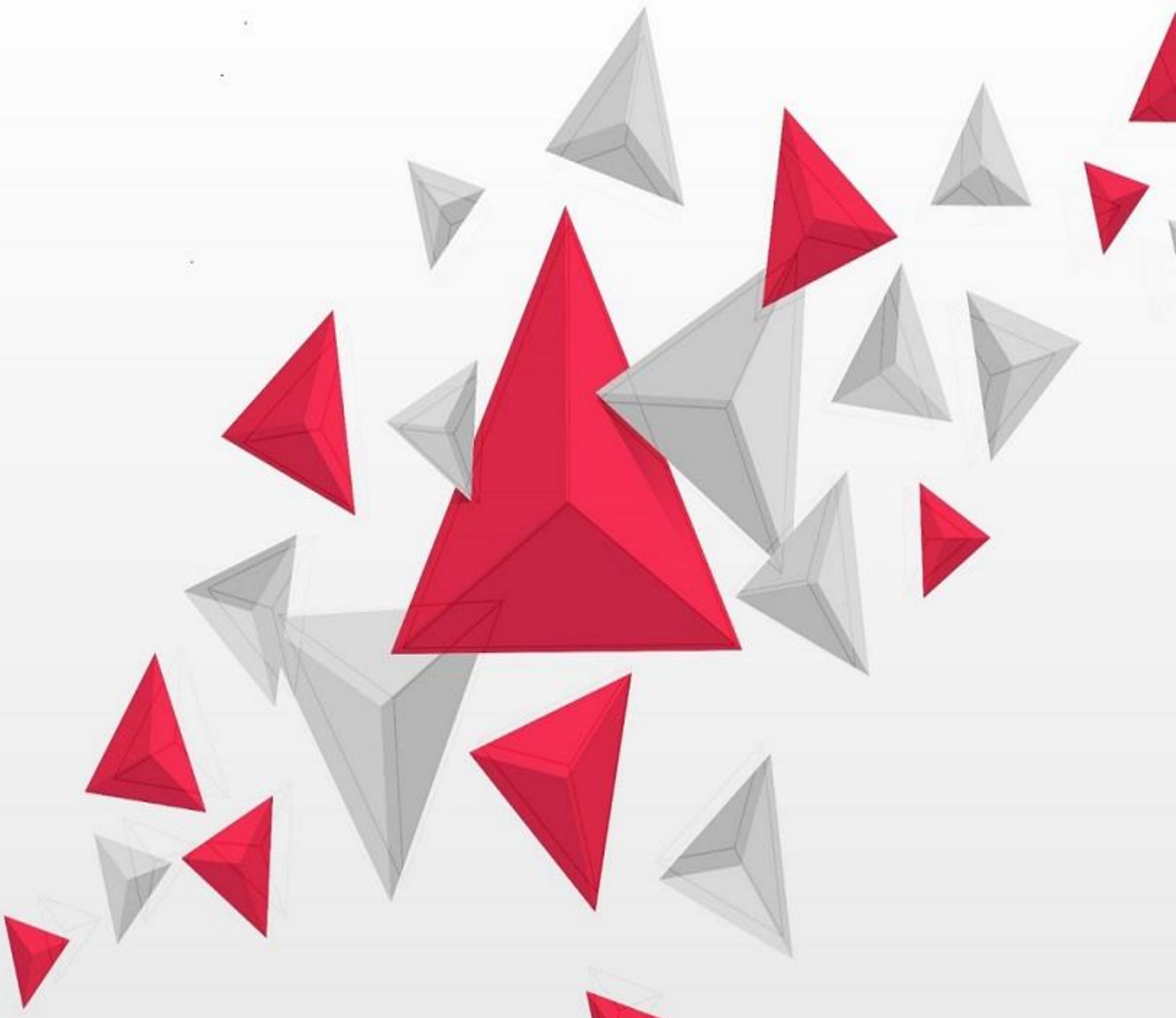


# Business Requirement Document (BRD)

## Guide Book





## BUSINESS REQUIREMENT DOCUMENT (BRD) – INTRODUCTION

If you have any experience (in whatsoever capacity) working in the field of Information technology and software development, the chances are that you have heard of the term ‘BRD’.

