API Documentation for Users and Developers

By

Carley Gilmore

A Major Qualifying Project (MQP)

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Bachelor of Science

in

Professional Writing

by

Carley Gilmore

May 2022

Approved by: Kevin Lewis, Professional Writing

Abstract

This study contains research on best practices for API documentation for users and developers. WPI students with experience in programming resources like forums and official API documentation participated in surveys that assess positive and negative elements of official API documentation. The study also contains a literature review and rhetorical inquiry regarding. Best practices were recommended and implemented in a documentation set that includes a user guide and standard operating procedure related to API development. The documentation set will be used by 7Factor, a software company specializing in software delivery and cloud services. It communicates important features and procedures of a Webhooks-as-a-Service (WaaS) API created by 7Factor's pilot Webhooks MQP team for future team use.

Table of Contents

Introduction

Application Programming Interfaces (APIs) are sets of software development tools that set a standard for integrating application software. In layperson's terms, an API is a library of functionality that allows computer programs to be developed more efficiently. For instance, if a developer is making a mobile application that can provide the weather to users, they do not need to write a program that fetches this information. Instead, the developer can turn to an API that can be integrated with their software with the method to fetch and present the weather data. Often, APIs are trustworthy, credible collections of source code utilized by many developers and ever-changing as their uses extend and offer more possibilities. APIs involve a contractual relationship between the API provider and the user, where the provider specifies the functionality of the API and developers agree to use the API as described (Jacobson, 2011). An example of a notable API is Google Maps, an API that is proof of Google's mastery of geolocation and supports countless applications that rely on this mastery from car services to dating apps.

APIs have redefined software development, as 90 percent of developers use APIs with this usage expected to continue increasing at an exponential rate (Voskoglou, 2020). APIs are popular because they provide preexisting libraries of software that can communicate with a developer's program to save time and resources. Without complete, consistent, and sufficient documentation, APIs can be overlooked. It is crucial that technical writers and developers investigate what makes poor technical documentation and capture the details that users and developers need. An inconsistent API tends to require more documentation, as it needs to outline and explain how to overcome the inconsistencies (Watson, 2014). When an API requires a sizable amount of documentation, there is a greater chance of errors in presenting information to developers clearly and consistently. Ignoring these risks can create documentation that is inconvenient and tedious for developer use. As developers combat inconsistencies in their APIs, it is encouraged that open communication with technical writers is maintained to keep the corresponding documentation as organized and updated as possible.

The goal of this Professional Writing Major Qualifying Project is to communicate best practices in technical documentation supported with research and a documentation set based on an API that I have built in my Computer Science (CS) MQP. As the usage and value of APIs have increased significantly, it is imperative that technical writers and developers have working documentation that addresses the user's needs. This project includes useful materials to be used as references for creating standard operating procedures (SOPs) and user guides for proper user and schematic documentation.

Background

The purpose of this project is to highlight elements of both user and reference documentation including API documentation that helps software engineers integrate programmatical solutions.

2.1 The Importance of User Documentation

There is a common misconception that user documentation for software is unnecessary and only relevant if it is to compensate for a software's defects. However, it can be argued that the need for user documentation in the area of software is always prominent since unlike operating physical tools that directly change the material world like assembling furniture, computer systems are self-contained worlds of their own where a user's actions have a direct result only on the software environment itself. The moving parts of software such as operating systems and other processing environments involve intricacies that determine the different ways a user can use a software system. To use software at its best and full potential, users need user documentation to understand how their own machine behaves and links with the software such as dependencies and specifications (Loggem, 2013).

2.2 The Challenges of User Documentation

User documentation comes with its own unique set of challenges, as it is difficult from a design perspective to address every expected user need from a software system. Documentation designers' expertise and skills have not been able to develop at the same speed and quality as the software that they document. This observation alone indicates the significant need for evidence-based information on proper design of user documentation for end users (Loggem, Interaction with User Documentation). A study conducted on users' preferences related to software help

systems indicates that there is a clear issue with deliverance based on the desired level of technical complexity in procedural explanations (Novick, 2006). The study results demonstrate that explanations frequently fail to meet the needs of the audience; for example, 72 percent of participants expressed that the documentation they used for the same software system was either too general or too specific to help them in their tasks.

2.3 The Importance of Reference Documentation

Improving reference documentation would allow for significant efficiency, as clearly outlined programming procedures can help developers maintain their productivity. According to IEEE (Institute of Electrical and Electronics Engineers), one minute of distraction from flawed documentation can cause a developer to lose their "flow" and take at least 15 minutes to regain it (Watson). Addressing common mistakes in reference documentation and implementing designs that maintain the focus of developers allows for better experiences in software development.

Research Methods

This project includes the use of surveys as well as a detailed literature review to communicate the need for better documentation conventions in technical settings as well as the solutions to improving one's writing for user and developer documentation, especially API documentation.

3.1 Surveys

To collect data regarding the effectiveness of computer programming resources and API documentation, I created an anonymous Qualtrics survey for WPI students who have experience using software documentation such as user guides, API documentation, and developer forums. WPI students in areas involving computer programming (Computer Science, Robotics

Engineering, and Electrical Computer Engineering) are encouraged to take the survey. The survey collected student feedback on experiences with resources for programming including experience with forums and official API documentation. It quantified the level of comfort that students have with documentation, specifically API documentation or any level of schematic documentation needed for integrating preexisting software solutions in program development. Appendix B contains the qualitative survey questions. Appendix D contains the quantitative survey results with revised survey questions.

Using an advertisement that specifies the goal of the survey, I shared the survey with the student body via email and on WPI social media platforms such as WPI Slack and Discord channels and the WPI subreddit. I analyzed the survey data with the following evaluation objectives:

- What qualities are favorable in sources that help programmers integrate technologies?
- What are the effective qualities of good technical documentation?
- How does the structure and design of the information presented affect the reader's experience?
- To what extent is API documentation helpful?
- Do developers turn to forums because official technical documentation is not organized enough for their specific needs?
- Are the poor experiences with API documentation or official technical documentation significant? How can this documentation be modified to fit developer needs?

In assessing these objectives, I gained a better understanding of how users and developers are affected by documentation intended to help with integrating a range of technologies. I learned which qualities are favorable across different platforms and investigated this further to navigate how to communicate instructional content to users and developers. The evaluation questions helped me to assess which qualities of online forums are suitable and applicable to official technical documentation sets. I also collected applicable information on what specific qualities users look for in technical documentation that retain a user or developer's attention and allow them to effectively integrate technologies.

These objectives allowed me to observe different perspectives and experiences with technical documentation. In addition, I kept note of the effects of stylistic factors such as layout and organization of documentation and investigated any specific API documentation that is deemed successful or unsuccessful listed by survey respondents. Getting an idea of what users look for in technical documentation, especially in this current, ever-changing age of technology helped me to communicate guidelines for best practices in technical documentation. I leveraged the survey to provide statistics that measure the problem of poor technical documentation and highlight the importance of thorough, organized, and up-to-date documentation for APIs and other technologies.

3.1.1 Qualitative Survey

To collect the constructive opinions of software developers on technical documentation, I first released a qualitative survey to allow for as much explanation as possible on each question, including yes and no questions. This was the first survey that I distributed on Qualtrics to collect qualitative feedback on experiences with resources for integrating technology. There were textboxes on each answer choice to encourage elaboration on questions regarding experiences

with specific API documentation/technical forums. Asking for specific examples and favorable elements of API documentation presented an opportunity to gain more perspectives and insight on the extent that API documentation currently helps users.

3.1.2 Quantitative Survey

To gather more statistical data and encourage more participants, I opened another version of the survey with straightforward, selectable answer choices. Some of these choices included multi-option fields that allowed for multiple responses. The questions regarding positive and negative experiences with API documentation directly outlined the characteristics that users might find to be positive and negative as select all that apply answers rather than examples for text responses. This change in formatting the survey made it a much quicker survey to fill out, making it appeal to a larger group of participants. It also gave me the ability to clearly visualize numerical data that indicates what many respondents look for and encourage in documentation.

3.2 Literature Review

To research and gather the best practices in technical documentation and create a documentation set that exemplifies said practices, I conducted a literature review. The review helped me to gain insight on topics within technical documentation such as API documentation that are relevant to the scope of my project. I used databases such as WPI Gordon Library and Google Scholar to find credible sources that showcase the need for consistent and clear documentation and the challenges in developing such documentation.

3.2.1 Educational Sources on User and Developer Documentation

I searched for sources that educate writers on key elements of good user and schematic documentation. Some of the key terms and phrases I searched for include:

- Technical documentation
- Reference documentation
- User documentation, challenges, best practices
- API documentation, technical writing for APIs

These sources will address the feedback from the survey respondents and aid in the formulation of my results and recommendations.

