🔥 Requirement Analysis Questions and Answers

Question 1:

What is Requirement Analysis in Software Testing?

Answer:

1. Requirement Analysis is the process of understanding what needs to be built and tested.

2. It helps QA align with business expectations and user needs.

3. It acts as a foundation for writing test cases and scenarios.

4. Without it, testing becomes guesswork instead of validation.

5. It ensures clarity before execution starts.

👉 **Example:** Understanding login rules before writing test cases.

💡 **Good testing starts with understanding, not execution.**

Question 2:

What is the main goal of Requirement Analysis?

Answer:

1. To understand requirements clearly and completely.

2. To remove ambiguity and confusion.

3. To identify missing or unclear details.

4. To prepare accurate and meaningful test scenarios.

5. To reduce defects in later stages.

💡 **Clarity today prevents defects tomorrow.**

Question 3:

What happens if requirements are misunderstood?

Answer:

1. Incorrect functionality gets developed.

2. Test cases become invalid or incomplete.

3. Bugs increase in production.

4. Rework and delays occur.

5. User experience and trust get impacted.

👉 **Example:** Wrong validation implemented due to unclear requirement.

💡 **A small misunderstanding can create big failures.**

Question 4:

Why is Requirement Analysis important?

Answer:

1. It ensures correct understanding of system behavior.

2. It reduces defects and rework.

3. It improves communication between teams.

4. It helps design better test cases.

5. It builds a strong quality foundation.

💡 **Strong analysis = strong product.**

Question 5:

Can testing start without proper requirements?

Answer:

1. Yes, but it becomes risky and unreliable.

2. Testing will depend on assumptions.

3. Coverage will be incomplete.

4. Critical bugs may be missed.

5. Overall product quality will suffer.

💡 **Testing without requirements is blind testing.**

Question 6:

What risks arise from poor Requirement Analysis?

Answer:

1. Missing important scenarios.

2. Incorrect system behavior.

3. High defect leakage to production.

4. Increased cost and time.

5. Poor user experience.

💡 **Poor analysis leads to expensive mistakes.**

Question 7:

What are functional requirements? Give real examples

Answer:

1. Functional requirements define what the system should do.

2. They describe features and user interactions.

3. They are directly testable.

4. They define system behavior.

5. They are core to application functionality.

👉 **Example:** User should be able to login using valid credentials.

💡 **Functional requirements define system behavior.**

Question 8:

How do you validate functional requirements?

Answer:

1. Map requirements with test cases.

2. Execute positive and negative scenarios.

3. Validate expected vs actual results.

4. Check business logic accuracy.

5. Ensure complete coverage of flows.

💡 **Validation ensures correctness, not assumptions.**

Question 9:

What are non-functional requirements?

Answer:

1. They define how the system performs.

2. They include performance, security, usability.

3. They focus on system quality attributes.

4. They impact user experience.

5. They ensure system stability and reliability.

👉 **Example:** System should load within 2 seconds.

💡 **Non-functional defines experience, not just behavior.**

Question10:

Difference between functional & non-functional requirements?

Answer:

Functional and Non-functional requirements are two fundamental parts of any system, but they serve completely different purposes. Functional requirements define **what the system should do**, meaning the actual features and operations of the application. Non-functional requirements define **how the system should perform**, meaning the quality, performance, and behavior of the system under different conditions. Both are equally important — functional ensures the system works, while non-functional ensures the system works well.

Functional Requirements	Non-Functional Requirements
1. Define what the system does	1. Define how the system performs
2. Focus on features and functionality	2. Focus on quality attributes
3. Directly related to user actions	3. Include performance, security, usability
4. Easier to test using test cases	4. Harder to test (need tools/environment)
5. Based on business logic	5. Based on system behavior and constraints
6. Example: Login, Signup, Payment	6. Example: Response time, load handling
👉 These answer the question: **“System kya karega?”**	👉 These answer the question: **“System kaise perform karega?”**

👉 **Real Example:**

Functional: User should be able to login with valid credentials.

Non-functional: Login should happen within 2 seconds and support 1000 users at the same time.

