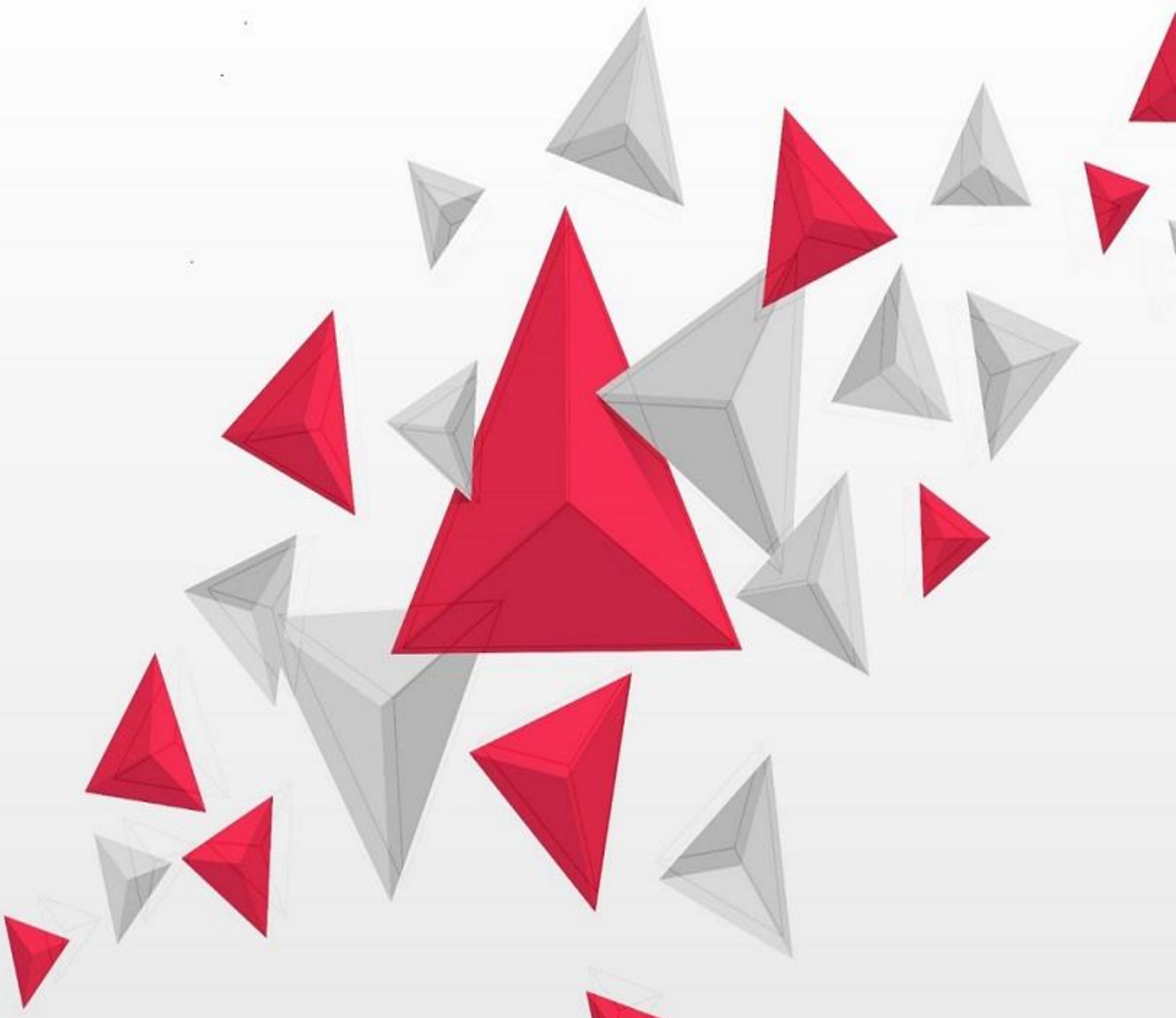


Functional Requirements Specification (FRS) Document

Guide Book





FRS DOCUMENT – INTRODUCTION

The Functional Requirement Specifications (FRS) document is the last of the specifications documents in a project and the most detailed and granular document of the three (BRD, SRS, and FRS).

FRS describes precisely ‘how’ the system is expected to function (its behavior) in order to satisfy all the requirements listed in the BRD and SRS

A Functional Requirement Specifications (FRS) document is a granular and low-level document that elaborates all the details around the functional requirements on a project.

The FRS document is prepared either by a systems analyst or a business analyst. However, in organizations where there is no dedicated role of a “Business Analyst”, the Implementation Leads assume the responsibility of authoring the FRS document.