3.2.2 Examples of API Documentation

After receiving examples of API documentation in the survey, I researched specific API tips for documentation such as Apple's explanations on how to use DocC syntax for creating documentation for your own API complete with tutorials. I also searched for terms and phrases on Google for specific examples of well-known and frequently used APIs and their official documentation. Some of these searches included:

- "Effective API documentation"
- "API developer experience"
- "Examples of API documentation"
- "Expert advice on API documentation"
- "Interactive, engaging [API documentation]"

The purpose of these searches is to find examples that I can reference in my results and recommendations as good documentation. The intent is to investigate good references to follow in creating documentation that is inclusive and addresses all audiences.

3.2.3 Sources on Standard Operating Procedures

I searched for terms and phrases on Google for specific examples of well-known and frequently used APIs and their official documentation. Some of these searches included:

- "Documentation in API Use"
- "Standard Operating Procedures in Building APIs"
- "Examples of API Standard Operating Procedures"

I researched these topics because I needed to learn more about how SOPs should be created for extending an API, since that would help in creating an example SOP that would essentially allow employees to understand and extend the API that my team designed.

3.3 Rhetorical Inquiry

With sources gathered from my qualitative survey, I conducted a rhetorical inquiry in which I examined examples mentioned by developers more closely. I looked to see if those example documentation aligned with best practices mentioned in articles from my literature review. In addition, I analyzed sources and example documentation through the lens of developer feedback on proper methods of API documentation. After I recorded and interpreted the results of the qualitative data, I performed a synthesis of the results and critiqued sources based on the conclusions I formed. I conducted this literary analysis to determine consistent patterns throughout my research and communicate these findings in my documentation set.

3.3.1 Surveys and Literature Review

First, I searched for examples mentioned by developers more closely. Example documentation provided by survey respondents includes:

- "Apple's developer documentation"

- “Discord API documentation”
- “PyGame documentation”
- “Microsoft documentation, FileDialog windows”
- “Robot Operating System (ROS)”

After searching for each documentation set, I scanned for patterns to get samples of the documentation that are favorable or unfavorable to the developer. After I pinpointed the patterns, I referenced the qualitative survey results and kept track of them for my literary analysis. I included this methodology in order to create a method of analysis to employ in the rhetorical inquiry.

In my rhetorical inquiry, I selected example documentation from my literature review on educational sources for API documentation and examples of API documentation. I used the conclusions drawn from my survey results to do the inquiry. The method allowed me to highlight the features that did not align with the survey results on favorable practices in documentation. I also was able to find where educational sources lack information on specific developer needs such as sample code and tutorials. I analyzed how sources I found in my literature review meet developer feedback from surveys and how model example documentation mentioned from respondents aligns with best practices communicated in studies on effective documentation.

Results

This chapter features the results of my surveys, literature review, and rhetorical inquiry mentioned in my *Research Methods* section.

4.1 Surveys

The qualitative and quantitative results of the surveys that I distributed can be found in Appendices C and D. The results of the qualitative survey and the respondent turnout resulted in the distribution of the quantitative survey. The quantitative survey has the same questions as the qualitative survey but is structured in a way the minimizes the need for lengthy text responses.

4.1.1 Qualitative Survey

The results of the qualitative survey include feedback from developers that indicate the preference for official API documentation with the acknowledgement that the official documentation can be lacking in some areas. One of the main things I noticed from Qualtrics that played a role in my restructuring and distribution of a new survey was that 27 students started to take my survey but did not submit it. 21 students responded to the survey with a significant amount of qualitative feedback on experiences with API documentation and technical forums. The following sections represent noteworthy feedback from respondents of the qualitative survey.

4.1.1.1 Commonly Used Programming Sources

For the first qualitative survey question, all 21 respondents selected Stack Overflow as a notable, useful source for integrating technology. 85 percent of the respondents marked official documentation as a source used, with most noting that while one may experience difficulties such as incomplete, incorrect documentation, the official documentation should not be ignored (Figure 1).

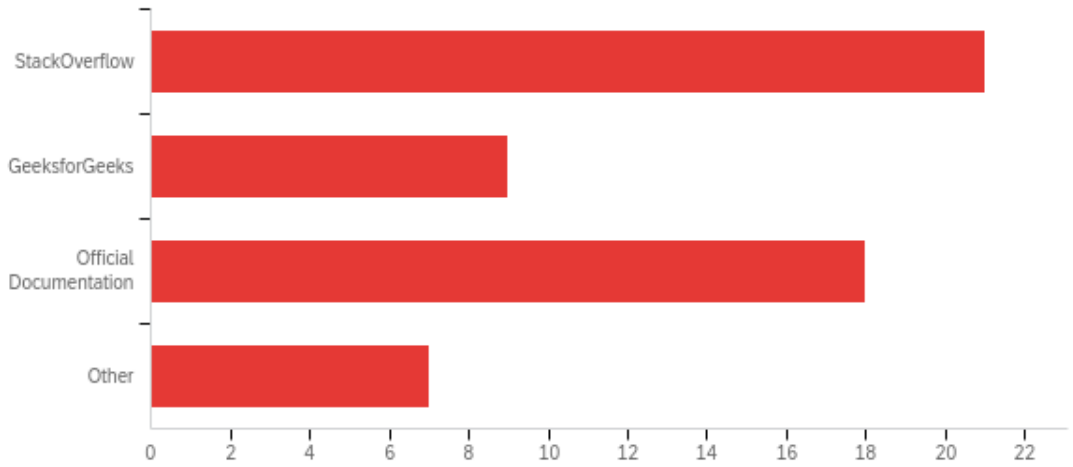


Figure 1: Qualitative Survey Question 1: What sources do you use for integrating technology/programming errors/troubleshooting? Ex. StackOverflow, GeeksforGeeks, etc.

Three respondents did not mark official documentation as a primary source that they use for integrating technology. One of these respondents clearly noted that they had used API documentation before but expressed a preference for Stack Overflow over official API documentation. With Stack Overflow they feel "almost guaranteed" to find a solution to their issue, whereas some official documentation "doesn't always explain" or often lacks example code that uses specific functionalities.

In the following question, respondents addressed what they particularly like about these sources. Some significant quotes include:

"These resources provide sample code to better understand the solution and how it works."

"Stack Overflow is good for common problems that have easy solutions. Official documentation usually works for everything else, except if the documentation is very bad (which it sometimes can be)."

One respondent mentioned the distinct usages of forums opposed to official documentation:

> Stack Overflow gives direct, concise answers--useful for learning "tricks" and solutions to small problems. Official Documentation varies project by project but can be useful for learning new capabilities of the software. This helps with planning future projects, and deepening engineering of larger projects.

Lastly, a developer shares their experience with each method of extracting information for aiding in technical projects:

> Stack Overflow: A ridiculous amount of information. The problem I'm having has usually been answered. GeeksforGeeks: Good as a tutorial resource, which will show multiple examples of an algorithm implementation or use of a library. Official Documentation: When I'm deep into a project and trying to get a library to work in a very specific way, official docs are my best bet. GitHub: When nothing else gives the results I need, I'll read through source code.

In the qualitative survey, participants did not explicitly rank whether they would use one source over another. Based on the results, every respondent mentioned using at least two sources and stated the qualities that they favored. Some respondents as shown above referred to the potential drawbacks, demonstrating that there are varied experiences with different forums and API documentation. The sample quotes above highlight that Stack Overflow and other forums are convenient tools for quick solutions to issues. Respondents noted that forums complement the specific, thorough documentation that companies provide for APIs and other technologies.

Relevant documentation sources mentioned in the “Other” category include:

- GitHub: direct source code and comments
- Stack Exchange: forums including Stack Overflow

- Tutorials Point: slides that cover the basics and multiple ways to solve problems
- "Random" Googling: sites that might include an answer
- Emailing developers directly: experts who play a direct role in API development

The sources outlined in the "Other" category provide results that largely stem from the listed sources not being able to meet developer needs. They are also useful and have been discussed in the sample quotes above and raw data in Appendix C.

4.1.1.2 Positive Experiences with Forums

Based on their experiences with forums, respondents tended to favor forums like Stack Overflow because they are widely used by other developers who often have the same questions and provide up-to-date working solutions. Respondents communicated that Stack Overflow is direct, concise, and practical; it is good for questions that are not mentioned in official documentation. Helpful features referenced from Stack Overflow include filtered answers in order of credibility, working examples on how to solve a problem, and visible indicators such as a green checkmark to denote an answer's validity. One piece of the positive feedback for GeeksforGeeks, another forum-like portal for programmers, is that the hub includes informative, easy-to-follow examples and tutorials. One respondent mentioned that GeeksforGeeks "provides multiple solutions for multiple languages and explains the topic at the top of the page." These functionalities make content more accessible and easier to read for different developers.