💡 **Functional makes the system work, Non-functional makes it reliable, fast, and usable.**

Question 11:

Which is harder to test, functional & non-functional requirements?

Answer:

1. Non-functional requirements are harder to test.

2. They require tools and environment setup.

3. They involve performance and scalability.

4. They are not always clearly defined.

5. They need deeper analysis compared to functional testing.

👉 **Example:** Testing login is easy (functional), but testing system under 1000 users load is complex.

💡 **Hard problems reveal real system strength.**

Question 12:

Why do teams ignore non-functional requirements?

Answer:

1. Teams focus more on functionality.

2. Time constraints and deadlines.

3. Lack of tools or expertise.

4. Requirements are not clearly defined.

5. Impact is underestimated.

👉 **Example:** System works fine, but crashes under high traffic because performance was ignored.

💡 **Ignored performance becomes production issue.**

Question 13:

What is a requirement document?

Answer:

1. It is a document that defines system requirements.

2. It includes features, behavior, and constraints.

3. It acts as a reference for development and testing.

4. It ensures all stakeholders are aligned.

5. It guides the entire project lifecycle.

👉 **Example:** Document describing login, signup, and payment flow.

💡 **Requirement document is the project blueprint.**

Question 14:

Types of requirement documents?

Answer:

1. BRD (Business Requirement Document)

2. SRS (Software Requirement Specification)

3. Use Case Documents

4. User Stories

5. Functional Specification Documents

👉 **Example:** BRD explains business need, SRS explains system behavior.

💡 **Different documents serve different clarity levels.**

Question 15:

Who prepares requirement documents?

Answer:

1. Business Analysts (BA) mainly prepare them.

2. Product Owners contribute requirements.

3. Stakeholders provide business needs.

4. Developers may add technical details.

5. QA reviews and validates them.

👉 **Example:** BA writes SRS after discussing with client and stakeholders.

💡 **Requirement is a collaboration, not a single role.**

Question 16:

What is SRS (Software Requirement Specification)?

Answer:

1. SRS is a detailed document of system requirements.

2. It defines functional and non-functional requirements.

3. It acts as a contract between business and development.

4. It guides development and testing.

5. It ensures clarity and completeness.

👉 **Example:** SRS defines login rules, validation, and performance expectations.

💡 **SRS converts business idea into technical clarity.**

Question 17:

What should an ideal SRS include and Who writes SRS?

Answer:

1. Functional requirements.

2. Non-functional requirements.

3. System flow and use cases.

4. Constraints and assumptions.

5. It is written mainly by Business Analyst.

👉 **Example:** SRS includes login flow, validations, performance, and security rules.

💡 **A good SRS leaves no room for confusion.**

Question 18:

How do you review SRS?

Answer:

1. Check for clarity and completeness.

2. Identify missing scenarios.

3. Validate business logic.

4. Ask questions for ambiguity.

5. Ensure testability of requirements.

👉 **Example:** Asking “What happens on invalid input?” if not defined.

💡 **Good QA questions improve requirements.**

Question 19:

What is BRD (Business Requirement Document)?

Answer:

1. BRD defines business needs and goals.

2. It focuses on what business wants.

3. It is high-level document.

4. It is written before SRS.

5. It helps understand business perspective.

👉 **Example:** BRD explains why an e-commerce system is needed.

💡 **BRD defines “why”, not “how”.**

Question20:

Difference between BRD and SRS?

Answer:

BRD (Business Requirement Document) and SRS (Software Requirement Specification) are two critical documents in the software development lifecycle, but they serve very different purposes. BRD focuses on the **business perspective** — it explains why the system is needed and what problem it will solve. It is written in a simple, non-technical language so that business stakeholders can understand it easily. SRS, on the other hand, focuses on the **system perspective** — it explains how the system will work in detail. It translates business requirements into technical specifications that developers and testers can use. In simple terms: 👉 BRD = **What and Why** 👉 SRS = **How**

BRD (Business Requirement Document)	SRS (Software Requirement Specification)
1. Focuses on business needs and objectives	1. Focuses on system behavior and functionality
2. Written in simple, non-technical language	2. Written in detailed and structured format
3. Created by Business Analyst after stakeholder discussions	3. Created by Business Analyst / System Analyst
4. Defines the purpose and goal of the system	4. Defines functional and non-functional requirements
5. High-level document — does not go into technical details	5. Low-level detailed document
6. Used by stakeholders, clients, and management	6. Used by developers, testers, and technical teams
7. Answers: **Why system is needed?**	7. Answers: **How system will work?**
👉 Example thinking: “We need an online shopping platform to increase sales.”	👉 Example thinking: “User can login, add items to cart, apply discount, and complete payment.”

