1. **Requirement and Types of requirements:**

Requirements are nothing but the needs of the client. This is also called as problems faced by client or opportunity for a solution. This basic need will transform into solution through various stages of SDLC and will solve any problems faced by the client and will give a new opportunity to develop business or growth. There are 4 basic types of requirements- Business requirements, Stakeholder requirements, Solution requirements (Functional and Non-functional requirements) and Transition requirements.

1. **Solution Requirements:**

Solution requirements are the characteristics of a solution that meets business and stakeholder requirements. They are developed and defined through requirement analysis. They are classified into 2 categories- Functional and Non-functional requirements. Functional requirements are the capabilities or the features that system will be able to perform to meet the business objectives. Non-functional requirements are the quality or supplementary requirements which describe the environmental conditions under which solution must remain effective.

1. **Business Process Modelling:**

Business Process Modelling is technique to represent the business processes which are collection of activities to generate an output from the input using the resources to meet the goal to provide some value to the customer or end user. Typically, business process modelling will have goal, inputs, outputs, activities, resources and value it brings to the customer.

1. **Reasons for Project Failure:**

Project will be failed due to various reasons as below.

1. Improper requirement gathering- Client and BA are responsible
2. Continuous change in requirements- Client is responsible
3. Lack of user involvement- Client and BA are responsible
4. Lack of executive support- PM & Management are responsible
5. Unrealistic expectations- Client is responsible
6. Improper planning- PM & Management are responsible
7. **Assumptions and Constraints of a project:**

Assumptions are conditions which are believed to be true or expected to be constant for the successful execution of the project. These are not proven or guaranteed but taken as granted or valid to execute the project. Expectation of correct requirements from stakeholders, timely approvals and availability of resources are examples.

Constraints are the limitations or restrictions that define the boundary within which project must be completed. These are non-negotiable factors that must be adhered to during the completion of the project. Limit in cost, time, scope and resources are examples of constraints.

1. **Gantt Charts:**

Gantt charts are generally used by Project Managers to show visually the rough estimation of timelines and the tasks/resources to plan and track the project efficiently. MPP can be used to create these charts. There will be start and end date against tasks/resources in rows and time (week/month) in columns.

1. **Dos and Don’ts of BA:**

**Don’ts:** Never resist the change and say ‘No’ to the stakeholders, don’t assume the requirements and anything in GUI, don’t accept anything by ‘default’, never try to give solutions straight away during requirement gathering

**Dos:** Question the existence of existence to understand the roots of needs, meet client with plain mind with no assumptions, listen carefully when client explain their problem without any interruption, try to extract the leads to solution from client itself.

1. **Challenging Areas of BA:**

Unavailability of stakeholders during important meetings, handling scope creep and change management- change requests of stakeholders, obtaining sign-off of documents, lack of domain knowledge and training, translating functional requirements to technical solution, handling the bugs during UAT and people management.

1. **Conflict Management- Thomas Kilmann technique:**

Conflict management is an important thing in project management. Identify the conflict, discuss the details, agree with the root cause, check for possible solutions and negotiate the solution to avoid future conflicts are the steps for conflict management. Thomas Killman technique is useful to categorize the stakeholders based on their assertiveness (x-axis) and co-operation (y-axis) in a plot as competing, avoiding, compromising, collaborating, accommodating stakeholders. This will help BA to take appropriate techniques to reach consensus with those stakeholders.

1. **V-model, an SDLC model:**

V-model is a type of SDLC model which is sequential in nature with phases one after the other. Each phase has specific deliverables and a review process. Left side of V represents the verification such as requirement gathering, analysis, design and detailed design. Right side of V represents validation such as Unit Testing, Component Testing, Integration Testing, System Integration Testing (SIT) before UAT. This model is best suited for smaller projects where requirements are very well understood.

1. **Iterative-RUP (Rational Unified Process) model:**

It is an iterative software development process created by Rational Software Corporation which was part of IBM. RUP is based on a set of building blocks or content elements such as roles (who), work products (what) and tasks (how). Each iteration has tasks which are categorized into 9 disciplines- 6 engineering (Business Modelling, Requirement, Analysis & Design, Implementation, test and Deployment) and 3 supporting disciplines (Configuration & Change Management, Project Management and Environment). There all are distributed in 4 project life cycles- Inception, Elaboration, Construction and Transition.