4.1.1.3 Negative Experiences with Forums

While the responses regarding forums are overwhelmingly positive, there are a few negative experiences reflected in the qualitative data that are important to review. In the second question of the qualitative survey, participants were asked what they liked about their experience

using their selected forums and/or documentation. In the results of this section, some respondents addressed both their likes and dislikes when it comes to forums. The negative experiences primarily outlined drawbacks or inconsistencies that predispose developers to avoid consulting a particular forum over official documentation or other forums. A respondent compared Stack Overflow to that of a "scavenger hunt," indicating that it is not always the case that the answers on a forum will be relevant to the application being used.

Most of the critical comments targeted GeeksforGeeks as a useful source. Another respondent mentioned that while they use GeekForGeeks, the source lacks "real application with real fixes instead of theoretical issues." Another respondent noted that the forum "feels much less curated and SEO'd (search engine optimized)." In other words, the results of a search on the GeeksforGeeks can lack detail and organization, failing to address developer issues. Lastly, GeeksforGeeks was mentioned to be an avoided source that lacks "official detail that would most likely be clarified in documentation."

4.1.1.3 Positive Experiences with Official API Documentation

16 respondents noted that they have used official API documentation for their project work. One respondent noted that while forums can be occasionally helpful, reviewing official documentation from a company or owner of an API is the "most reliable" way to find information. Their reasoning is that everyone has a "slightly different application they need the problem solved for." Another respondent also expressed that "nothing beats the official documentation," but Stack Overflow was worth noting as a useful source for working with new APIs. While it is difficult to know whether API documentation will be useful or not, if written correctly, it is often the primary source for learning capabilities of software.

A respondent shares their stance on what makes official documentation reliable:

> The best documentation must include the purpose of a specific implementation, examples of how to use it, common problems, and parameter details. Without these core pieces, the official documentation can become less relied upon, forcing you to ask questions and gain clarification from sources like Stack Overflow.

Another respondent said that their experience with official documentation has been mostly positive. They compared the documentation to the textbook of a project, acknowledging that "in theory, everything you need to know will be in one place," and "in practice, this isn't always true, but it usually is." The existence of the official documentation alone was mentioned to be useful in and of itself, as the intended use cases are noted with "full functionality and behavior description." It is described as "more niche, but highly focused and effective when done well."

Relevant qualities and features mentioned when asked about positive experiences with documentation include:

- Well Explained Examples such as videos or written code
- Up-to-date, Consistent, Correct, and Complete Content
- Well-organized, Clear, and Readable for seamless extraction of necessary data
- Easy-to-follow project scope and abilities

4.1.1.4 Negative Experiences with Official API Documentation

In addition to positive experiences with official API documentation, developers responded by providing negative experiences with API documentation. One respondent experienced cryptic formatting of documentation, stating "sometimes the format of the documentation is just hard to read, especially notation used for parameters, etc." Another respondent reported annoyance with out-of-date documentation. They specifically mentioned the

trouble with obsolete material, “especially when a service hosts multiple pages for documentation.” In these cases, the respondent mentioned that the information is incorrect and supports “deprecated” versions. This feedback encompasses some of the drawbacks in overall experience with API documentation.

Characteristics of poor documentation expressed when prompted to discuss negative experiences include:

- Incomplete, Obsolete, Incorrect, and Vague material
- Lack of examples or explanations
- Repetitive and hard to follow organization
- Cryptic, hard to understand (from a lack of explained prerequisite knowledge)
- Supports depreciated versions of software
- Inconsistencies between the code and the documentation
- Hard to read formatting and notations

4.1.1.5 Elements of Forums in Official API Documentation

Most of the participants of the survey indicated that there are elements of forums like Stack Overflow and GeeksForGeeks that can and should be included in official API documentation. One respondent mentioned that in a set of documentation for PyGame, a cross-platform collection of Python modules for writing video games, there was a useful way for developers to comment on API sections. This ability opened areas of discussion about how to use library features and was often useful. Another respondent mentioned the practicality of including written code excerpts that are all too common in programming forums like Stack Overflow. They revealed that “examples and situations of not just how, but when to use certain functions” should

be adopted in official API documentation. It can be acknowledged in both the literature review and the survey data that programmers like tangible ways to play test their code. A respondent mentioned that only after exhausting the common online forums for the correct solution, they turn to official documentation. The same respondent stated, “when nothing else gives the results I need, I’ll read through source code [from GitHub].” The online community of open source is vast, and often, developers will reach out to this community and look through workable examples on GitHub, a popular version control software management tool that inspires collaboration and the most innovative programming solutions. Example programs and pre-existing code bases are among the most communicated desires from programmers in technical documentation since these are featured in forums. While most respondents answered that it would be beneficial to have elements of forums in official API documentation, there were some instances where respondents expressed some concern with official API documentation having a forum-like nature in style. One respondent mentioned that community-driven documentation could be a “nightmare” due to its disorganization and that it would be good to keep these resources separate.

4.1.1.6 Lack of Defined Prerequisite Information in Documentation

While most of the respondents answered that they did not recall any instances where prerequisite knowledge was vague or not mentioned, about 44 percent of the respondents indicated that they have had experiences where there was a notable lack of prerequisite knowledge defined. Three respondents made a point to fill out additional qualitative data in this question and specifically mentioned that they encounter a lack of prerequisite knowledge “all the time.” One developer mentioned that it seems that “the assumed or prerequisite knowledge is often implied and rarely explicitly stated” in the documentation. Another respondent mentioned

that a lot of times they have seen methods that are listed as existing methods without context for “non-experts” to comprehend.

A respondent expresses their experience with referencing other libraries implying that developers should know the material:

> There are times where something will reference another library, or standard library as if one should just know it. Or it will reference things like "use try catch with the following block", but their block doesn’t have that, so you spend twenty minutes looking up syntax and realizing you need to add a null return because they use bluebird, or some silly thing like that.

In the response above, the developer voices their frustration about having to look up syntax, something that is assumed prerequisite knowledge, and wasting time doing so since the documentation lacks clarity and readability. The data of this question indicates that when users are aware of the knowledge, they should have prior to integrating a technology, it can save time in the long run.

4.1.1.7 Respondent Experience with Creating APIs

Eight out of fifteen respondents answered that they have created their own API before. Questions that related to respondents who have created their own API ask about sources developers refer to and if they feel there are adequate resources for creating an API. Some relevant sources and methods mentioned for creating an API include:

- Open API 3.0 Specification
- Swagger Hub
- Stack Overflow

- Referencing official documentation
- Maintaining a consistent style with thoroughly commented code
- Example projects and pre-existing code bases

Some of these sources collected from the qualitative survey were used in the quantitative survey to be used as example sources for developers to select if they have referenced them or not (4.1.2.5).

After providing feedback on the sources they used, most respondents answered that they do feel there are adequate resources and documentation. One respondent said that, while they responded with yes, "it also depends on which service and what you are trying to accomplish." The same respondents answered whether they would use official documentation or forums if they had to choose one option over another. The results in the following section indicate a main distinction between official documentation and forums.

4.1.1.8 Official API Documentation vs Forums

Eight percent more respondents mentioned that they would be more likely to use official API documentation over forums and other help. One respondent mentioned that they use official API documentation over forums or other help. Their rationale is that official documentation is "generally the MOST correct and updated when changes or depreciations happen." Another respondent mentioned that they rely on forums or other help because documentation is "more often than not just an info dump that is unhelpful to someone who's trying to learn." The same respondent goes on to say that "having a complete reference is important, but useless if I can't figure out how to work any of it."

4.1.2 Quantitative Survey

After the results of the qualitative survey were obtained, I distributed a quantitative survey to gather more concrete statistics on the resources that developers use and their favorable qualities. More people responded to the quantitative survey than the qualitative survey. I released the quantitative survey primarily for more widespread results and in hopes that more people with API experience would respond. The revisited question format of the survey proved to be successful as the quantitative survey garnered 52 respondents. The following sections summarize the quantitative survey data.

4.1.2.1 Commonly Used Programming Sources

For the first quantitative survey question, choices were offered as a result of mentioned sources in the "Other" category of this question in the qualitative survey response data (4.1.1.1). Respondents were asked to select all that apply and provide any other sources if applicable. More respondents indicated Stack Overflow and Random Googling as methods that they use for integrating technology, handling programming errors, and troubleshooting. Figure 2 shows that nearly 81 percent of respondents selected Stack Overflow, an option indicated more than the "Official Documentation," "Directly consulting developers," and "Other" categories combined.

Random Googling was chosen by 47 respondents as a method used.