👉 **Real Example:**

BRD: “We need an e-commerce system where users can buy products online.”

SRS: “User can register, login, search products, add to cart, apply coupons, and make payment using different methods.”

💡 BRD gives vision, SRS gives execution. 💡 BRD is for business understanding, SRS is for system implementation.

Question 21:

Who writes BRD?

Answer:

1. Business Analyst (BA) primarily writes BRD.

2. Inputs are taken from stakeholders and clients.

3. Product Owner may contribute requirements.

4. BA ensures business goals are clearly defined.

5. It is reviewed by stakeholders before approval.

👉 **Example:** BA collects client needs and writes BRD for an e-commerce system.

💡 **BRD is written by BA but driven by business needs.**

Question 22:

What is a use case?

Answer:

1. A use case describes how a user interacts with the system.

2. It defines step-by-step user actions.

3. It focuses on real-world scenarios.

4. It helps understand system flow.

5. It is used for designing test scenarios.

👉 **Example:** User logs in → selects product → adds to cart → makes payment.

💡 **Use case = real user journey.**

Question:

Difference between use case & test case?

Answer:

Use Case and Test Case are closely related, but they serve different purposes in the software development and testing process. A **Use Case** describes how a user interacts with the system in a real-world scenario. It focuses on the flow of actions from the user's perspective and helps in understanding how the system should behave. A **Test Case**, on the other hand, is used to verify whether the system behaves correctly. It contains detailed steps, expected results, and conditions to validate the functionality. In simple terms: 👉 Use Case = **Understanding the flow** 👉 Test Case = **Validating the flow**

Use Case	Test Case
1. Describes user interaction with system	1. Validates system functionality
2. Focuses on real-world scenarios	2. Focuses on correctness and validation
3. High-level representation of flow	3. Detailed step-by-step instructions
4. Used during requirement analysis	4. Used during testing phase
5. Helps understand system behavior	5. Includes expected results
6. Does not contain expected results	6. Helps find defects
👉 Answers: **User kya karega?**	👉 Answers: **System sahi kaam kar raha hai ya nahi?**
Example thinking: User logs in → selects product → makes payment	Example thinking: Enter valid username → expect login success Enter wrong password → expect error message

👉 **Real Example:**

Use Case: User logs into system, searches product, adds to cart, and completes payment.

Test Cases:
– Login with valid credentials → success
– Login with invalid password → error
– Add product to cart → product added
– Payment with valid card → success

💡 **Use Case shows the journey, Test Case checks the correctness. 💡 Use Case is story, Test Case is verification.**

👉

💡

👉

💡

👉

💡

👉

💡

Question 28:

What is requirement gathering? Who is involved in it?

Answer:

1. Requirement gathering is the process of collecting business needs.

2. It involves understanding user expectations.

3. It defines what system should achieve.

4. It is the first step of SDLC.

5. It sets the project direction.

👉 **Example:** Client explains need for an online booking system.

💡 **Good requirements start with good communication.**

Question 29:

What techniques are used for requirement gathering?

Answer:

1. Interviews with stakeholders.

2. Questionnaires and surveys.

3. Workshops and meetings.

4. Observation of users.

5. Document analysis.

👉 **Example:** BA conducts interview with client to understand system needs.

💡 **Right questions lead to right requirements.**

Question 30:

What is requirement validation? Who validates and when is it done?

Answer:

1. Requirement validation ensures correctness of requirements.

2. It checks completeness and clarity.

3. It is done before development starts.

4. BA, QA, and stakeholders validate it.

5. It avoids future defects.

👉 **Example:** Reviewing requirement document before development begins.