And, if you are aware of BRD, then you must have always wondered what’s the difference between a BRD, i.e., Business Requirement Document, an SRS, i.e., System Requirement Specification, and an FRS, i.e., Functional Requirement Specification is!!

Although the scope of this very lesson is talking and understanding all about a BRD, let me first clarify a lot of notions (or shall I say baggage) that are prevalent within the industry about the terminology and usage of requirements and specifications documents.

If you have noticed, I have segregated the documents under two broad headings, i.e., *requirements & specifications*, and going by their very definitions:

- *Requirements* outline ‘what’ the software or the system is expected to do
- *Specifications* contain a *detailed description* of ‘what’ the system will do it

Just remember this clear and straightforward distinction, and you will never get confused between the plethora of business analysis documents and their usage.

Requirements documents are always created first and contain the need, the users, and a ‘very high level’ description of the software or the product being produced. They are made by the business analysts after detailed discussions with the client stakeholders and are considered as formal documents, reviewed by all the critical project stakeholders, i.e., the project manager, program manager, client SMEs, project investors, etc....

Specifications documents are created after the 'basic requirements' are defined and contain the 'high-level drill-down' of each requirement, elaborating the details, components, and other aspects. These documents may be created solely by the Business Analyst or in conjunction with other project members like the Technical Architect or the Project Lead. The audience of specifications documents ranges from business users, professional team members, and testing (quality) team members.

**Requirements documents:**

- Business Requirements documents or BRD
- System Requirements documents or SRD

**Specifications documents**

- Functional Requirement Specifications or FRS
- System Requirement Specifications or SRS
- Software Requirement Specifications or SRS
- User Requirement Specifications or URS
- Technical Requirement Specifications or TRS

Let's strengthen our understanding with a quick example:

**BRD** may have a statement like, "The system should produce pre-defined reports about the performance of the candidates."

**FRS** will include the details like "The user, post login, will be redirected to the dashboard, containing the following reports:

- Assessment shared: Assessment-wise numbers of the candidates with whom the assessment is shared.
- Assessment attempting: Assessment-wise numbers of the candidates who are attempting the assessment
- Assessment completed: Assessment-wise numbers of the candidates who have completed the assessment
- Assessment cleared: Assessment-wise numbers of the candidates who have cleared the assessment.
- Assessment Failed: Assessment-wise numbers of the candidates who have failed the assessment
- Yet to take the assessment: Assessment-wise names of the candidates who are yet to take the assessment and their due date has not elapsed."

In many (or shall I say most) organizations and institutions, the above requirements documents are not created separately but are used interchangeably with sections so intermingled that one cannot distinguish whether it's a BRD or an SRS - that's the primary reason why you will not be able to get an industry-wide standard definition of the BRD or SRS.

For the sake of simplicity and understating, we will discuss all the significant sections of a requirement document under the nomenclature BRD, starting with its aspects.



## ASPECTS OF A BUSINESS REQUIREMENT DOCUMENT

1. A project BRD defines the very need of initiating the project. It contains the primary business objectives, describes what business problems will be solved by having such a product or software, and lastly, what benefit the client (customer) gains from successfully completing the project – often called the ‘*business case*’.

Defining this *project rationale* is very important as it becomes a yardstick to validate that all the subsequent requirements/features/modules added to the project should fulfill the original project rationale.

2. The formal BRD is agreed upon and signed as the part of the ‘contact\*’ between the client (customer) and the contractor, and it’s a *legally binding document* (meaning that both the parties have to acknowledge the requirements and terms written in the document, failing which a party can take legal action upon the other party).

\* In some organizations, the business requirements are detailed out in the SOW document itself, whereas, in others, the BRD is added as an addendum to the SOW.