The FRS document is very descriptive and elaborates on all the functional requirement (including business logic, compliance, security requirements, etc..) unambiguously by specifying all the fields and user interactions within every section/module of the software being built. Also, to aid better comprehension, process flow diagrams, UI elements, and wireframes are added to the FRS.

It should be noted that the FRS document is created from the perspective of a user and describes how the software will behave when an external user interacts with it.



ASPECTS OF AN FRS DOCUMENT

- *Not all organization create the FRS document*

Don't be surprised if you don't find the FRS document within some organizations. Some organizations or small projects do not create an FRS and instead elaborate their BRD or their SRS documents to accommodate the functional, non-functional requirements and use case details.

- *Not in the FRS, not in the project scope*

FRS contains the most comprehensive listing of all the project's requirements – all within a single document. If a requirement is not listed in the FRS document, then it can be safely assumed that the requirement is not under the project's immediate scope.

- *FRS document is sometimes referred with some other names*

FRS may sometimes be called Functional Specifications Document (FSD), Functional Specification (FS), Product Specification, and Functional Specs., however, the essence and the contents of the documents remains the same.

- *FRS document is prepared after the BRD and SRS documents*

Since the Functional Requirement Specification (FRS) document is a logical extension and elaboration of the SRS document, it is prepared after that.

- *The FRS document is prepared in the planning phase of the project*

Like the SRS document, the FRS document is also prepared during the 'planning' stage of the project and is entirely a functional document that doesn't contain any technical details



AUDIENCE OF AN FRS DOCUMENT

Let's take a quick look at the primary audience of the FRS document

- Project managers oversee the preparation of the FRS document and ensure the alignment of its content with the BRD and SRS
- SMEs (subject matter experts) evaluates the correctness of the information presented in the FRS
- The project's technical architects and Implementation Lead responsible for the design and development of the listed functionalities
- Business stakeholders carefully review the FRS in order to ensure it lists all the system features and functionalities
- Development/Technical team to understand what exactly they have to build
- Testing Team to know what are the different scenarios & test cases on which the software should be tested



HOW TO CREATE AN FRS DOCUMENT

To create a thorough, comprehensive, and all-inclusive FRS document, a business analyst has to elicit requirements through multiple ways, including – workshops, interviews, casual communications, documentation review, and research.

Described below are the sections that one has to elaborate exclusively for the FRS document.

System Features (Module-wise)

A Functionality is defined as a desired output or result expected from the system after a (series of) input. For any system with a moderate level of complexity, such functionalities are grouped to form a ‘feature’, and similar features constitute a ‘module’.

- **Business Requirement ID:** Each requirement should be backward traceable by explicitly reference its source (business requirement ID) in earlier documents and should be organized in the below format
- **Functional Requirement ID:** All the functional requirements in the project should have a unique reference number (functional requirement ID) to allow forward traceability
- **Requirement definition:** The format of the functional requirement definition should be: The system shall <requirement of what the system should accomplish>

Business Req. ID	Functional Req. ID	Requirement definition
BR 1.0	FR1.0	The system shall [parent requirement - module 1]
BR 1.0	FR1.1	The system shall [parent requirement - feature 1]
BR 1.0	FR1.1.1	The system shall [child requirement – functionality 1]

BR 1.0	FR1.1.2	The system shall [child requirement – functionality 2]
BR 1.0	FR1.2	The system shall [parent requirement - feature 2]
BR 1.0	FR1.2.1	The system shall [child requirement – functionality 1]
BR 1.0	FR1.2.2	The system shall [child requirement – functionality 2]
BR 2.0	FR2.0	The system shall [parent requirement - module 2]
BR 2.0	FR2.1	The system shall [parent requirement - feature 1]

Business Process Flow

Most of the time, software applications follow a hierarchical/sequential flow of events that are initiated after a specific sequence of inputs is carried out.

Any such details and/or diagrams related to the solution process flow, data flow, site flow, and information flow should come here.

Wireframes/Prototype

Prototype or mockup is an initial version of a product and gives a visual depiction of the end product. A prototype or wireframe of the software being created should be included under this section to depict any data or process-based navigation, to represent different scenarios and to project the general look and feel of the application being created.

The aim of the prototypes should be to validate the understanding of the requirements and get initial feedback against the user interface and design elements.

User Interface Requirements

This section contains the requirements about the look and feel of the interface using which the user will interact with the software. Following details shall be included under this section:

- The general layout of the application
- The way content, as well as data, is presented to a user
- Different kinds of navigation available in the software
- Representation of dynamic design elements (widgets & menus)
- Does the user interface differs based on the role/user group of the user?

