1. **Elicitation techniques:**

Elicitation is the process of gathering requirements, needs, and information from stakeholders to understand the business problem, opportunities, and the desired solution. It involves techniques that help to collect accurate and relevant data that will guide the project.

1. **Business Analysis:**

**Business Analysis** is the practice of identifying business needs, defining solutions to business problems, and ensuring that the requirements of a project or initiative align with the organization’s strategic goals. Business analysis helps organizations improve processes, systems, and software, ensuring they achieve their objectives in a more efficient and effective way.

1. **SDLC Methodologies:**

SDLC (Software Development Life Cycle) methodologies are structured approaches to software development that define the processes, phases, and practices involved in creating software applications. These methodologies aim to streamline development, ensure quality, and meet customer requirements efficiently. Here are some common SDLC methodologies:

1. **Sprint:**

A **sprint** is a time-boxed period, usually lasting 1 to 4 weeks, within which a Scrum team works to complete a specific set of work items (user stories or tasks) from the product backlog. The goal of each sprint is to deliver a potentially shippable product increment or achieve a defined outcome.

1. **Daily Stand-up** :

A **Daily Stand-up** is a key practice in **Agile** methodologies, particularly within **Scrum**. It is a short, time-boxed meeting where team members share updates on their progress, discuss any obstacles they're facing, and align on the next steps.

1. **SWOT Analysis**:

**SWOT Analysis** is a strategic planning tool used to assess the **Strengths, Weaknesses, Opportunities**, and **Threats** related to a business, project, product, or even an individual. It helps organizations understand internal and external factors that may impact decision-making, strategy formulation, and goal setting.

1. **Gap Analysis**

Gap analysis is a strategic tool used to compare the current state of an organization, process, or system to its desired future state. It identifies gaps between "where we are" and "where we want to be" and provides actionable insights to bridge those gaps.

1. **Epic:**

An **epic** is a large user story or a group of related user stories that share a common business goal but cannot be delivered in a single sprint or development cycle. Epics are broken down into smaller, more manageable user stories or tasks. They are often used to organize the product backlog in agile methodologies.

1. **Stakeholder:**

Stakeholder is any individual, group, or organization that has an interest in, is affected by, or can influence a project, process, or business outcome. Stakeholders can include internal team members, external parties, and end-users.

1. **The** **Agile Model**   
   agile is an iterative and incremental approach. Development is done in small, manageable units (called iterations or sprints), with frequent collaboration and feedback from stakeholders.
2. **MVC Architecture**:   
   (Model-View-Controller) is a software design pattern commonly used for developing user interfaces, particularly in web applications. It separates an application into three interconnected components, allowing for modularization and improving maintainability. Each component is responsible for a specific role within the application, and their interaction ensures efficient handling of user requests, data, and display.
3. **Requirement analysis:**  
   **Requirement Analysis** is a crucial phase in the **business analysis** and **software development lifecycle**. It involves the process of defining, documenting, and managing the needs and expectations of stakeholders, ensuring that the final solution aligns with the business objectives and requirements. It helps to bridge the gap between what the stakeholders want (business needs) and how these needs will be met (system or product functionality).
4. **Brainstorming**:

**Brainstorming** is a creative problem-solving technique that encourages the generation of ideas and solutions through group discussions or individual thinking. The goal of brainstorming is to generate a wide range of ideas without judgment, focusing on quantity over quality initially. Once a large pool of ideas is generated, the most promising or feasible ones can be refined and developed further.

1. **Waterfall** **Model**:   
   The **Waterfall Model** is one of the earliest and simplest Software Development Life Cycle (SDLC) models. It's a **linear and sequential** approach, where each phase of the development process is completed before the next one begins, and there’s little to no overlap between phases.
2. **Prototyping**:   
   **Prototyping** is an iterative software development technique used to visualize, test, and refine a solution before it is fully built. It involves creating a working model (or prototype) of the system or product, which serves as an early version of the final product. Prototyping allows stakeholders and users to interact with the product, provide feedback, and suggest changes, making it a highly valuable approach in software and system development.
3. **Root Cause Analysis :**

**Root Cause Analysis (RCA)** is a problem-solving technique used to identify the underlying cause(s) of an issue or problem. Rather than just addressing the symptoms, RCA seeks to understand the root cause to prevent the issue from recurring.

1. **Use case diagram:**

A **Use Case Diagram** is a visual representation of the functional requirements of a system, showing how users (or "actors") interact with the system and what actions they can perform. It's commonly used in **Unified Modeling Language (UML)** to capture and clarify system behavior from an end-user perspective. It helps in understanding the system's functionality and is often used during the **requirements gathering** phase of system design.

1. **3-Tier Architecture:**

**3-Tier Architecture** is a software design pattern that divides an application into three distinct layers or tiers, each with a specific role. It provides a way to separate concerns, improve scalability, and simplify maintenance. This architecture is widely used in both web and enterprise application development.

1. **Requirements Engineering**:   
   **Requirements Engineering (RE)** is a critical phase in the software development lifecycle that focuses on defining, documenting, and managing the requirements of a system. It ensures that the end product aligns with the expectations and needs of stakeholders, including end users, customers, and business units. RE aims to capture, analyze, validate, and communicate what a system should do and the constraints it must operate within.
2. **Challenges face by Business analysis :**

Unclear or Evolving Requirements, Stakeholder Conflicts, Lack of Engagement, Communication Barriers, Limited Access to Key Information, Changing Business Environment, Scope Creep, Inadequate Tools and Resources, Lack of Clear Metrics for Success, Balancing Business and Technical Perspectives.