**WHAT IS A PRODUCT REQUIREMENTS DOCUMENT**?

A Requirements Document should act as the starting point for your product. Outlining its purpose, who will use it, and how to use it. It is an essential precursor to product and development.

**THE FRAMEWORK**

Below are the sections suggest for a simple product requirements document:

Goals

User Personas

User Stories

Sitemap

Page Descriptions

Wireframes (optional)

Non-Functional Requirements

Risks

Future Iterations

1. **GOALS**

The business goals and objectives serve as the context, showing the dev team why they are building what they are building.

Here are some common questions to ask when fleshing out this section:

What is the purpose of this project?

What are the problems it will solve?

How will it streamline or improve the current process or facilitate a new process?

What is the product vision?

Check out our Free Product Requirements Document Template for more examples.

1. **USER PERSONAS**

User Personas are hypothetical individuals who match your desired, or actual, audience. Thinking about the background of these users will improve your ability to create a product that meets their needs.

User personas are one of the areas that people tend to drown in. So let’s establish some rules to keep you afloat.

* Cover the primary types of users. You don’t need to create a persona for every kind of user, just enough to illustrate the main groups who will use your product. For simple applications, 3 good profiles will cover more than 80% of your user base, but if your application is complex you may need more.
* Focus on what you know. Making up details to simply fill an end user profile is counterproductive.
* Start with these 5 metrics. Occupation, age, gender, location, education. They will be the framework for your profile.
* Name your End Users. It’s easier to relate to “David the Executive” than to “End User D”.
* Give each profile an objective. Each user needs a goal that fits their profile. This will be their raison d’etre.

Essentially it boils down to this: your personas should be detailed enough to allow you to see the product through their eyes. Each profile should function like another person in the room when making a decision. “Would David login to read his message or should we email it to him?”

If you have an existing website, use Google Analytics to help you create these profiles. It gives you invaluable information you can see where your visitors came from, what keywords they used to find you, and what content and behaviors they exhibit online. If this is a new product, Alexa (and other similar sites like Similar Web) will give you similar data for your competition.

1. **USER STORIES**

User stories are short descriptions of a feature, told from the perspective of one of your newly created end user profiles. They are typically structured in the following fashion:

As a [type of user], I want [some goal] so that [some reason].

User stories are a starting point, not a destination. It’s often best to think of them as pointers to your eventual requirements. These stories are vital because the discussions they start will help shape your content architecture and design.

So how can you use this concept to move your product forward without getting bogged down in hundreds of stories and feature ideas?

* Limit yourself to high level (and must have) stories. These high level stories are known as Epics and can be broken down later into smaller more manageable stories.
* Your stories should be short and specific.
* They should describe who needs what and why. They must be user-centric. Remember you can use your personas.

Here is an example focussed on a user role:

* As an Admin, I want to be able to see new products and products categories as they are ordered so I can write content for the website.

And here is one more focused on the user persona:

* As a Website User, making the correct product choice is vital. I want to see my product options side by side so I can make an educated choice quickly.

You don’t have to write a user story for every little bit of functionality in your application up front. You should focus on the Epics because details will be fleshed out when wireframes and designs are created. Epics are broken down into smaller stories in backlog grooming sessions. Remember, at this point, you are writing to give the designers and developers the minimum information needed to start a productive conversation about how to create your application.

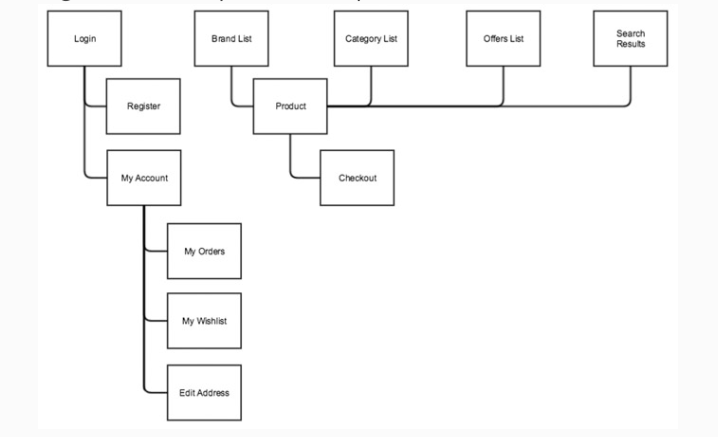
1. **SITEMAP**

Once your main user stories are defined, you should have a solid idea of what pages or screens your application will page. The sitemap is the first step to documenting this.

A good sitemap includes the following:

* A comprehensive list of all pages or screens. This includes everything from pages detailing information about products and services to an “About Us” page and a privacy policy. Spend some time brainstorming to ensure everything you want in your application is listed.
* The hierarchy between these pages (optional). If you have an idea of how your pages relate to each other hierarchically, then communicating this will be helpful for designing the navigation. However, this step is not a must, and a comprehensive list of all pages is enough for the bare minimum.