Use Cases

Use cases accurately describe how a user (actor) will interact with the system being developed, through a flow of events.

This section should contain comprehensive details around all the use cases that are created as a part of the application requirement documentation, in the below format:

Use Case ID	Use Case Name
Pre-condition(s)	Describe the conditions assumed to be true before the use case can be started
Trigger	Describe the condition/action that initiates/starts the use-case
Post-condition(s)	Describe the state of the system at the conclusion of the use case execution
Actor(s)	An actor is a person or other entity external to the system who interacts with the system. Define all the main actors of the use case here.
Main flow of events	Describe system responses that will take place during the execution of the use case under normal, expected conditions
Alternate/extension flow	Describe legitimate branches from the main flow to handle special conditions
Exception flow	Describe any anticipated error conditions that could occur during the execution of the use case
Related use cases	List any other use cases that related to this use case
Additional Information	Any additional notes and details against this use case will come in here

Field level specifications

This section contains the requirements, properties, and validations against each of the field data elements in the format below:

Field Label	UI Control	Mand?	Editable	Data Type	Value Set	Default Value
User name	Textbox	Yes	Yes	Text	None	NA

- Field Label: Specify the name of the label for this field
- UI Control: Define what kind of UI control this field element will have
- Mandatory: State whether this field element is compulsory or not

- Editable: State if this field element can be edited by the user or is read-only
- Data Type: If editable, specify the type of data that can be entered by the user in this field
- Value Set: If a set of values is populated for this field element (like a dropdown), specify the complete value set
- Default Value: Specify if this field element should default to any value]

Business Rules and Dependencies

This section should contain the business rules and dependencies against the field data elements in the format below:

Field Label	Business Rules	Error Messages	Data Dependencies
< Field label for which the business rules are to be defined >	< Specific rules based on which the data should be populated or validated >	< In case the business rules are not met, state what error message should be displayed and under what conditions >	< Specify if the data being entered is dependent on any other field/value >

User Interface (UI) Elements

This section contains the requirements against specific user interface elements like button, links, and icons in the format below:

Type	On-click Event	Other Events	Enabled / Disabled	Navigate To	Validation
'Create User' button	Display a toaster message 'User created successfully'	Create a user at the backend acc. to the data entered	Enabled, by default.	User list page	Verify if the mandatory fields are filled

- Type: Define the type of user interaction element i.e., button, icon or link
- Label: Specify the label of the UI element

- On Click Event: Define the action performed by the system when this UI element is clicked
- Other events: Define action performed by the system on other interaction events for this UI element
- Enabled Vs Disabled: Define the conditions based on which when UI element will be enabled/disabled
- Navigate to: Specify the page to which the user shall be navigated when this UI element is clicked
- Validation: Specify if the system has to perform any validation when this UI element is clicked



FRS DOCUMENT - BEST PRACTICES

Since the FRS documents closely resemble the way information is structured and presented in an SRS document, let's look at the key best practices for the specifications documents, one more time.

1. **Don't Ramble! Make your sentences short and crisp**

Traditional Research says that a human being can hold 7 objects in his working memory at a time, and the latest studies point that this number has come down to 4. So, if you are using excessively long sentences, you are increasing the chances of your stakeholders missing or misunderstanding your FRS document. So, make your sentences short and chunk the information into multiple sentences to allow better understanding for the business users. Also, developers take one requirement at a time, and they will be more focused and clear with crisp and straightforward sentences.

2. **Avoid ambiguity around your FRS document**

You should strive to be as specific as possible in your requirements. Avoid using words like approx., etc., some, sometimes, ordinarily, most, mostly, usually and might.

Also, make sure you avoid duplications wherever possible and should also steer away from contradictory and debatable statements.

Crisp documents portray clear thinking and methodical outlook, and both these skills are so very integral to the role of a BA.

Let's see a small example around **Specific and Precise** requirements:

The user can change their password by clicking on the change password link on the login page, followed by writing their registered email ID on the following

screen. Then, the system will send an email on their registered email id, which, when clicked, will bring the user to a new screen. On this new screen, the user can now change his password.

The steps to be followed while changing the password are:

1. Click on the 'Change Password' link on the 'Login' page
2. The flow goes to a 'Confirmation' screen where the user's registered email ID should be entered.
3. The system sends a 'Reset Password' link on the registered email ID (entered in the previous step)
4. Clicking on this link will bring the user on the 'Change Password' screen.
5. The new password could be set on this screen.