3. The BRD contains purely business-oriented information (‘what’ and ‘why’ of the solution), and no technical or infrastructure-related details are ever included in a BRD.
4. It contains the ultimate feasible solution proposed to solve the business problems the client faces, its features, and different aspects. Alternatively, it

defines the very boundaries of the work (i.e., project scope) that will be executed as a part of the project.

Having a clearly defined project scope is of prime importance as it prevents scope creep and decreases the chances of project failure.

5. The requirements defined in the BRD should have the complete consensus of all the relevant stakeholders of the project (see the audience section below). This agreement is imperative as it establishes that all the parties have been consulted, their requirements and priorities are acknowledged, and they agree to the solution being developed.

Additionally, this agreement reduces the chances of having any future disputes or differences in opinions.

6. The BRD defines the project success factors or the completion criteria and sets expectations with all the stakeholders as to when the project will be called 'successfully delivered'.

Not only that, it contains a host of other sections that define the aspects around the business requirements, like assumptions and constraints, dependencies, risks, limitations, and stakeholder lists.

7. The BRD also does an excellent work of serving as a reference document for all communications between the business and development teams. Not only this, but it also serves as an input to some of the future documents (like the functional specifications and the technical architecture documents) of the project.



## AUDIENCE OF THE BUSINESS REQUIREMENT DOCUMENT

A BRD is a collaborative document that involves the inputs and reviews of a lot of stakeholders. Let's see who all is included:

1. The *management* (VPs, business partner heads, and account heads) is responsible for explaining the project's business case, gives periodic suggestions & comments on various features and aspects of the BRD, and reviews/sign-offs the completed BRD
2. The *core project team* (project managers, business analysts) is involved in the creation of the BRD, with the BA taking the lead, talking to all the relevant stakeholders, drawing out information, and validating them
3. *Subject matter experts and process owners*
4. *Quality department (PMO)*
5. *Change management team*
6. *Development/Technical team*



## HOW TO CREATE BUSINESS REQUIREMENT DOCUMENT

Okay, so get ready for some documentation heavy-lifting!

The BRD is created in the initial/planning stages of the project, and it's usually one of the first documents created by the business analyst in a project's lifecycle.

To create the BRD for a completely new product, a BA has to take the project Vision document as a base and identify the project's key stakeholders (they are usually listed in either the project charter or the vision documents). Then, the BA should start to get more details from them about the product's features, one requirement at a time. Using the project charter/vision documents as a base enables us to review the accuracy of the information listed in them while serving as a reference document for everybody.

Irrespective of the reference/base document used, the BAs should conduct workshops and meetings with the customer and concentrate on answering the likes of the following questions within the BRD:

- What is the problem that the organization is trying to solve?
- What is the business objective of the project?
- Are there any restrictions that need to be considered?
- What are the underlined business rules?
- What are the qualities (non-functional requirements) expected from the product?

While writing a BRD, a BA might get tempted to include the solutions or 'How' the application/product will solve the business problems, however, remember a BRD is solution independent document, and such details should go into one of the 'specifications' document.

Let's see what the different sections within a BRD are and what each one of them contains:



## 1. Executive Overview

This section is an introductory paragraph to the BRD and should include the following items:

- An introduction of the client (customer) and the vendor
- The purpose of creating the document
- A brief overview of what all this document contains
- The details of the intended readership of this document
- List of activities/documents this BRD will serve as an input to

## 2. Client Details

Knowing about the client, their domain, and business operations helps develop a better overall understanding and supports the project core team (especially BA) in recommending a solution that aligns better with their existing business framework.