1. **Evolutionary- Spiral model:**

This model gives more emphases on risk analysis. There are 4 phases in this model- Planning, Risk Analysis, Engineering and Evaluation. Software repeatedly passes through these phases in iterations or spirals. The baseline spiral starts from planning phase. Requirements are gathered during planning phase. In the risk analysis phase, a process is undertaken to identify the risks and alternate solutions and a prototype is produced at the end. Software is produced in the engineering phase. Customers are allowed to evaluate the output of the project till date in the evaluation phase before it enters into the next spiral. Angular component of the spiral represents progress and radius represents cost. This is best suited for large and mission critical projects where software is produced early.

1. **OOA- Object Oriented Approach:**

Object Oriented Approach is a method that emphasizes designing IT solutions by modelling real world entities as objects, which includes both data and behaviors. OOA involves identifying and defining key entities, their identity, attributes and behaviors and their interactions through UML diagrams such as use case, class, activity and sequence diagrams. This method promotes modularity, reusability and scalability.

1. **Implementation of OOA:**

Implementing OOA is based on 4 concepts. **Abstraction** means considering what is required and ignoring what is not required. That is to filter out an object’s properties until just the ones we need are left. **Encapsulation** is information and complexity hiding technique to hide internal data structure and working. **Inheritance** is the property that a child class inheriting the properties of parent class. **Polymorphism** represents the property that a single instruction can do the multiple operations because of the same name in different classes.

1. **Relationships in OOA:**

Relationships in OOA exists either between classes or between objects, but not between a class and an object. 4 types are there. **Association** is a relationship having 4 kinds as unary, binary, multiplicity and reflexive. **Generalization** is a relation exists between generalized and specialized class. **Aggregation** is a ‘part’ and its ‘whole’ relationship. The ‘part’ remains even if the ‘whole’ dies. Home computer system is an example. **Composition** is a strong type of aggregation. Here the ‘part’ dies along with ‘whole’. Coffee table (table top, legs) is an example.

1. **Steps to draw use case diagram from business requirements:**
2. Identify the functional requirements (from business process or functionality), don’t focus on technical information like names of systems, architecture, DB names etc.
3. Write all sequence of actions
4. Differentiate actors and information
5. Identify essential and supporting use cases from actions
6. Find out which actor is performing these actions
7. Identify modules with respect to functionality or usage
8. Select the relationship between actor and use cases appropriately.
9. **Deriving Test Cases using Use cases:**

Use case diagram and description document will provide the basic flow, alternate flow and exceptional flow of all the functionalities to be implemented which are identified from business requirements. Now try to identify the test scenarios for each use cases and these flows. Test cases are derived from these test scenarios by considering various conditions and write 3-5 test cases for each scenario. Test data should be available in the testing environment to do the actual testing.

1. **Three Tier Architecture of an IT application:**

Each IT application will have 3 layers as Application Layer, Business Logic Layer and Data Layer. Application layer will have all the screens and pages in GUI, validations, company specific logic and functionalities. Business Logic Layer will have all reusable components such as industry specific logic, frequently changing components, governing body rules and regulations and compliance related components. Data Layer consists of data base components connecting to a DB having Db designs which includes DB schema (objects and meta data) and ER diagrams.

1. **MVC Architecture and rules:**

To identify classes from use case, MVC rules can be applied. MVC stands for Model, View and Controller. Model class is entity class and which represents the data and governing rules to access the data. View Class is a boundary class and which represents the presentation of the application. Controller class is responsible for intercepting the requests from view and passes it to the model for action. After the action, it again directing back to view to user. CC is working based on user’s command. Three MVC rules are there- Use case will result in a controller class, actor will result in an entity class and combination of one actor and a use case will result in a boundary class.

1. **Guidelines to place MVC classes in 3 Tier Architecture:**
2. Place all entity classes in DB layer
3. Place controller class in Application Layer
4. Place boundary class associated with Primary actor in Application Layer
5. Place boundary class associated with governing body influence or reusability in the Business Logic Layer.