Now, let's see an example of a consistent requirement:

Inconsistent

Consistent

The application shall	The User management tool shall
The User management tool shall	The User management tool shall
The system shall	The User management tool shall
The user management system shall	The User management tool shall

Now, let's see how the requirements are complete:

Incomplete	Complete
While resetting a password, it should not be the same as the previous passwords. Also, the password guidelines should be followed	While resetting a password, it should not be the same as the last 5 passwords. Also, the password should be a minimum of 8 characters long with at least one numeric and one special character

Let's see an **unattainable** requirement:

With one application server, the User management tool shall serve 2 million concurrent users, and the response time should not be more than 1 second.

An example of how the requirements should be **testable**:

The web application should have a multilingual support	The web application should support English, French and German languages
The user interface should work in all the major browsers	The user interface should work in IE, chrome and safari browser
In case of wrong credentials system should give a 'proper' alert message	In case of incorrect credentials, the system should show an alert message reading 'Invalid Credentials. Please check the email ID/password entered.

3. Know how to use: Shall, Will, and Should

Many times, due to our lack of knowledge of words, we use them interchangeably, confusing for the end-user. The most common is the use of Shall, Will, and Should.

- "Shall" should be used where requirements are being stated, - The system shall have a response time of < 200ms
- "Will" should be used to represent statements of facts – The system will have a separate login for administrative users
- "Should" represents a goal that needs to be achieved. – The system should concurrently cater to 10,000 users at any given point of time.

4. WHAT and not HOW

First thing first – You, as a business analyst, have to describe 'What' the system will do. *That's it.* Now, although you might think that this tip is so very obvious, you forget the fact that many times we inadvertently document 'How' the system does it.

E.g., You, in your use case, might write something like, When the user clicks on the 'save' button on the 'Create User' page, the system will create the user in the application portal by saving all the attributes defined in the page in the MySQL database.

Do you think it's correct? No, it isn't.

In the first half, you describe what the system will do, and in the next half, you described how. So, catch yourself when you're mentioning programming language and software objects in your 'requirement' documents.

5. Define one and only one requirement at a time

Your requirements should be atomic, meaning it should comprise of only one component. Don't be tempted to combine multiple requirements by using conjunctions like *and*, *or*, *also*, *with* and the like. This is particularly important because words like these can lead to developers and testers missing out on some of the requirements. One way to achieve this is by breaking the requirement down until it can be considered a discrete test case.

One other great piece of advice is to express any equation or formula being used in the requirements in words - This will be a massive aid to someone reading your requirements. Go ahead at giving at least 2 examples to corroborate your equation or formula.

6. Document reviews are as important as the document itself

There is a lot of difference between a document and an approved document. An approved requirement document is a baseline document and an artifact that is verified and validated. But an unverified use case is just a piece of client's wish-list. So, always have a dual review of your documents, first by an internal peer or superior and then by the client. The review date, reviewer name, and reviewer's comments should also be mentioned in the document.

7. Check before you channel

There are few things which you should check before sending your FRS document for a review. Let's quickly see what they are:

- Formatting of your FRS document – Never send an ill-formatted document as they reflect your work ethics and professionalism
- Spelling and grammar check – Spelling and grammatical mistakes reflect a careless attitude. Something that nobody wants to be associated to
- Accuracy of the attached diagrams – attachments should be adequately defined. Also, make sure you are attaching the right document as I see people attaching the wrong documents in a hurry
- Names and terminology used – Be careful about the names of persons and vocabulary of terms in your documents because if such a mistake is made, it causes a lot of confusion
- The authenticity of Links and references to other documents – if you are referring to any external source or link in the FRS document, make sure it's from a credible source or else you will be considered as a gullible analyst

8. 'Out' of scope but 'In' the document

In the initial stages of my career, I was working with an aviation client on a 'Workflow Management System'.

At the time of our UAT, the client asked for a feature that was discussed briefly and was thought by the team as an item for the next phase of the project.

However, the client was adamant and demanded us to show where it is documented that the feature is out of scope for that phase!

We never had that documented, and that caused us 10 days of constant effort to have that feature of ready for them. That day I learned, 'If a feature was discussed even in passing, your client might assume it is "in-scope" until and unless you clearly document it as "out of scope"'.

9. A final tip - Be flexible in developing the FRS document iteratively

Many times, during the planning stages of a project, a light-weight FRS document is developed, which is then extended during the elaboration or clarifications sessions. You should understand that your FRS document will work as a central mechanism for reducing risks and requirement ambiguity. To achieve so, you will have to create different versions of the FRS document reflecting the change or updating in the requirements. I am speaking from experience that not many BAs maintain versions of their FRS documents, and if you are doing so on your project, then you are sure to gain attention and recognition for that.

And when you are updating versions, make sure to update the associated diagrams and matrices as well.