Some of the information that should be a part of this section:

- Client name
- Client's domain/nature of business
- Number of years of operation
- Their existing products/services
- Their core beliefs/vision
- Their strengths/Unique selling proposition (USP)

## 3. Stakeholders Categories

This section contains the different verticals or categories of clients that have a vested interest in the project's success and could be affected by the project's outcome. The following subsections might be used to depict the details:

- Stakeholder Type: The types are broad areas to which the stakeholders could belong. The permitted types are
  - o Project Management,
  - o Business Analysis,
  - o Testing,
  - o Application development,
  - o Subject Matter Expert (SME),
  - o Client,
  - o Management,
  - o Vendors,
  - o Suppliers,

- Stakeholder Category: There are only two categories here:
  - o Internal (the core project development team, i.e., the PM, BA, testers, and development team)
  - o External (Vendors, suppliers, and outside organizations)
- Key Stakeholder Names: Names of the stakeholders for each of the stakeholder type
- Responsibilities: Key responsibilities of the stakeholders for every stakeholder type

#### 4. Background

This information could also be taken from the project vision document and contains the following details:

- Details of the existing business environment
- Current situation/process flow/flow of events
- Issues and problems faced today
- How will the product/project solve those problems?

#### 5. Business Objectives and Vision

This is the section where the BA should define what the project intends to achieve and what needs of the users will be fulfilled after the successful completion of the product.

It should contain:

- The goal of the project (should be a logical solution to the problem explained in the above section)
- Relevant benefits that the users will achieve after using the product
- Name or unique identifier of the product/application being developed (if decided by then)
- Brief description of what components/modules the system is expected to have
- Details of actions/activities that could be performed within each of the components

#### 6. Scope

This section documents a list of functionalities and features that should be a part of the project development efforts. Also, the requirements under this section should be specific, clear, and crisp while being realistic (i.e., achievable).

- a. **In-Scope:** Provide a *high-level summary* of what is to be included (in scope) for project completion and should be considered part of project efforts – the overall project boundaries.  
Care should be taken with this section to include only the ‘What’ of the solution here (and not ‘How’). Also, items under the scope and functional requirements sections should be subjected to a thorough review (from both internal and external stakeholders) as they will be referenced in case of change requests discussions and disputes.
- b. **Out of Scope:** Provide a high-level summary of what should not be considered (out of scope) for project completion and shouldn’t be a part of the current project scope (these requirements can always be a part of future phases of the project).

## 7. Functional Requirements

This section contains a more granular level of requirements and includes the specific behaviors, characteristics, and attributes that are expected from the application or the software being developed. Additionally, a listing of use cases, high-level workflow diagrams, wireframes, or other similar types of information should be supplemented here.

- a. **Business Requirements:** Following information that gives details about the solution’s business requirements should come here:
  - Information about different kinds of product users
  - Roles and responsibilities of those users
  - Expected system behavior
  - Concepts and scenarios
  - Use cases listing (if available by then)
  - Wireframes
  - Any other type of requirement related information
- b. **Business Process Flows:** Any details and diagrams related to the solution process flow, data flow, and information flow should come here. Also, if there are multiple processes within the system, the details of where those processes fit in, how are they interconnected (if they are), and to what effect they are used.

## 8. Non-functional Requirements

The system's operational characteristics or non-functional requirements, like system response time, performance, scalability, and usability, are included here. While adding requirements over here, try to see them as qualities that the system should have.

- a. Performance Requirements:** List all the performance-related attributes and characteristics here. All such details will help the technical team carry out the capacity planning for the servers and other hardware as well as software components of the system being developed.

It should have details for:

- Average load on the system (number of users)
- Response time (minimum acceptable)
- Turnaround time
- Throughput for optimal performance
- Maximum (peak) workload the system should handle
- Scalability (ability to manage the additional workload if additional hardware/computation processors are added)
- Any assumptions regarding the performance
- Specify limits for how long it's tolerable for different types of fault to remain undetected (fault detection and prevention)
- How long will the system be available (all day or at specific times)
- How will the users learn of the unavailability
- Any fallback activities needed in case of non-availability

- b. Usability Requirements:** Applications and products should be easy to learn and efficient to use. Any requirements regarding the usage of the system being developed should come here; examples include:

- Ease of use
- Effectiveness of use
- Speed of performance
- Intuitiveness/understandability
- Satisfaction

