1. **Explain the following agile methodologies Scrum, Kanban, Extreme Programming.**

Scrum: Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Kanban: Kanban is a workflow management method designed to helps in visualizing the workflow, maximize efficiency and be agile. From Japanese, kanban is literally translated as billboard or signboard.

Extreme Programming: Extreme Programming (XP) is an agile software development framework that aims to produce higher quality software, and higher quality of life for the development team. XP is the most specific of the agile frameworks regarding appropriate engineering practices for software development.

2. **Who are the members of an agile team and what are their roles?**

Small Agile Team

Team lead :- This role, called “Scrum Master” in Scrum or team coach or project lead in other methods, is responsible for facilitating the team, obtaining resources for it, and protecting it from problems. This role encompasses the soft skills of project management but not the technical ones such as planning and scheduling, activities which are better left to the team as a whole.
Team members :-This role, sometimes referred to as developer or programmer, is responsible for the creation and delivery of a system. This includes modeling, programming, testing, and release activities, as well as others.
Product owner :-The product owner, called on-site customer in XP and active stakeholder in AM, represents the stakeholders. This is the one person responsible on a team (or sub-team for large projects) who is responsible for the prioritized work item list (called a product backlog in Scrum), for making decisions in a timely manner, and for providing information in a timely manner.
Stakeholders :-A stakeholder is anyone who is a direct user, indirect user, manager of users, senior manager, operations staff member, the "gold owner" who funds the project, support (help desk) staff member, auditors, your program/portfolio manager, developers working on other systems that integrate or interact with the one under development, or maintenance professionals potentially affected by the development and/or deployment of a software project.

Large Agile Team

When the size of an agile team gets to be around twenty or more you discover that you need to divide and conquer and take a “team of teams” approach. The typical strategy is to organize your larger team into a collection of smaller teams, and the most effective way to do so is around the architecture of your system. Each subteam should be responsible for one or more subsystems, enabling them to work as a small agile team responsible for delivering working software on a timely basis.

The additional roles on agile teams at scale include;

Architecture owner :- This person is responsible for facilitating architectural decisions on a sub-team and is part of the architecture owner team which is responsible for overall architectural direction of the project. The architecture owner leads their sub-team through initial architecture envisioning for their sub-systems and will be involved with the initial architecture envisioning for the system as a whole . Architecture owners are different than traditional architects in that they are not solely responsible for setting the architectural direction but instead facilitate its creation and evolution.

Integrator :- The subteams are typically responsible for one or more subsystems, and the larger the overall team generally the larger and more complicated the system being built. In these situations the overall team may require one or more people in the role of integrator who are responsible for building the entire system from its various subsystems. These people often work closely with the independent test team, if there is one, who will want to perform system integration testing regularly throughout the project.

3. **List 5 Project Management Tools**

   - Cross-functional resource management
   - Dashboards
   - Reporting
   - Gantt-Chart
   - Timesheets

4. **List the Phases of Software Development Life Cycle.**

   - Planning.
   - Analysis.
   - Design.
   - Development and Implementation.
   - Testing.
   - Maintenance.