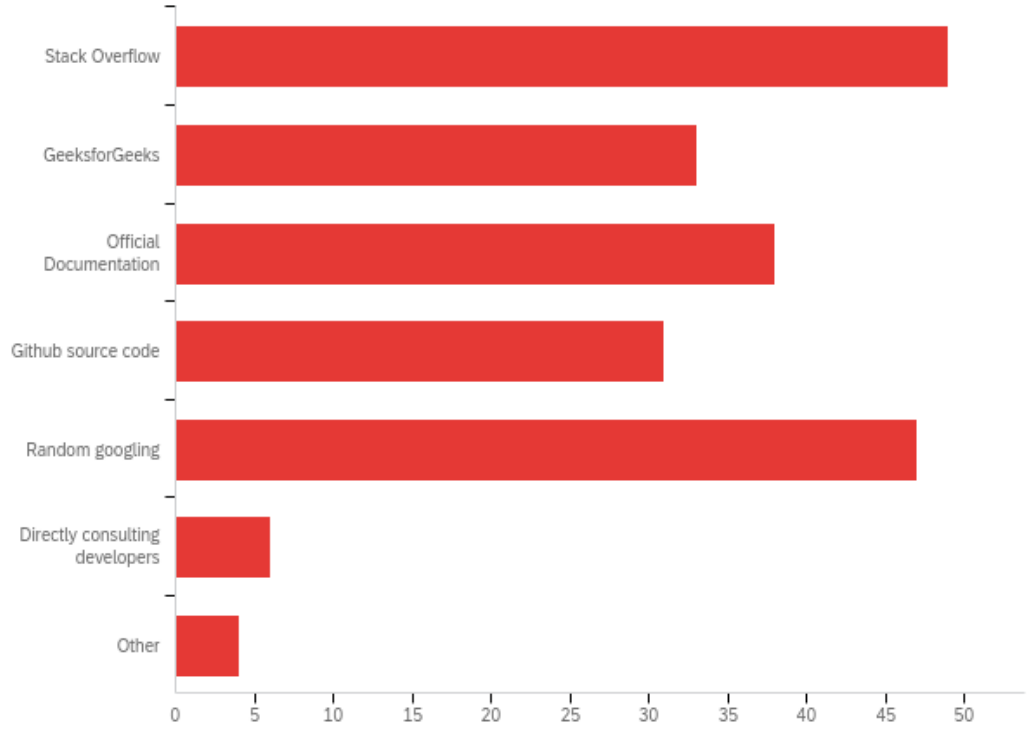


Figure 2: Quantitative Survey Question 1: What sources do you use for integrating technology/programming errors/troubleshooting?

In the “Other” category, the other sources mentioned include:

- Existing examples in the codebase
- Experienced friends
- W3Schools
- YouTube

In Figure 2, most developers reference more than one source, with a significant number of respondents opting for use of forums. When asked in Question 7 if respondents think that there are elements of sources like Stack Overflow and other forums that could be adopted in official API documentation, about 81 percent of respondents answered yes.

4.1.2.2 Ranking Sources on Effectiveness

In the second quantitative survey question, respondents were asked to rank the sources from Question 1 based on their effectiveness, particularly the likelihood that the source could help with integrating technology. Figure 3 shows the sources that were ranked as the number one most effective source.

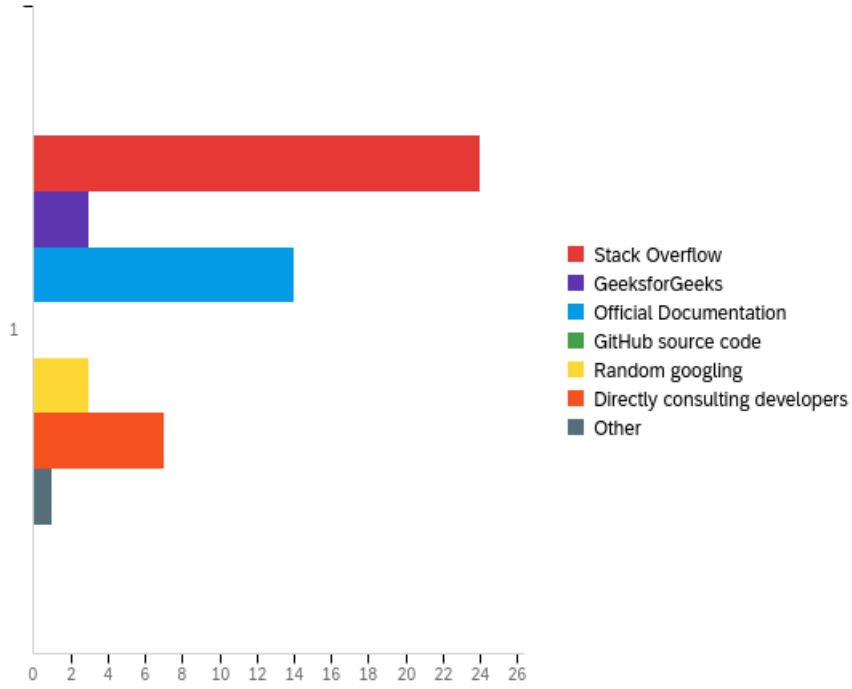


Figure 3: Quantitative Survey Question 2: Number 1 rankings based on the likelihood that a source will be able to effectively help with integrating technology/troubleshooting.

Based on Figure 3, Stack Overflow was the most chosen option for the source most likely to help with integrating technology selected by 24 respondents, nearly half of the total respondents. Official Documentation ranked as a close second as a notable option for the most effective source. The option of source code directly from the developer via GitHub was not chosen as a first choice by any of the respondents. Respondents reported that they were more

likely to get help from random googling of their specific problems than from the source code directly.

As shown in Figure 4, Stack Overflow and Official Documentation are tied as the second most useful source. GeeksforGeeks was also recognized as a useful source in the second ranking.

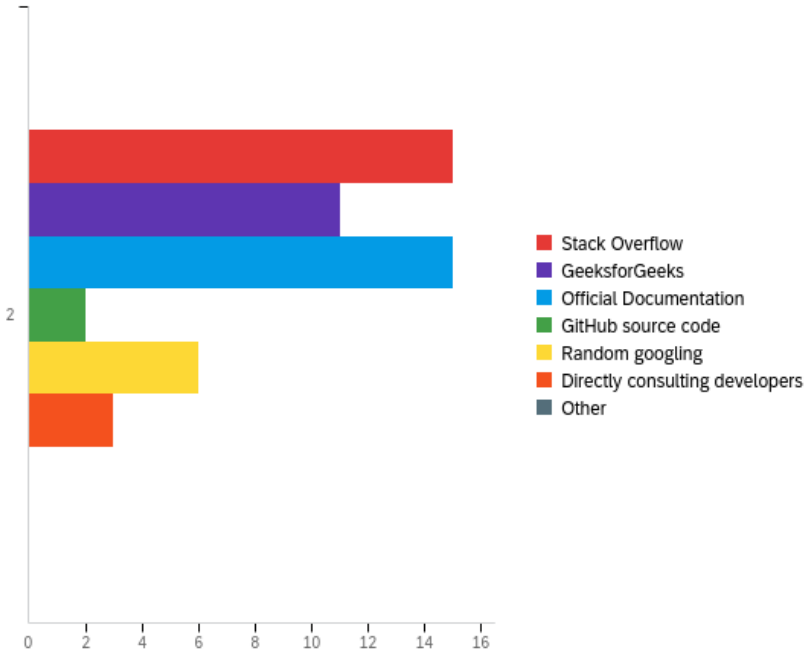


Figure 4: Quantitative Survey Question 2: Number 2 rankings based on the likelihood that a source will be able to effectively help with integrating technology/troubleshooting.

Random googling was ranked overall as the third most useful option (Figure 5). GeeksforGeeks and Official Documentation were equal in the third ranking as well as Stack Overflow and GitHub source code respectively. The four sources differ by a close margin (Figure 5).

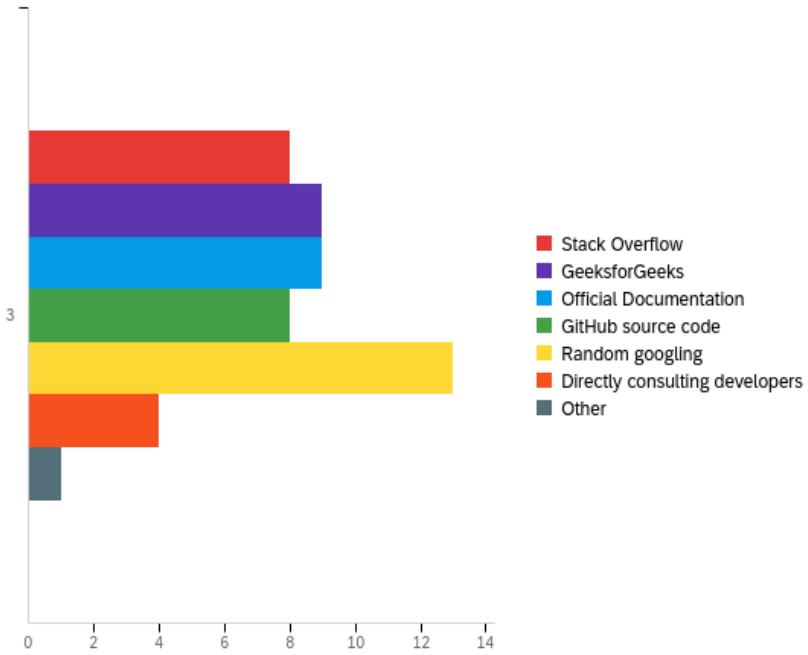


Figure 5: Quantitative Survey Question 2: Number 3 rankings based on the likelihood that a source will be able to effectively help with integrating technology/troubleshooting.