- c. Security Requirements:** These requirements define how secure (from unethical and non-permitted users) the application and its network, servers, operating systems, and infrastructure are and depend on the kind of software or application being built and the data it contains. The analyst

should spend a fair deal of time with the technical and IT department of the customer and propose the security requirements applicable in the industry, like:

- User authentication
- User authorization
- Data access
- Access control
- Data integrity
- Vulnerability (to hacking) testing

**d. Training Requirements:** Any requirement where the users have to be trained regarding the usage of the system should come here. It includes

- End-user training
- Training of support personnel
- Preparation of training manuals/guides/videos

**e. Recovery Requirements:** These are the requirements that ensure that the business and services are up and running even in the event of any natural disasters and unforeseen circumstances (also called Business Continuity). It should contain:

- What will qualify as a disaster - its definition?
- Creation of DR plan: A plan which contains how the services will be restored in the event of a disaster
- Recovery Point Objective (RPO): Acceptable amount of data that can be lost in the event of a disaster
- Recovery Time Objective (RTO): Acceptable amount of time the application and services will be down in the event of a disaster
- What are the performance degradation tolerance limits in the event of a failure?
- How often a DR mock drill needs to run?

## 9. Quality Control and testing

This section contains all the requirements and activities that need to be exercised to ensure that the quality of the application is of high standards and stays within the defined limits through the project development life cycle. These activities include, but are not limited to:

- Define the extent of testing required (testing scope and out of scope)
- Identify the nature of the testing (manual or automated)

- Define the type of testing to be performed (functional, performance, security, etc.)
- Testing techniques, methodologies, and strategies to be used
- Frequency of the testing activities
- What all types of documentation need to be prepared against the testing
- Metrics that needs to be captured to know the progress of testing on the project
- Any other testing related requirements

## 10. Other Requirements

Requirements that don't fit in any sections defined above should come in here. Some of the sub-sections that could be included here are:

- **User Interface requirements:** Requirements related to the user interface design of the whole application or individual modules
- **Project management requirements:** Any requirements related to how the project should be managed, monitored, or controlled should come in here.
- **Reporting requirements:** Reporting the project progress could be done in many ways, ranging from sending a weekly or monthly status report or having a daily status update call with the client – all such reporting requests should be here
- **Roll-out and milestones:** Milestones are special events (like delivery of functionality or completion of a module) in the lifecycle of a project, and all such details should come in here
- **System maintenance:** Any requirements related to how the system needs to be maintained post its go-live
- **Storage requirements:** Any requirements as to how the data should be stored, segregated, or clustered (particularly in data or process heavy applications)
- **Resource/workforce requirements:** Special notes regarding the type, experience levels, competency of resources that needs to work on developing the application

## 11. Completion Criteria/Exit Criteria

It's essential to have the definition of 'Complete' defined from the project's perspective, and the criteria that need to be fulfilled to mark the completion of project tasks are detailed in this section.

It should include:

- Compliance with requirements/project business case
- Unit testing of functionality
- Code review
- System testing
- Integration testing
- No/low defects

## 12. Constraints

Constraints are any factors that determine the boundaries around the project functions or dependencies that explain how the project activities are carried out. There could be several factors that could be a project constraint:

- User Interface Design: A UI design that is difficult to achieve technically
- Code Review/Due diligence: Code reviews couldn't be avoided, but they slow down the application development pace
- Human Resources: Getting skilled/specialized resources to work on the project or a fixed size of the development team
- Legal: Certain laws or regulations that restrict project activities
- Organizations Process: Processes that are defined at the organizational level and needs to be adhered to
- Quality: Certain quality or conformance related requirements that should be fulfilled
- Technical: Any condition that is not possible to achieve/has a technical limitation
- Duration: Completion of a specific activity within a stipulated time duration
- Budget: The project budget controls the amount of money that could be spent and thus places a boundary on many project functions.

## 13. Assumptions

Any belief/events/premise you consider to be true in the near future is an assumption.