💡 **Validate early to avoid costly mistakes.**

Question 31:

What is requirement prioritization?

Answer:

1. It is the process of ranking requirements.

2. It is based on business importance.

3. It helps focus on critical features.

4. It supports release planning.

5. It improves resource management.

👉 **Example:** Login feature prioritized over UI improvements.

💡 **Not everything is equally important.**

Question 32:

Who decides priority?

Answer:

1. Product Owner decides priority.

2. Business stakeholders influence it.

3. BA provides input.

4. QA may suggest based on risk.

5. Final decision is business-driven.

👉 **Example:** Payment feature prioritized for release.

💡 **Priority = business value.**

Question 33:

What is ambiguity in requirements?

Answer:

1. Ambiguity means unclear or confusing requirement.

2. It can be interpreted in multiple ways.

3. It creates misunderstanding.

4. It leads to incorrect implementation.

5. It must be clarified early.

👉 **Example:** “Fast response time” without defining exact time.

💡 **Ambiguity creates defects.**

Question 34:

How do you identify ambiguity? Give real-time example

Answer:

1. Look for unclear words.

2. Check missing details.

3. Ask questions.

4. Analyze multiple interpretations.

5. Validate with stakeholders.

👉 **Example:** “User should get notification” → QA asks when and how.

💡 **Questions remove confusion.**

Question 35:

What is incomplete requirement? How do you handle it?

Answer:

1. Incomplete requirement lacks necessary details.

2. It creates gaps in understanding.

3. QA identifies missing parts.

4. Clarify with BA or stakeholders.

5. Document assumptions carefully.

👉 **Example:** Login defined but no error handling mentioned.

💡 **Missing details create hidden defects.**

Question 36:

What is conflicting requirement? Give example and how to resolve it?

Answer:

1. Conflicting requirement has contradictions.

2. It creates confusion in implementation.

3. It must be resolved before development.

4. Discuss with stakeholders.

5. Final decision should be documented.

👉 **Example:** One doc says password mandatory, another says optional.

💡 **Conflict resolved early saves rework.**

Question 37:

What is requirement traceability? Why is it important?

Answer:

1. Traceability links requirements with test cases.

2. It ensures all requirements are covered.

3. It helps track changes.

4. It improves visibility.

5. It ensures complete testing.

👉 **Example:** Mapping login requirement with test cases.

💡 **Traceability ensures nothing is missed.**

Question 38:

What tools you used for requirement traceability?

Answer:

1. Jira

2. Excel

3. TestRail

4. Azure DevOps

5. ALM tools

👉 **Example:** Mapping requirements and test cases in Jira.

💡 **Tools help track, but thinking ensures coverage.**

Question 39:

What is RTM (Requirement Traceability Matrix), what fields does it contain and who maintains it?

Answer:

1. RTM maps requirements to test cases.

2. Fields: Requirement ID, Test Case ID, Status.

3. It ensures full coverage.

4. QA usually maintains it.

5. It helps in tracking progress.

👉 **Example:** Login requirement linked to all login test cases.

💡 **RTM = coverage visibility.**

Question 40:

What is gap analysis and when is it done? Give example

Answer:

1. Gap analysis identifies missing requirements.

2. It is done during requirement review.

3. It compares expected vs actual.

4. It helps identify missing logic.

5. It improves requirement quality.

👉 **Example:** Login feature missing forgot password flow.

💡 **Gaps reveal hidden risks.**

Question 41:

What is feasibility analysis and who performs it?

Answer:

1. Feasibility analysis checks if project is possible.

2. It evaluates technical and business viability.

3. It is done before development.

4. BA, architects, and stakeholders perform it.

5. It reduces project risks.

👉 **Example:** Checking if system can handle 1M users.

💡 **Feasibility decides possibility.**

Question 42:

What are the types of feasibility analysis?

Answer:

1. Technical feasibility.

2. Economic feasibility.

3. Operational feasibility.

4. Legal feasibility.

5. Schedule feasibility.

👉 **Example:** Checking cost, time, and technical capability before project.

💡 **Feasibility checks reality before execution.**