In the lower rankings, GitHub source code, directly consulting developers, and mentions in the "Other" category were chosen the most. GitHub source code was the most chosen option for both the 4th and 5th rankings. 27 respondents chose "Directly consulting developers" as the 6th ranking, just above the 49 respondents who chose "Other" for the least likely method of choice.

4.1.2.3 Positive Experiences with Official API Documentation

When asked about what applicable qualities in official API documentation have been encountered and favored by developers, about 83% of respondents selected "Well Explained Examples." As shown in Figure 6, slightly more respondents selected correct documentation over complete documentation. The least number of people, neglecting the "Other" category, chose "Consistent" as an option both encountered and favored.

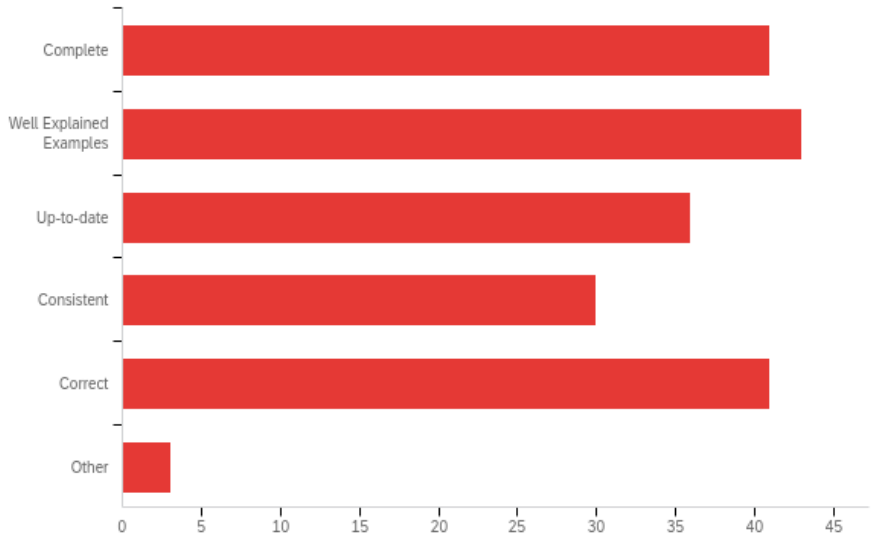


Figure 6: Quantitative Survey Question 4: In your positive experiences, what applicable qualities in official API documentation have you encountered and like?

Three respondents mentioned other positive qualities they have found in official API documentation. The additional qualities fall into more descriptive or navigable elements of documentation and include sandbox demos (prefilled editors that allow for a demo return call), parameters and their types, and easy to search format.

4.1.2.4 Negative Experiences with Official API Documentation

When asked about what negative qualities in official API documentation have been encountered and disliked by developers, nearly 83% of respondents selected “Ambiguous.” As shown in Figure 7, more respondents encountered incomplete official API documentation than incorrect or inconsistent documentation. The second most selected negative quality was obsoleteness. This data goes hand in hand with up-to-date and consistent traits as the least chosen encountered qualities shown in Figure 6.

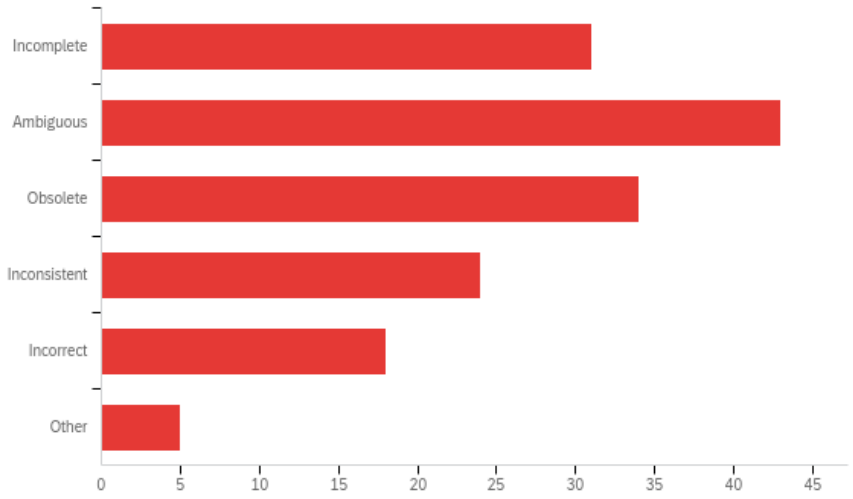


Figure 7: Quantitative Survey Question 5: In your negative experiences, what applicable qualities in official API documentation have you encountered and dislike?

Participants gave their opinion on other flaws encountered in API documentation. One respondent mentioned that they had trouble with API documentation "only including one example for complex/multi-use features" and mentioned that this quality makes the API seem one-dimensional, only able to be utilized in one way rather than being a multi-use library. Another respondent mentioned that sometimes the documentation is beyond comprehension, noting that they feel "overwhelmed with information." Another respondent expressed their distaste for auto-generated API documentation. Lastly, a participant made a point to mention "lack of examples" as a quality they especially dislike.

When respondents were asked if they could recall any instances where prerequisite knowledge was vague or not mentioned, 42 answered yes. This question, along with most other restructured questions, was also presented in the qualitative survey results. In contrast with the quantitative survey and by a close margin, more responses indicated no recollection of experiences with unclear prerequisite information.

4.1.2.5 Respondent Experience with Creating APIs

31 respondents answered that they have experience creating their own API. When asked about what sources the developers who have created APIs referred to, most respondents chose "Example projects and pre-existing code bases." As shown in Figure 8, more respondents chose example projects and forums than official specification guidelines and Swagger Hub, a notable platform mentioned in the qualitative survey for API design and documentation.

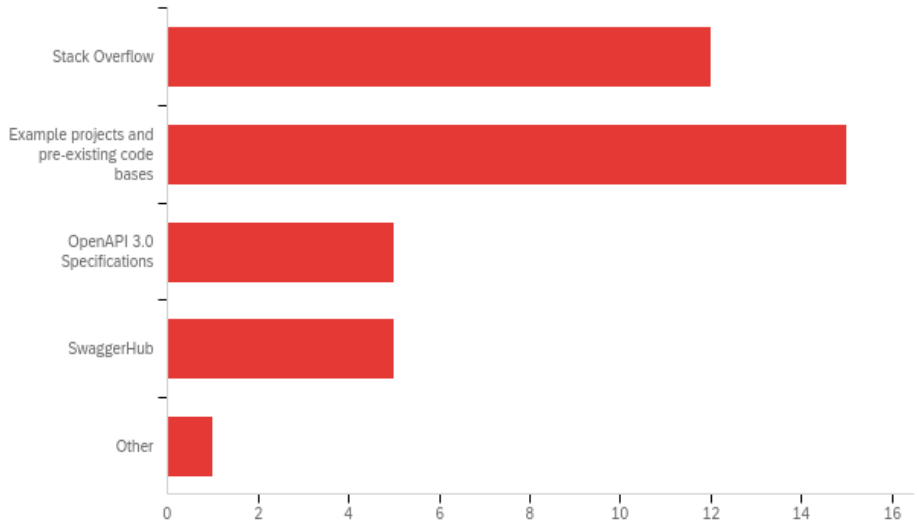


Figure 8: Quantitative Survey Question 9: What sources did you reference to create your own API?

One respondent mentioned language documentation like Flask's official site as a source referenced in creating an API. In the following question, most respondents answered that they do feel there are adequate resources and documentation when creating an API. Most respondents also mentioned that they find it easy to find the sources used to help with using specific technologies.

4.1.2.6 Official API Documentation vs Forums

In contrast to the qualitative survey data, more respondents mentioned that they would be more likely to use official forums and other help over official API documentation. Ten percent

more respondents indicated that they would rely on forums and other help compared to official API documentation.

In Figure 9, most participants who responded with official API documentation selected that they found the official source to be more credible and reliable than forums and other help. About 14 percent of responses noted that they would choose official API documentation over forums and other help because it is easier to read. About 23 percent of respondents who chose official documentation selected official documentation having more examples and tutorials as a reason why they rely on official documentation more than forums and other help.

