1. **Product Burndown chart:** A product burndown chart shows how much work remains for the entire project. A product burndown chart collects a larger amount of data. The Scrum Burndown Chart is a visual measurement tool that shows the completed work per day against the projected rate of completion for the current project release. Its purpose is to enable the project is on the track to deliver the expected solution within the desired schedule.

**Sprint burn down chart:** A burndown chart shows the amount of work that has been completed in an epic or sprint, and the total work remaining. Burndown charts are used to predict the team's likelihood of completing their work in the time available.

1. **Impediment Log:** An impediment log is also known as an issue log or obstacle log. It is a tool or document used in Agile software development to track and manage obstacles, or anything that stops the team from getting work done and slows down the velocity. These impediments could be any issues, blockers or problems that hinder the team’s ability to complete their work efficiently.
2. **Prototyping:** In this technique, mock-up of the system is created early in understanding how the actual system will work. we try to understand the requirement based on the mock-ups. It helps in identifying the problems early. Tools used are Balsamiq, Axure. The prototype can be shared with the users and key stakeholders for feedback and refine the system design.
3. **Reverse Engineering:** It is understanding the existing system (as-is) and documenting the same. It can be of 2 types – Black Box Engineering (if we are not aware about the system) & White Box Engineering (if we are aware about the system). This technique is mostly used in migration projects.
4. **Observation:** Observing or shadowing the users in their working environment to understand their workflows and documenting the same.Beneficial when the users are not able to clearly explain their requirements. Two basic approaches are – Active Observation & Passive Observation.
5. **Velocity:** Velocity is a measure of the amount of work a team can tackle during a single sprint and is the key metric in Scrum. Velocity is calculated at the end of the Sprint by totalling the points for all fully completed User Stories. The calculation of velocity is performed by the development team itself, as they are responsible for estimating the effort required to complete each user story or backlog item. The velocity of the team is calculated by the scrum master. It can be calculated by - Story Point Estimation, Tracking complete Work, Summing Story points and Average velocity.

1. **Product Backlog:** In the product backlog all stories and all requirements are included that is needed to accomplish the project vision. Product Owner owns the product backlog. It includes requirements, defects, feature ideas, bug fix, and documentation. Everyone contributes to the product backlog. Release burndown metric is used. Estimation is done at user story level.
2. **Sprint Backlog:** In the sprint backlog anything needed to fulfil the sprint goal is included. The development team owns the sprint backlog. It is a subset of product backlog and defined as a priority by the product owner. Only the development team contributes to the sprint backlog. Sprint planning meeting is to refine the sprint backlog items. No changes are allowed to sprint backlog items once the sprint has started. Sprint burndown metric is used. Estimation is done at the activity or task level. Daily stand-up meeting discusses the sprint backlog in accordance with sprint goal.
3. **Product Grooming:** Product grooming is also known as backlog grooming, it is an important activity in Agile software development that involves reviewing, prioritizing, and refining the items in the product backlog. The goal of product grooming is to ensure that the backlog is well prepared, organized and for implementation in upcoming sprints. It is a collaborative effort involving the product owner, development team and other relevant stake holders. It is conducted by the following ways:
* Setting the context
* Backlog review
* Prioritization
* Refinement and Estimation
* Dependency Analysis
* Acceptance Criteria
* Backlog grooming meeting
1. **Scrum Master:** Scrum Master acts as a team coach and is responsible for maintaining the quality of the product. He will monitor the performance of the team within the sprint. All the issues raised by the team will be taken care by scrum master. He will facilitate the scrum process by resolving the issues and creating a self-organized environment for the team. Scrum Master reports to the top management about the efficiency of the team and the quality of the product.
2. **Product Owner:** The product owner will decide what should be there in the product and will be responsible for how the product has to be. He will regularly interact with the BA and the customers. He is responsible for the product vision. He can decide if a project will be accepted or rejected, whether development needs to be continued or not and whether the ship the product or not. Product Owner reports to the top management and the clients.
3. **Sprint Size:** The sprint size refers to the length or duration of a Sprint in scrum. A sprint is a time boxed period during which the development team works to deliver a potentially shippable product increment. The sprint size is determined during the project planning phase and typically ranges from two to four weeks, but it can vary depending on the project’s needs, complexity, and team dynamics.
4. **Scrum Size:** The Scrum team size refers to the number of individuals who collaborate to deliver the product increment in scrum. The scrum team is a self-organizing and cross-functional team, typically consisting of a Product Owner, a Scrum Master, and the development team. The scrum team size is small, ideally between 5 to 9 members, to enable effective communication, collaboration, and flexibility. However, there are no strict rules regarding team size, and it may vary depending on the specific project requirements and organization.
5. **Prioritization Technique:** Prioritization techniques are the techniques for queuing the requirements for the development process.

