responding to Change Over Following a Plan

Traditional software development regarded change as an expense, so it was to be avoided. With Agile, changes always improve a project; changes provide additional value.

The twelve principles of agile development include:

1. Customer satisfaction through early and continuous software delivery – Customers are happier when they receive working software at regular intervals, rather than waiting extended periods of time between releases.

2. Accommodate changing requirements throughout the development process – The ability to avoid delays when a requirement or feature request changes.
3. Frequent delivery of working software – Scrum accommodates this principle since the team operates in software sprints or iterations that ensure regular delivery of working software.
4. Collaboration between the business stakeholders and developers throughout the project – Better decisions are made when the business and technical team are aligned.
5. Support, trust, and motivate the people involved – Motivated teams are more likely to deliver their best work than unhappy teams.
6. Enable face-to-face interactions – Communication is more successful when development teams are co-located.
7. Working software is the primary measure of progress – Delivering functional software to the customer is the ultimate factor that measures progress.
8. Agile processes to support a consistent development pace – Teams establish a repeatable and maintainable speed at which they can deliver working software, and they repeat it with each release.
9. Attention to technical detail and design enhances agility – The right skills and good design ensures the team can maintain the pace, constantly improve the product, and sustain change.
10. Simplicity – Develop just enough to get the job done for right now.
11. Self-organizing teams encourage great architectures, requirements, and designs – Skilled and motivated team members who have decision-making power, take ownership, communicate regularly with other team members, and share ideas that deliver quality products.
12. Regular reflections on how to become more effective – Self-improvement, process improvement, advancing skills, and techniques help team members work more efficiently.

Q3. Compare Agile with other Process models

ANS:

Agile	Waterfall
The project development lifecycle is divided into sprints.	There are many stages to the software development process.
It takes a step-by-step method.	The waterfall technique is a way of designing in a progressive order.
Agile methodologies are noted for their adaptability	Because waterfall is a structured software development technique, it may be extremely strict at times.
Agile may be thought of as a collection of many projects.	The development of software will be done as a single project.
Agile is a flexible strategy that enables modifications to project development that needs to be made even after the original planning is complete.	Once the project development begins, there is no way to change the specifications.
Because Agile methodology uses an iterative development approach, planning, development, prototyping, and other software development stages may occur several times.	In the Waterfall approach, all project development stages, such as design, development, and testing, are completed once.

Agile	Waterfall
After each sprint, the test plan is evaluated.	During the testing phase, the test strategy is seldom communicated.
Agile development is a method of software development in which needs are anticipated to change and evolve over time.	The technique is excellent for projects with specific needs and unanticipated modifications.
Testing is done simultaneously with software development in the Agile process.	The "Testing" step follows the "Build" phase in this technique.
Agile presents a product mentality in which the software product meets the requirements of its end users and adapts to their expectations.	This approach demonstrates a project mentality and focuses only on completing the job.
With Time & Materials or non-fixed financing, the Agile approach performs incredibly effectively. In fixed-price circumstances, it may create tension.	By obtaining a risk agreement at the start of the process, it is possible to reduce risk in firm-fixedprice contracts.
Small, focused teams with a high level of coordination and synchronization are preferred.	The ability of a team to coordinate and synchronize is severely hampered.
Almost every day of a project, the product owner and his team create requirements.	Prior to the start of a project, business analysts establish requirements.
The test team may easily participate in requirement changes.	Any change in requirements is tough for the test to commence.
During the SDLC process, the description of project specifics may be changed at any moment.	To use the waterfall software development technique, a detailed description is required.
As a consequence of the Agile Team's interchangeability, they work more quickly. Project managers are also unnecessary since the projects are controlled by the whole team.	Because the process is usually easy under the waterfall technique, the project manager is needed at every level of the SDLC.

Agile Model	RAD Model
The Agile paradigm discourages the creation of prototypes, preferring instead to focus on the methodical development of each incremental feature at the conclusion of each cycle.	The core concept of RAD is to create rapid and dirty prototypes that are later developed into production-quality code.

Agile Model	RAD Model
Agile initiatives logically split the solution down into features that are created and delivered in stages.	The RAD paradigm focuses on building all of an application's functionality by doing it poorly at first and then gradually improving the code over time.
After each iteration, the Agile team only shows the client finished work.	Customers may be shown screen mockups and prototypes by RAD teams, although these may be based on simplifications such as database lookup rather than real calculations.
Tiny projects are not ideal for the agile approach since it is difficult to break the project into small components that may be produced progressively.	When a corporation hasn't worked on a project that's nearly identical, it's difficult to employ the RAD paradigm since previous code can't be reused.

Incremental model vs. Agile model:

Agile model	Incremental model
Each incrementally delivered portion is built via an iteration following each time box in the Agile paradigm. The Agile model's core idea is to create agility by eliminating tasks that waste time and effort.	The software's requirements are broken down into various modules that may be built and supplied in stages. The basic features are created initially, then additional features are added to the program in consecutive releases.
In the Agile paradigm, an iteration's end date is set and cannot be amended. To finish that iteration on schedule, the development team may have to opt to minimize the supplied functionality.	There is no set deadline for completing the next iteration in the incremental development methodology.

Spiral Model vs. Agile Model:

Agile model	Spiral model
The Agile model's core idea is to create agility by eliminating tasks that waste time and effort.	The Spiral model's fundamental concept is risk management.
The Agile strategy emphasizes delivering an increment to the client after each Time-box, resulting in more frequent customer involvement.	The spiral model primarily deals with many types of unplanned hazards, however, there is minimal client engagement.
The agile paradigm is best suited for huge projects that can be broken down into tiny chunks and developed progressively over time.	The Spiral model is appropriate for projects that are vulnerable to a variety of hazards that are difficult to predict at the outset.