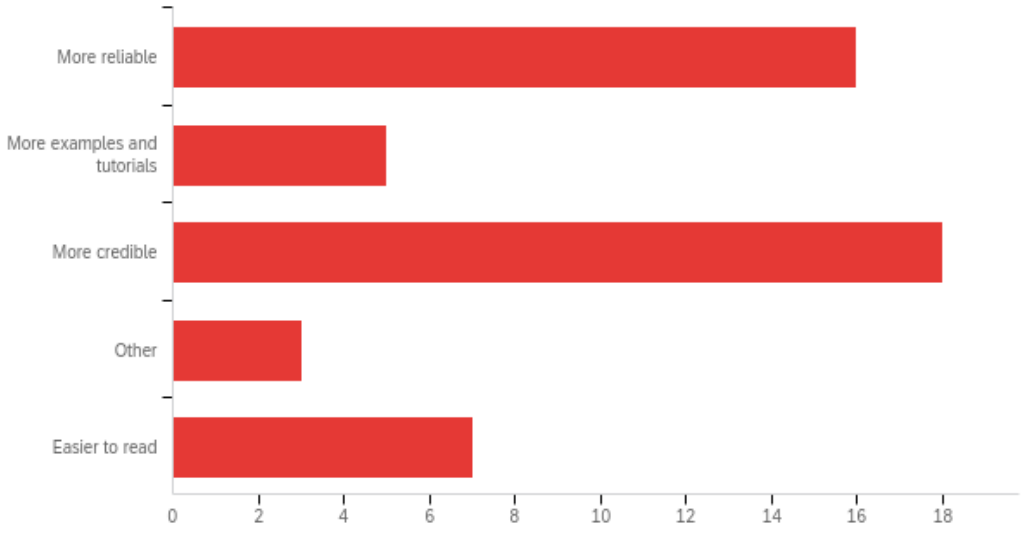


Figure 9: Quantitative Survey Question 13: What are your reasons for relying on official API documentation more?

Among the responses under the "Other" category, one developer mentioned that they find official API documentation to be easier to access. Another developer's reason for prioritizing API documentation is that it ensures best security practices. One respondent shared their experience searching for answers regarding API integration, stating that "one will spend an hour googling a question" while a shorter time browsing official documentation is often more fruitful.

In Figure 10, 24 participants confirmed that a reason why they use forums and other help over API documentation is because the other sources contain more examples and tutorials. The second most chosen reason for participants is that forums are easier to read than official documentation.

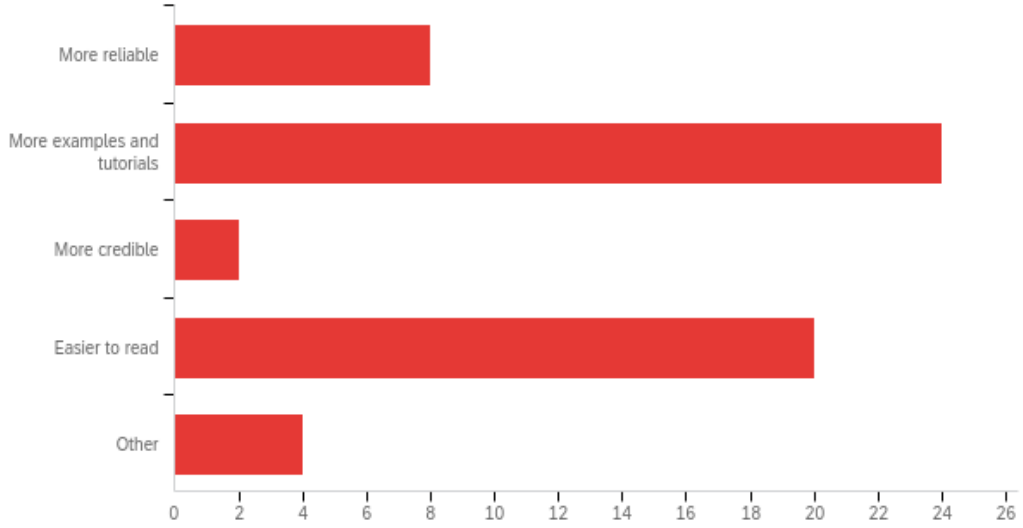


Figure 10: Quantitative Survey Question 14: What are your reasons for relying on official API documentation more?

In the "Other" category, two respondents mentioned that they believe that forums are up to date resources. Two other respondents noted that forums make it easier to grasp topics they have found difficult to understand because of the likelihood of finding relevant examples.

4.2 Literature Review

The following sections highlight the most valuable and relevant article data that I found when conducting my literature review. The content includes elements of API documentation that are similar to the survey data and best practices for writing effective API documentation and SOPs

4.2.1 Educational Sources on User and Developer Documentation

To start with, in my literature review, I collected sources that relate to general user and developer experience with documentation. User studies conducted on interaction with documentation are educational sources that draw notable conclusions on the effective aspects of technical documentation. I also examined historically how research findings in technical communication, specifically documentation in general, have evolved, or in some cases remained the same.

In a 2017 GitHub survey, 5,500 respondents contributed to over 3,800 open-source software (OSS) repositories on GitHub. Data from the survey shows that 93 percent of respondents reported incomplete or obsolete documentation to be a "pervasive problem." A 2018 ethnography on documentation references this survey, stating that it supports the notion that documentation in the open-source ecosystem is often "notoriously considered low-quality, sparely written, out of date, or simply non-existent (Geiger, 2018)." Part of this issue is that 60 percent of the contributors to GitHub codebases expressed that they rarely or never contribute to documentation. In the ethnography, 10 Docathon participants, developers experienced in writing documentation for OSS libraries, were interviewed on themes that emerged in issues of documentation. The Docathon was held by the authors of the ethnography piece and served as a space for researchers to study time-bound collaboration around documentation. Interviews emphasized the importance of documentation in addressing different kinds of learners and serving as "external memory or a living document." The study encourages documentation to facilitate newcomer onboarding and incite collaboration among developers. One interviewee shared their view on the lack of documentation for software:

> "...if there's no documentation to help, that user is basically lost for that software project and will say, 'I tried that, but it didn't work.' You need documentation...to create a minimal user experience and have it in the documentation how to set the thing up and how to do the thing that it's supposed to do."

A 2013 experimental study by the Open University of the Netherlands and Mälardalen University in Sweden was conducted on interaction with user documentation. The study involved the use of a mock software simulation with only a printed user manual for assistance in completing exercises. Results of the survey suggest that users prefer "procedural information, even to meet non-procedural information needs (Lundin, 2013)." The five participants were observed to have accessed more procedural information even though the manual contained more non-procedural information than procedural topics. In the non-procedural tasks, respondents created their own procedures from the manual. For instance, if a series of mouse clicks were not listed in an explicit sequence, the participants would write the steps out and continue using them in subsequent answers. Any practice to develop a correct "mental model" of the tool they are working with goes a long way in completing tasks. The study's findings indicate that "half the times that an information need was vocalized" prior to skimming through the included manual, it was a need that was of a "procedural nature."

A 2018 academic journal called "Advances in Intelligent Systems and Computing" featured an article on a conceptual evaluation model of API Documentation. The article touches on the basic elements of API documentation, including sample code, video tutorials, and API reference and directives (Inzunza, 2018). 95 software developers working in the industry participated in an experimental study validating the importance of these basic API documentation elements. Participants were asked to rate the elements from very important to not

important. Table 1 is a table from the article showing the importance level of basic documentation elements in percentages.

Element	Very Imp.	Important	Moderately Imp.	Slightly Imp.	Not Important
API Overview	**51.60**	13.70	33.70	1.10	0.00
Get Stared	**60.00**	7.40	29.50	1.10	2.10
Sample Code	**66.30**	11.60	21.10	1.10	0.00
Video Tutorials	25.30	20.00	13.70	**30.50**	2.10
API Reference	**44.20**	23.20	30.50	2.10	0.00
API Directives	27.40	**32.60**	26.30	10.50	0.00
Status and Error	**56.80**	10.50	27.40	4.20	1.10

Table 1: Importance level of basic documentation elements in percentages

Based on the results, the most important elements are sample code and getting started respectfully and the least important is video tutorials. By normalizing the average result for each element in the survey, the conceptual API documentation evaluation model was created in Figure 11. The total score of the weights is 3.50.

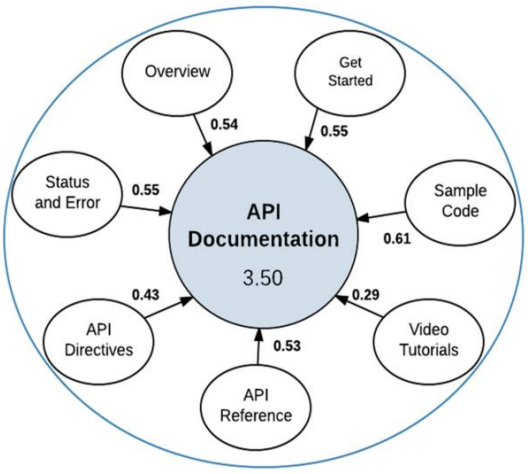


Figure 11: Importance level of basic documentation elements in percentages

The article applied this model to evaluate online API documentation. The results of these findings can be found in the second part of the literature review, "4.2.2 Examples of API Documentation."