**MoSCoW:** It is a prioritization technique used for software development of placing the importance of each requirement.

MoSCoW means –

M – Must have this requirement to meet business needs

S – Should have this requirement if possible

C – Could have if it is not contradicting with the above two requirements

W – Would like to have next time

1. **Minimum Viable Product (MVP):** MVP is an early version of the product which has sufficient features to be used by the early customers who can provide feedback for the development of the product. The developers develop MVP to avoid lengthy and unnecessary work. It focuses on the core functionality that is necessary to deliver value to the user. It has minimum feature set—just enough to get the job done. It is a simplified design that is easy to use and understand.
2. **Camel-casing:** It is a naming convention in which words are combined without spaces and each word in the sequence (except for the first word) starts with an uppercase letter. It is commonly used in sequence diagram. By using camel casing, developers can create meaningful and readable names that are easier to understand and follow coding standards. It promotes consistency within the codebase and improves collaboration among team members. Eg: turnLeftAndThenRight();
3. **MVC Architecture:** Model-View Controller framework is an architecture that separates an application in three main logical interconnected components – **Model** – It represents the application's data and business logic and is responsible for retrieving, storing, and manipulating data. It notifies the view when changes occur in the data. **View** – It represents the user interface (UI) and displays data from the model to the user. It receives user input and forwards it to the controller. **Controller** – It acts as an intermediary between the view and the model and handles user input from the view. It updates the model or view based on the user's actions.
4. **SWOT Analysis:** SWOT Analysis is a model which is used to understand what are the influencing factors and how will they affect the project. SWOT is the acronym of Strengths, Weaknesses, Opportunities and Threats. Out of these, internal factors are strengths, weaknesses and external factors are opportunities, threats.
5. **Definition of Ready (DOR):** It defines the criteria that a user story must meet before it is considered ready to be included in a sprint. It ensures that the team has sufficient information and clarity about the user story, reducing the likelihood of misunderstanding or delays during the sprint. The user story has a clear and concise description and acceptance criteria. The user story is appropriately sized or estimated in story points or other relevant units. Any dependencies or external resources needed for the user story are identified and accessible.

**Definition of Done (DOD):** It outlines the criteria that a user story of any other backing item must meet to be considered complete and ready for release. It establishes a shared understanding of what it means for work to be considered as done and ensures that all necessary aspects such as quality, testing, and documentation are addressed.

1. **3-tier Architecture:** It is a software or system design pattern in which the developers will divide the project in three tiers and developers will work on each and every layer. **Application Layer:** This is the topmost layer and interacts with the user. This is the front end and it consists of GUI like Screens, pages and organisation specific business logic. User interacts with application layer and it sends request to the business logic layer then displays the response to the users. **Business Logic Layer:** This is the middle layer of the architecture. It contains of the functionality of the application. It has all the reusable components which changes frequently. This layer executes the logic required for the application to function. **Database Layer:** This layer is responsible for storing and managing the data. All the components connecting to databases will be in this layer. This layer ensures the data security.