Best practices when writing product documentation

Product documentation is crucial to get your product ready for release; it also helps developers to use your product and solve issues as they come up. Well-written docs help users to understand your products and its features better, which can help boost adoption rates.

There are different types of documentation, varying in the details you can add or even the docs' end users, which includes product, user, process, project, system, and system admin documentation.

You can also differentiate documentation in regard to the goal of the doc. For example, a doc can be a how-to guide or a tutorial meant for devs to learn more about a feature.

No matter the type of documentation you want to create, here are some best practices we hope will guide you through every single one of them.

Best practices	Breakdown
Add essentials elements to the product doc	While every product documentation is different because no product is the same, there are a few elements that most, if not all, docs, have. These include:

	• A definition of what the product is and what it does, • The features available on the product, • If it’s an API, the function’s parameters, • The product specs and the system requirements, • Instructions to set up the product, including configuration, • Answers to frequently asked questions, • Use cases, and • Troubleshooting information.
Be clear	Make sure your documentation is clear, concise, and useful – and, of course, easy to research and find. This helps your docs to stand out and, most importantly, to give developers the answers they’re looking for. Some **principles of straightforward, engaging language** include: • Writing for your reader, not yourself (in this case, the end user of the product). Who are you writing the doc for? What do they need from it?

	• Make use of the pronoun 'you', which helps users to identify more easily with the content. • Offer an overview of your main points before going into details. • One paragraph = one idea. Don't mix different topics in the same paragraph, as this can muddy your content. • Avoid passive voice when possible. • Keep your sentences short. • Don't publish your documentation without having your content proofread by someone else. It's easy to miss grammar mistakes, typos or repeated words, for example, since your brain already knows the sentence and instinctively 'gives' you the correct version.
Create a table of contents	A table of contents can help users to locate the information they need without having to search the whole document from top to bottom. Make sure you hyperlink your table of contents as well, ensuring the content is searchable and quick to access. **Accessibility** is another important issue, so keep in mind: • All the information is easy to find,

	• The data looks good no matter the device people use to access it, • Your doc is always up-to-date, • Add alt text (AKA, alternative text) to images, and • Consider screen reader software. These are just some ideas to get your started, so make sure you research possible accessibility options when creating your doc.
Include a README file	README files are crucial to explain to people how they can install or use the product, as well as helping them to navigate the doc. Make sure to include a description of the product, installation instructions, and even a simple tutorial or step-by-step guide. You can create your README file any way you want, including using an online tool to facilitate the task!
Keep an eye on the prize	Make sure you're focused and always keep your goal in mind – do you want to provide a step-by-step doc, a tutorial or a how-to guide? Remember this as you're writing your documentation, since it will help you to add value to the content and ensure it answers the questions it needs to answer. In essence, your documentation should be action-oriented so that every single point you make is

	clear, understandable and genuinely helpful.
Organize your content	Make sure your product documentation is organized as well! There are several ways to go about it, including categorizing the information, linking to README files and installation steps, and reviewing the documentation regularly. To make the job easier, simply keep in mind that your content needs to be accessed at any time and from anywhere, should be easy to understand even if someone is completely new to the product, and it needs to be shareable too. Other things you can do to keep it all organized is to: • Make sure to add **headings** to break up the content and to facilitate navigation between topics. • Create **lists**. They provide a simple structure that users can follow, are easy to both read and write, are easily shareable, keeps you organized, etc. • Consider adding **tables** as well, since they have several benefits, including organizing complicated data, helping people to quickly see results, allowing you to highlight patterns, and making the content more readable. Overall, by breaking up the content, you're avoiding a wall of text (which would make it hard for readers to stay focused on what you wrote) and making the content more digestible and easy to

	understand.
Choose the right format for your documentation	Choosing the right file format for your docs is also important. Think about your audience and where the documentation will sit, and decide from there what works best for everyone. There are **many different types of formats**, but here are a few to help you narrow your choices down: • PDF • DOC • CHM (or Compiled HTML, which is often used in software documentation) • Online documentation (for example, on GitHub) Online docs are hugely popular because of how versatile they are and the benefits they offer. For instance, readers can access them quickly with a browser and internet connection, and you're also able to add different types of media, including videos.
Add visuals	This will not just break up your content and make it more visually appealing, but also help people to understand information better. Everyone learns differently and having images or videos to explain a topic or a piece of code can help developers to find your content useful. Visuals can be anything from tutorial and how-to videos to GIFs, graphs and diagrams – you can opt

	for several and spread them throughout the product documentation. You can also add audio content. Multimedia elements are useful to clarify instructions, showcase processes, communicate knowledge that can be difficult to put into words, and so much more.
Implement a logical flow	It goes without saying that your documentation should have a logical flow – you don't want to start by showing advanced features when you've not explained the basics yet. Your doc's hierarchy must have a logical flow that helps developers learn how to use your product.
Ensure good design	While the content of your product documentation is more important than the way the doc is designed, you shouldn't underestimate the importance of good UX either! It can help to increase time on page and boost your chances of devs actually reading through your entire document (which we all know is rare…). Some **good practices** include: ● Giving developers a document they love to look at, ● Making it user-friendly, ● Avoiding long paragraphs,

	• Taking full advantage of white space, and • Being consistent with fonts and font sizes.
Links to further resources	Another thing you might want to consider doing is linking to other resources. This can be any company website pages you believe are relevant, links to tutorials and FAQs, links to user forums, contact lists for further support, etc.
Add a 'getting started' page	This can help guide developers when they first come into contact with your documentation. Where do they even start? A short and snappy guide can make all the difference, as it provides an overview of what to expect and can even link to other sections of the documentation.
Contextualize everything	When explaining your product, use snippets of code, API endpoints, parameters, and more, to make sure there's context for everything. This way, readers will find it easier to keep track of your explanations. You also don't have to keep repeating sections of your documentation if you want to refer to them. To streamline the doc, create references to different sections, link out to them, etc.
Add a use case if	Case studies are excellent ways to showcase that your product or feature actually works. Developers

possible	trust the opinion of other devs, so having proof that your product truly offers the results it promises can go a long way to increase your adoption rates.
Test your doc and get feedback	It'd be easy to simply write it and put it out of your mind (especially if it's a long product doc that took you ages to write), but it's critical that you test it out first. This is because you want to make sure all the necessary information is there, everything is working properly (such as hyperlinks), etc. Asking for feedback is important as well because it helps you to improve the document before it goes live. In terms of receiving **feedback**, consider this: • Who should be giving you feedback? • Make sure people understand what you need from them when requesting their opinion of your doc. • Tell them which stage your documentation is at – are you doing a first draft or is it pretty much ready to go? • Respond to feedback and address important points. • Make changes as you see fit.

Review your doc every so often	The only way to make sure your product documentation remains up-to-date and easy to read/understand, is to review it regularly. As your product or its features evolve, so should your doc. Update it with new sections, tweak existing content and improve multimedia offerings, for example. However, only make changes if they're necessary, and not just for the sake of it. This means taking into account: ● The documents people are accessing the most, ● How users engage with your pages, ● Their on-site journey, ● The search terms they use, etc.
Make use of tools	If you're struggling to keep your documentation organized or wondering where to even start, there are some product documentation tools out there that can make the job a lot easier. Here are some examples for you to explore: ● Document360 ● GitHub

- ProProfs
- Read the Docs
- ClickHelp
- iA Writer
- Bit.ai

You can also find online tools depending on your programming language. For instance, if it's Python, then Sphinx or Numpydoc can be useful. For C++, Ghostdoc might help, while Javadoc is used for Java and Docurium for Ruby.