In a 2013 study on open-source API documentation, 33 of the most popular APIs listed on ohloh.net were accessed by four researchers with expertise in development and experience using APIs and API documentation. More than half of the open-source documentation sets had both high-quality design and high-quality writing. This finding supports the idea that craftsmanship is valued by developer communities. About 82% of the API documentation studied had high-quality writing compared to the 61% that had high-quality design (Watson, 2013). The study found that aspects such as "design quality, writing quality, terminology, and navigational affordances" are critical elements of documentation. Proper formatting and terminology play a crucial role in the delivery of this educational information, as observers Ko and Riche found that documentation could exist, but "remain invisible" if the user does not know the correct vocabulary (Ko, 2011). The researchers of the study also acknowledged the notable variances among the distinct documentation sets analyzed, as the diversity in content and format presented a challenge. It was concluded that since the documentation sources were from the community it serves, it is "reasonable to assume that each specific community tailors the documentation for that community." This means that the diverse content found for each different documentation set is a good thing yet presents a challenge to developers who work with varying libraries and products. The study also investigated four past, notable studies on important elements of API documentation. Analysis of these studies suggest elements including overview documentation, short code snippets and examples, task-based documentation as helpful or

critical to learning an API. All in all, the study found that computer users desire accuracy, completeness, and correctness in the documentation they reference.

4.2.2 Examples of API Documentation

The conceptual API documentation evaluation model article from Educational Sources on User and Developer Documentation applied the results of their experimentation on existing online API documentation (Figure 11).

API	Overview	Get Stared	Sample Code	Video Tutorial	API Reference	API Directives	Status and Error
Facebook	1.00	0.92	0.75	0.58	0.92	0.58	0.42
Google	1.00	1.00	1.00	0.29	0.93	0.57	0.50
Microsoft	1.00	1.00	1.00	0.33	1.00	0.89	0.78
UM4RS	1.00	1.00	1.00	0.00	1.00	0.75	0.25
Average	1.00	0.98	0.94	0.30	0.96	0.70	0.49
Std Dev	0.00	0.40	0.13	0.24	0.40	0.15	0.22

Table 2: Importance level of basic documentation elements in percentages

In the experimental design of the study, 23 undergraduate computer engineering students at the Autonomous University of Baja California navigated API documentation from the companies in Table 2. The participants were instructed to navigate the links presented in the documentation, but not perform searches to information outside of the official documentation such as Stack Overflow. Based on the results from developers assessing the API documentation, the scores from Table 2 were created from the API documentation evaluation model in Figure 11. The best score documentation was the Microsoft Face API; it performed well in the Sample Code, API Directives, and Status and Error Code categories as the information was easy to locate (Inzunza, 2018).

A 2015 scholarly journal article from IEEE provides useful information on how API documentation can fail. It contains a study in which researchers investigated ten common API documentation problems detected from examples of documentation from experienced developers. The problems include:

- Incorrectness
- Incompleteness
- Ambiguity
- Outdatedness
- Unexplained Examples
- Inconsistency
- Fragmentation
- Bloat
- Tangled Information
- Excessive Structural Information

A mass-distributed survey measuring experiences regarding these problems garnered 69 software industry employee responses. Respondents provided 179 examples of good or bad documentation for 131 documentation units within 72 different APIs. A documentation unit (DU) refers to an "explicitly designated block of information in the documentation that's related either to an API element or to the API itself." Examples of this include method documentation and overview pages. (Uddin, 2015) 89 of the examples collected were of bad documentation, an area that the researchers focused on, as there is more to learn from studying documentation failures. In Table 3, the problems reported by software experts were sorted into main topics such

as incompleteness, inconsistency, and incorrectness. It indicates the number of specific examples of poor documentation mentioned and the number of developers who reported such an issue.

TABLE 2

API documentation problems reported in the exploratory survey.

Category	Problem	Description	E*	D*
Content	Incompleteness	The description of an API element or topic wasn't where it was expected to be.	20	20
	Ambiguity	The description of an API element was mostly complete but unclear.	16	15
	Unexplained examples	A code example was insufficiently explained.	10	8
	Obsoleteness	The documentation on a topic referred to a previous version of the API.	6	6
	Inconsistency	The documentation of elements meant to be combined didn't agree.	5	4
	Incorrectness	Some information was incorrect.	4	4
		Total	**61**	**57**
Presentation	Bloat	The description of an API element or topic was verbose or excessively extensive.	12	11
	Fragmentation	The information related to an element or topic was fragmented or scattered over too many pages or sections.	5	5
	Excess structural information	The description of an element contained redundant information about the element's syntax or structure, which could be easily obtained through modern IDEs.	4	3
	Tangled information	The description of an API element or topic was tangled with information the respondent didn't need.	4	3
		Total	**25**	**22**

** E is the number of examples that mentioned a problem; D is the number of developers who reported a problem.*

Table 3: API documentation problems reported in *How API Documentation Fails*

The survey conducted in the study collected qualitative data on documentation problems. Respondents were asked to provide comments on their experiences with documentation issues. For each problem, they rated the severity of the documentation problem from "Not a Problem" to "Blocker." The response results included fruitful information from experienced developers, mostly qualitative data on problems with auto-generated documentation, fragmented, and/or superfluous documentation. After collecting these responses, the researchers conducted a validation survey that reached 254 software developers and architects at IBM Canada and Great

Britain. The study investigated three elements of responses: problem frequency, their severity, and the necessity to solve them. These elements are plotted in Figure 12.

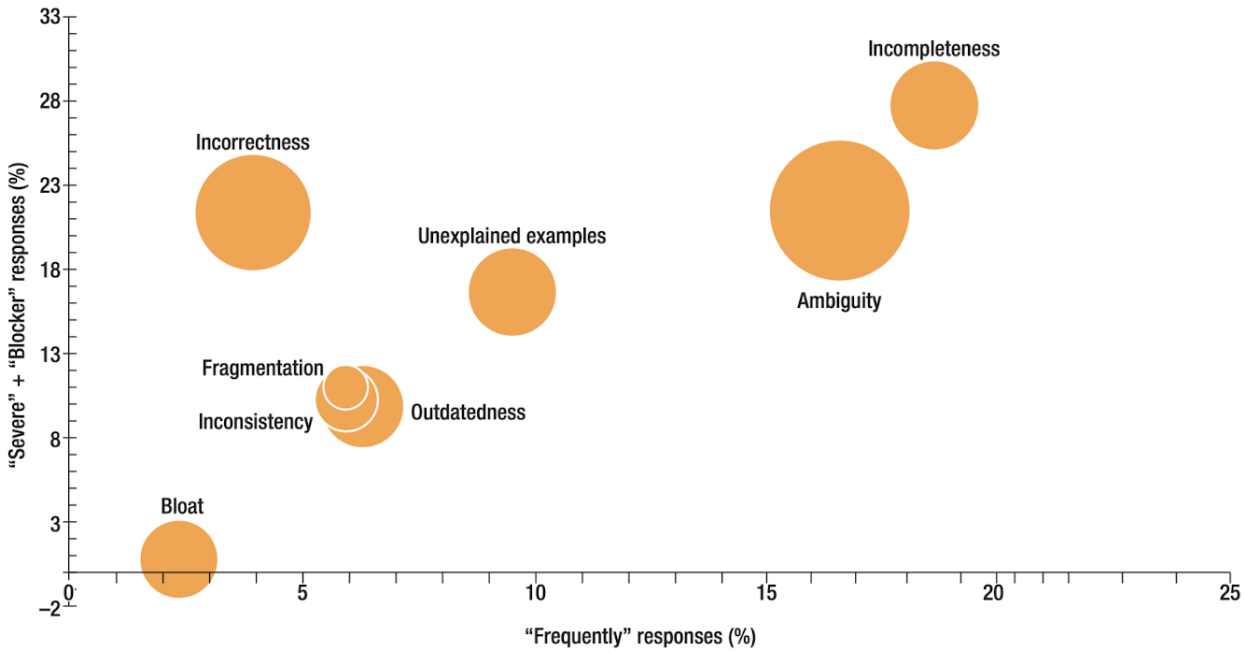


Figure 12: Analysis results of eight problems' frequency, severity, and the necessity to solve them. A circle's size represents the percentage of respondents who gave that problem top priority.

The primary finding of the study is that quality content matters the most. Researchers noted that "respondents prioritized addressing five content-related problems over any presentation problem." The largest problems with API documentation such as ambiguity and incorrectness shown in Figure 12 call for the most technical expertise. Therefore, the study communicates the need for recommendation systems that allow experts to focus on the "value-producing" part of the explained task.

One of the examples of API reference documentation is a prototype of API documentation for supporting a Lottery API. Different functions of the API are clearly outlined with provided syntax, parameters, return values, and code examples. The reference documentation is the result of an IEEE scholarly article on best practices for API documentation.