The reason why assumptions are made is that since you don't know everything about the project aspects right from the very start, certain things are assumed to be real and future plans are made based on this.

In the absence of assumptions, a lot of decisions and plans couldn't be made at all. However, since all the assumptions don't tend to be accurate, it's imperative to document them and communicate with all the stakeholders about the same.

There are some common areas where assumptions are taken:

- **Resource assumptions:** Assumptions that the skilled resources will be available as per the schedule and will work with their full capacity
- **Technology assumptions:** Assumptions that there will be no technical limitations or hindrances
- **Availability assumptions:** Assumptions that all the schedule functionalities, resources, and vendor-related deliverables will be available
- **Requirements assumptions:** Assumptions that all the requirements are thoroughly thought about, reviewed, and approved
- **Accuracy of schedule:** Assumptions that all the in the project tasks and functionalities will be available/delivered as per the schedule

#### 14. Dependencies

Typically, many project activities are dependent on each other, and the important ones should be listed here. Some examples include:

- Resource-based dependencies
- Task-based dependencies

#### 15. Risks

Any uncertain events or factors that threaten the successful implementation of the project activities are classified as a risk. This section should contain any of the initial risks that are foreseen based on what is known about the project to date.

Some of the areas from where risk could emanate are:

- Unavailability of resources/skilled resources
- Incorrect/overly optimistic estimations
- Assumptions tend to be false
- Unclear/poorly defined requirements
- Scope creep
- Learning curve
- Lack of change management
- Requirements are not reviewed/signed off
- Vague or inadequate project communications
- Technical/implementation risks



## 16. Glossary of terms

Alphabetical listing of all the business language used in the Business Requirement Document and their associated explanation is provided under this section



## BUSINESS REQUIREMENT DOCUMENT - BEST PRACTICES

### 1. Always create the BRD with the business need in mind

It's quite common for the customers to get carried away by the bells and whistles features - requirements that are nice to have but do little in fulfilling the actual business objectives. Also, in the absence of facilitated workshops and brainstorming sessions, the business stakeholders will not be able to notice any conflicting requirements.

The business analyst should act as the rudder who sets the project requirements on the right course and validates each one of them against the project's business case (okay, bad analogy, but you get the drift?).

### 2. Elicit and validate requirements, not collect and gather

When you are asked to collect something, it's already lying out there. You have to just go and pick it up. However, *elicitation* is not that's simple!

You have to get your hands dirty, draw it out, extract it, and evoke it.

So, if your stakeholders are dictating to you the requirements with you just rephrasing, it won't be called business analysis.

Before it becomes a part of the BRD, each of the requirements should be talked through, properly validated for its necessity, and the analyst should look at all the not-so-obvious details that could be derived from the high-level requirements. Missing such attributes in the BRD could lead to scope creep in the future, resulting in deviations from the schedule or, in worst cases, project failures!

A significant takeaway here - all the project requirements should be tracked back to the BRD.

### 3. Ensure visual depiction of your requirements

Human mind processes and retains visual information twenty times faster than textual information. Thus to aid better comprehension of high-level requirements and allow their logical dissimulation into precise lower-level details, the analyst should ensure that the application processes, complex information, and data flows are supplemented with adequate visuals.

### 4. There is no room for ambiguity in the BRD

It's always a given that the requirements, before being a part of the BRD, are properly thought about, and thus there shouldn't be any statements or phrases that are open for interpretation.

So, look out for the following words in your BRD:

Maybe	If necessary	Around
Approximately	Imaginably	Something like
Could be	Roughly	Give or take (a few)

### 5. Use a well-structured and standardized template

Using an adequately structured template ensures that the BRD contains all the required sections, and the flow of information and requirements in the BRD are logical, clear, and intuitive.

If no requirements are identified for a particular section for the time being, then the analyst should update 'None Identified' rather than just simple 'None' or N/A. This way, one can always come back to that section when more details are available around the project rather than leaving that section altogether.