Figure 1: Example Sitemap



A good solution to quickly create a sitemap that you can easily share with your stakeholders is Gloo Maps. Or just use one of the free add-ons for Google Docs.

1. **PAGE DESCRIPTIONS**

Once you have a list of all of the pages or screens in your application, you can take the next step and define what will be on each page. Make a simple numbered list of the major items that will be included on each page and put the most important items at the top of the list (this will convey the relative importance of each item which will inform the design). Here are some simple examples from an e-commerce website.

|  |  |  |
| --- | --- | --- |
| Page | Element |  |
| Homepage  / | 1. Hero branding area  2.Featured product section  3.Browse Category boxes  4.search from |  |
| Category page  /*category/:category\_name* | 1. Featured Category product  2. Product filter  3. Product list |  |
| Product page  */product/:product\_id* | 1. Product image  2. Product tittle  3. Product description  4. Add to cart button  5. Sizing chart  4. Product specifications  5. Product review |  |
| Shopping Cart page  */shopping-cart* | 1. List items in cart  2. Proceed to checkout button  3. Ability to change quantities in cart |  |

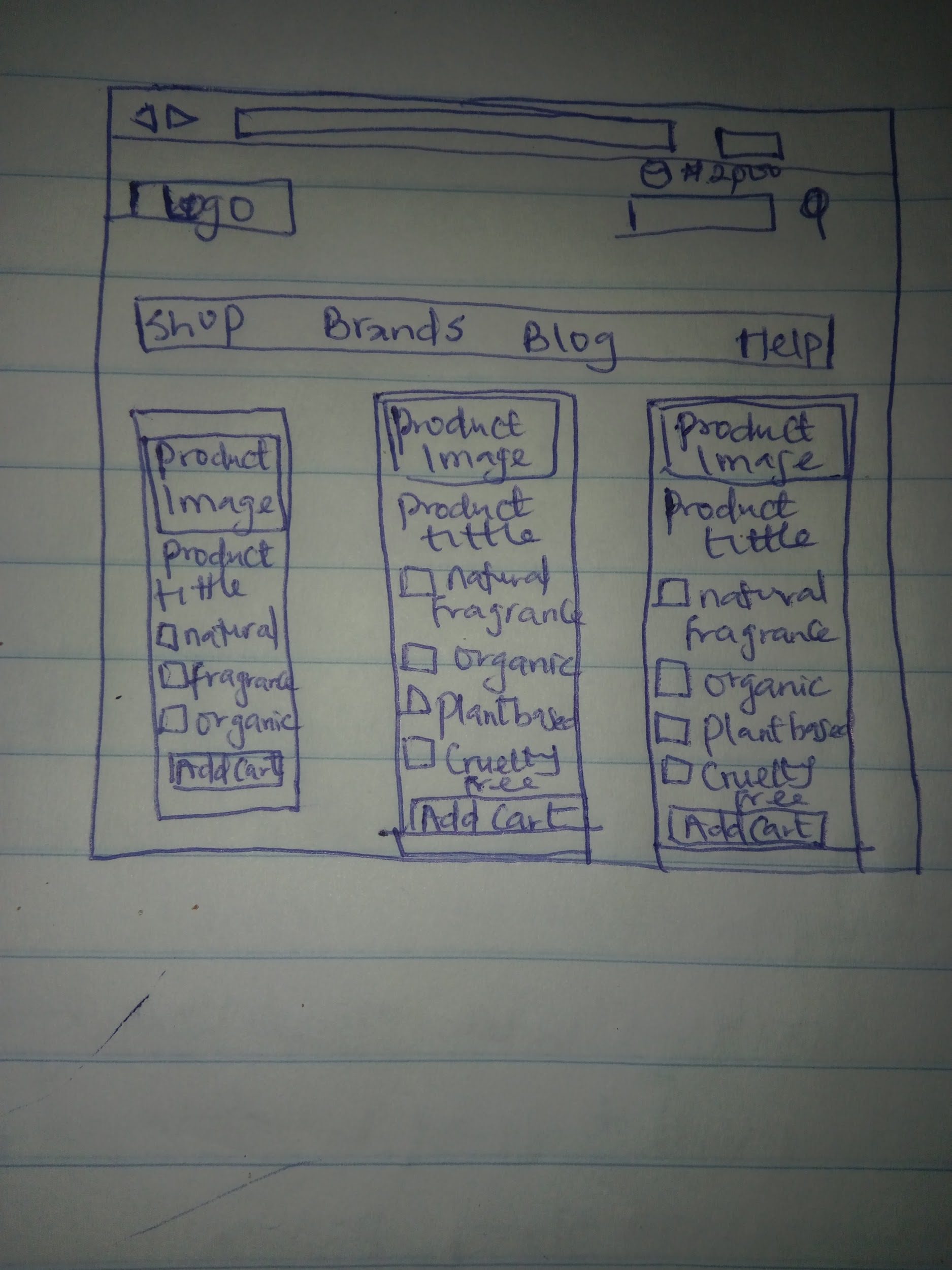
1. **WIREFRAMES**

Wireframes are simple page layouts that outline the size and placement of elements, features on a page. They are generally devoid of color, font styles, logos or any design elements.