Agile model	Spiral model
Documentation is not used in the agile paradigm.	For the Spiral model, proper documentation is essential.

Agile model vs V-model:

Agile Model	V-Model
It is a software development model in which development and testing are concurrent.	It is also a software development model but development and testing are not concurrent.
It consists of different sprints.	It has two phases-Verification and Validation.
Testing is easy as compared to V-model.	Testing is hard as compared to the Agile model.
It consists of total of five phases.	It consists of five verification and five validation phases.
Communication is easy between end users, the development team, and the testing team.	The development team and testing team don’t interact much with end users.
Developers and testers are dependent on each other.	Developers and testers are independent.
It is iterative.	It is not iterative.
It is incremental.	It is not incremental.
It is suitable for small or large projects where the work needs to be completed iteratively.	It is suitable for large projects where the work needs to be completed sequentially.
In Agile model, the coding and testing phase gains more focus than the design phase.	In V-model, the design phase gains more focus than the implementation phase.

Agile model	Spiral model
In agile model, working software is available early as compared to V-model.	In V-model, working software takes time in comparison to the agile model.
It is a proactive model.	It is a reactive model.
It is more flexible compared to V-model.	This model is less flexible and more rigid compared to the agile model.
Relatively less testing cycle time than V-model because testing is done in parallel with the development.	Testing cycle time is more than the agile model.

Q.4 Elaborate when will you choose which process mode

ANS:

The software process model framework is specific to the project. Thus, it is essential to select the software process model according to the software which is to be developed. The software project is considered efficient if the process model is selected according to the requirements. It is also essential to consider time and cost while choosing a process model as cost and/ or time constraints play an important role in software development. The basic characteristics required to select the process model are project type and associated risks, requirements of the project, and the users.

One of the key features of selecting a process model is to understand the project in terms of size, complexity, funds available, and so on. In addition, the risks which are associated with the project should also be considered.

The selection criteria for various models are listed as tables below:

Project Type and Associated Risks	Waterfall	Prototype	Spiral	RAD	Formal Methods
Reliability requirements	No	No	Yes	No	Yes
Stable funds	Yes	Yes	No	Yes	Yes

Reuse components	No	Yes	Yes	Yes	Yes
Tight project schedule	No	Yes	Yes	Yes	No
Scarcity of resources	No	Yes	Yes	No	No

The requirements of the software should also be clearly understood before selecting any process model. Various other issues related to the requirements are listed in Table.

Requirements of the Project	Waterfall	Prototype	Spiral	RAD	Formal Methods
Requirements are defined early in SDLC	Yes	No	No	Yes	No
Requirements are easily defined and understandable	Yes	No	No	Yes	Yes
Requirements are changed frequently	No	Yes	Yes	No	Yes
Requirements indicate a complex System	No	Yes	Yes	No	No

The comprehensibility of the project increases if users are involved in selecting the process model. Various other issues related to the user's satisfaction are listed in Table.

User Involvement	Waterfall	Prototype	Spiral	RAD	Formal Methods
Requires Limited User Involvement	Yes	No	Yes	No	Yes
User participation in all phases	No	Yes	No	Yes	No

No experience of participating in similar projects	No	Yes	Yes	No	Yes

Q5. Understand different terms of Agile – Epics, Issues, Sprint , Scrum , Project Backlog, PBI, Daily Scrum.

Ans:

Epic: An epic is a big idea or feature that can be broken down into smaller user stories. Each user story, in turn, can be broken into manageable tasks.

Issues: Issues are the building blocks of any Jira project. An issue could represent a story, a bug, a task, or another issue type in our project.

Daily scrum: It's a daily meeting usually hosted by the Scrum master. Each member briefly talks about the following topics:

- What they plan to do today
- What they did yesterday
- Issues they have encountered

The Scrum master steps up to solve any impediments the Scrum team might have.

Sprint: sprint is a Scrum term and is usually a phrase used by Scrum teams. It's another term for an iteration. sprints maintain a uniform length (2-4 weeks) during the Agile development process.

Scrum: Scrum is an Agile methodology in which a team works in short bursts of work ranging from 2-4 weeks, called sprints. At the end of the sprint, they deliver the product to the customers, and in turn, the customers give the developers their feedback.

Project backlog: It’s a list of new product features, updates, bug fixes, etc. that are required by the user.

PBI: A Product Backlog Item (PBI) is a single element of work that exists in the product backlog. PBIs can include user stories, epics, specifications, bugs, or change requirements. The Product Owner of an Agile team compiles and prioritizes the product backlog, putting the most urgent or important PBIs at the top.

Q6. List different Agile methodologies

ANS:

1. Kanban
2. Scrum
3. Extreme Programming (XP)
4. Crystal
5. Dynamic Systems Development Method (DSDM)
6. Feature Driven Development (FDD)
7. Lean Software Development
8. Scaled Agile Framework (SAFe)

Q8. Which are various tools for Project Management, list them.

Ans:

Some important project management tools used in software projects are:

1. Gantt chart
2. PERT chart
3. Logic network
4. Product breakdown Structure(BSS)
5. Work Breakdown Structure
6. Resource histogram
7. Critical path analysis

Q7. Explore the Jira Tool and Demonstrate a case study and share print screen of same

ANS:

Jira Software is part of a family of products designed to help teams of all types manage work. Originally, Jira was designed as a bug and issue tracker. But today, Jira has evolved into a powerful work management tool for all kinds of use cases, from requirements and test case management to agile software development.