Think of a wireframe as a blueprint that shows you the location of elements set apart from the design of those elements. Issues will often surface that may be only noticeable via a visual tool. I often find myself moving or even removing, features of a website when I get to the wireframing stage.

Due to the iterative nature of the process, I will sketch the first few attempts onto a notepad and then move into a simple prototyping tool like Balsamiq. The figures below show you the progression of my wireframes from paper sketches to a final version.

Figure 2: A Sketched wireframe of the product comparison table on the website



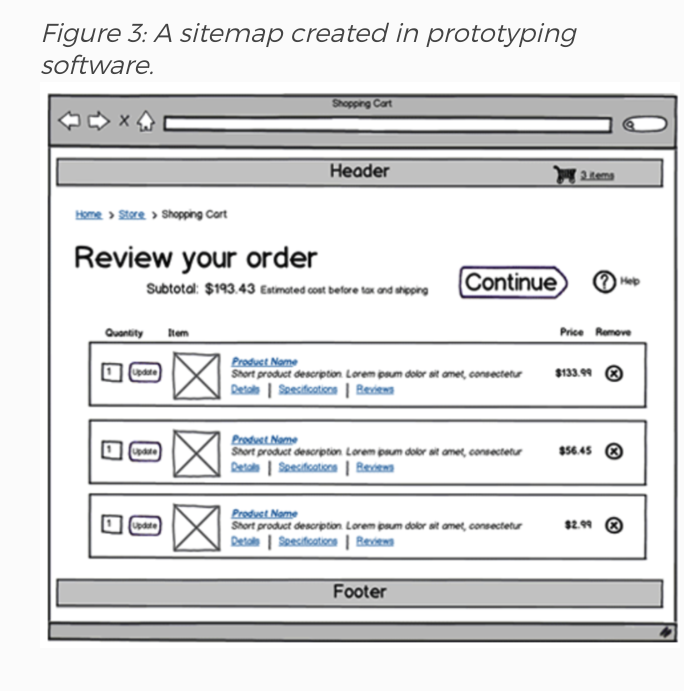
**A sketched wireframe.**

Wire framing is probably the most time-consuming step of this process and for some simple projects, it may be overkill. For complex projects where serious design thinking needs to happen, wireframes are an indispensable tool.

As the Product Owner, you may not have experience with wireframing, but I would encourage you to try doing it yourself. Doing sketches on paper and taking a picture with your phone can work remarkably well. I find that when entrepreneurs have taken the time to learn how to wireframe and deeply think about how their product will work, their development projects have gone more smoothly.

That said wireframes are an optional part of the ‘bare minimum’ framework because it can be shifted to design phase of a project. When a Product Owner provides us with all of the other information in this article, we have enough information to prepare a reasonably accurate estimate of the effort it will take to build the product and get started.

Figure 3: A sitemap created in prototyping software.



1. **NON-FUNCTIONAL REQUIREMENTS**

While the bulk of the product requirements document defines how the software will function (functional requirements), this part of the document defines requirements that may be important to your business, but are not about how the software itself functions. This is a place where you can communicate any special parameters that the developers will need to take into consideration. Here are some examples:

* The application must be built in Ruby on Rails
* The application must be hosted on AWS
* The application must use Stripe for payment processing
* The application must work in all modern browsers
* The application must be responsive (work well and look good on all screen sizes)
* The application must be able to support 1000 simultaneous users

1. **RISKS**

If there are any significant, known risks that the project faces, it is a good idea to document them. Frequently, it is a good idea for a development team to try to tackle risky parts of a software project first so you can know early on before you invest too much time and money if a particular feature is not feasible.

Here are some examples:

* Our predictive recommendation engine, which is a key differentiator for our start-up, may be difficult to code.
* Our business account may not be approved with Stripe.

9. **FUTURE ITERATIONS**

While you are writing the product requirements document, you should constantly be thinking “Is this absolutely necessary for my MVP?”. If a feature is a good idea and important to the long-term plan for the business, but doesn’t make it into the MVP, you should document it briefly in the future Iterations section so that it is not forgotten and developers can make sure the application can be extended in the future to support these key features.