Figure 13 shows the platform created which provides tutorials for programs to use functionality of an API where applicable. It provides an Overview, Getting Started, and API Reference complete with navigation and plentiful code examples.

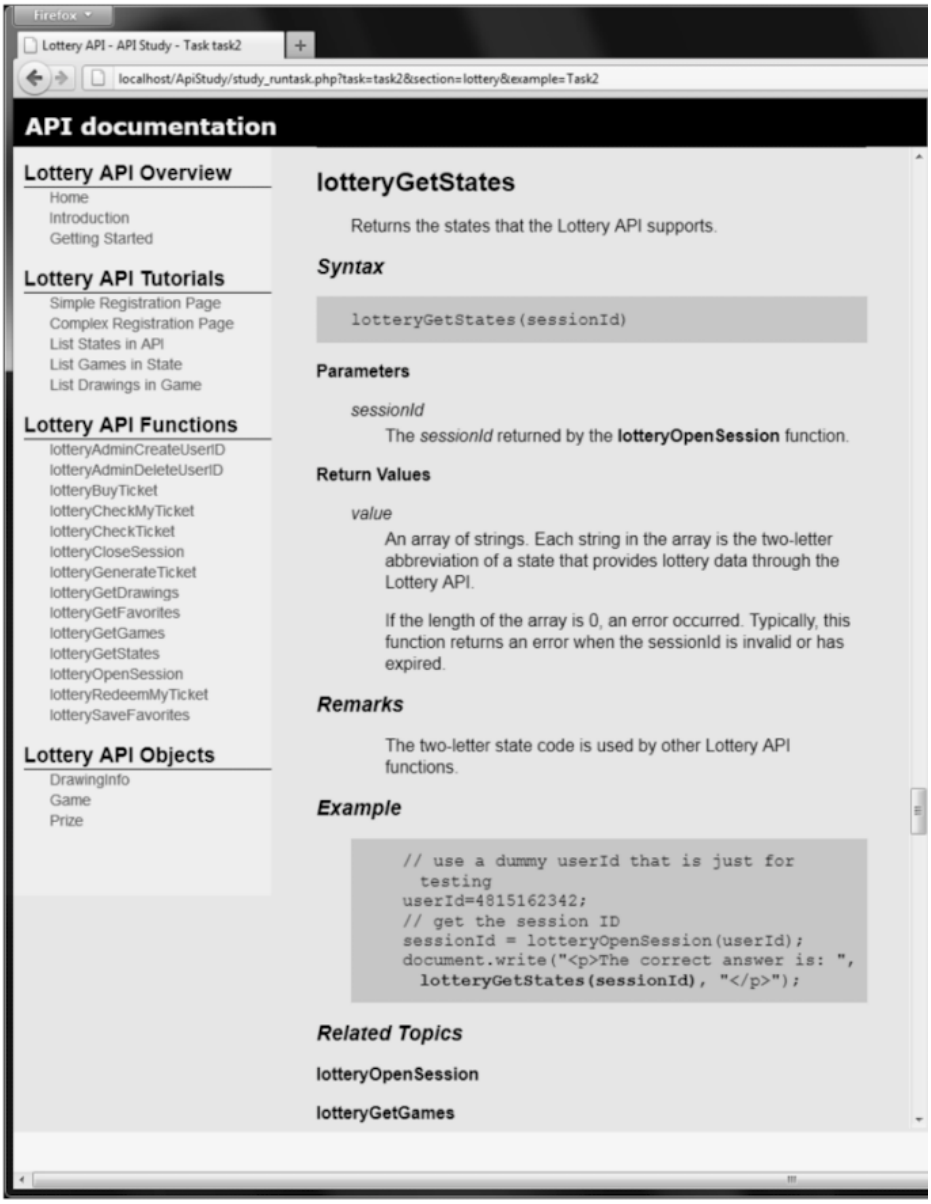


Figure 13: Screenshot of the Lottery API online and interactive API documentation

The platform specifically functions to study how programmers learn new APIs. In the design phase of this API documentation, researchers explored key elements of online document

design. One of the most important aspects of online document design mentioned is “design for the reader who skims and scans.” Lengthy excerpts of explanations can be broken up into different topics with concise language for easy scanning. The study also highlights that “the task design in a study of an API’s usability should consider the learning style of the target audience and the API.” The learning style of an API should match the learning style of the user for optimal results. In Figure 13, a JavaScript window for running and testing code was included for users. In the results, the researchers found that the coding window did not help with troubleshooting in the ways programmers expect from integrated development environments (IDEs), but rather “provided very little, if any” support. Developers spent most of their time with non-task related problems like typographical errors. Therefore, the window did not significantly help with task completion, yet elements like an overview and example code indicated clearly and concisely to users the functionality of the API.

4.2.3 Sources on Standard Operating Procedures

To create a basis of best practices in standard operating procedures (SOPs) related to software, I collected sources that exemplify standard operating procedures that help software developers or communicate technical concepts and technologies to users.

A document on developing effective SOPs from Cornell University outlines specific elements of SOPs that promote efficiency and quality control. The text emphasizes that SOPs with complete instructions ensure trainers that nothing is missed and provide a strong reference for trainees. It outlines the ways that SOPs have a positive impact in team settings. One of these ways is that having documentation such as SOPs promotes a culture of collaboration among teammates and “can encourage regular evaluation of work activity and continuous improvement in how things are done (Grusenmeyer, 2003).” To organize an SOP, the guide suggests

identifying areas in an operation that call for the use of an SOP and focusing on top priority areas. A SOP's purpose should be to primarily give instructions so that the individual can do a task correctly or have a "bare-bones" overview that can guide them in completing a task.

An article by experts in IT from BMC blogs communicates favorable features in SOPs and the challenges that come with writing them for different audiences. The article states that conciseness and usability is key to a successful SOP. It stresses that one of the greatest hardships in developing an SOP is "ensuring that it has enough detail to be usable by your audience, but not so much detail that your audience doesn't want to use it (Watts, *How to Write an SOP*)." The article has a list of 7 tips for the most effective SOPs. The list in summary is as follows:

1. Focus on the process—not the tools
    a. SOPs are different from work instructions (WI).
    b. Try to be external tool- or software- independent.
2. Be concise
    a. SOPs should be short and readable.
    b. Keep the SOP targeted; other tied yet significantly large concepts can be turned into SOPs that can be referenced.
3. Write for your audience
    a. Consider prior knowledge of the audience to avoid unnecessary explanations.
    b. Avoid jargon that can confuse or mislead the reader.
4. Clearly define steps and roles
    a. Steps should give clear directions to the reader.
    b. Each step should be mapped to a responsible party if applicable.
5. Seek input from relevant team members and stakeholders

    a. Talk to applicable teams before, during, and after drafting an SOP.

6. Test your SOP
    a. Test the SOP to ensure accuracy and usability.
    b. Outline any problem areas to be corrected in future versions.
7. Review regularly
    a. Update and review the SOP on a regular basis
    b. Look for steps that are obsolete or can be simplified

These helpful points made in the article show that SOPs should have a balance of detail around what team members should accomplish. Mapping out the process with an end goal in mind, revising it to meet user needs, and testing it sufficiently will confirm that an SOP's purpose is fulfilled.

A study on SOP design for requirement engineering, the process of defining system requirements and constraints, conveys an approach to designing an SOP for engineering in software development. To determine the need for an SOP in requirement engineering, the study suggests using Soft System Methodology (SSM). SSM is described to help in identifying a problem, determining an SOP need, and designing an SOP. It starts with identifying a real-world problematic situation. The situation is then expressed as a rich picture that describes current conditions around the problem to make it easier to understand. Next, a selection of a relevant system where the needs of the SOP are defined and communicated to solve the problem. The study investigated existing literature, best practice, previous studies, and interviews to adopt this methodology and create a conceptual model. The SSM approach can be applied to other SOPs in other fields including but not limited to software development (Albareta, 2019).

4.3 Rhetorical Inquiry

For inspiration on crafting a useful documentation set, I navigated several pieces of API documentation based on survey results and referenced documentation from the literature review. These sources have been regarded for their effectiveness and are based on popular APIs of reputable providers such as Apple, Discord, and Microsoft. I conducted an inquiry that shows how documentation meets or fails to meet expressed developer needs. The results indicate whether claims that a piece of documentation has favorable elements is supported by the articles from my literature review. They also show how example documentation mentioned in the literature review could be lacking in content deemed effective by the respondents of my survey.

4.3.1 Surveys and Literature Review

The main example of positive API documentation that I analyzed is Discord's API documentation. The survey recommended this documentation platform with an emphasis that its organization is clear and easy to follow. The documentation is housed in GitHub's Developer Portal. In Figure 14, the documentation features a navigation bar on the left for quick scanning for topics of interest. There is a Getting Started Section with hyperlinks for navigating the documentation with an Overview section and task sections for getting started using the Discord API.

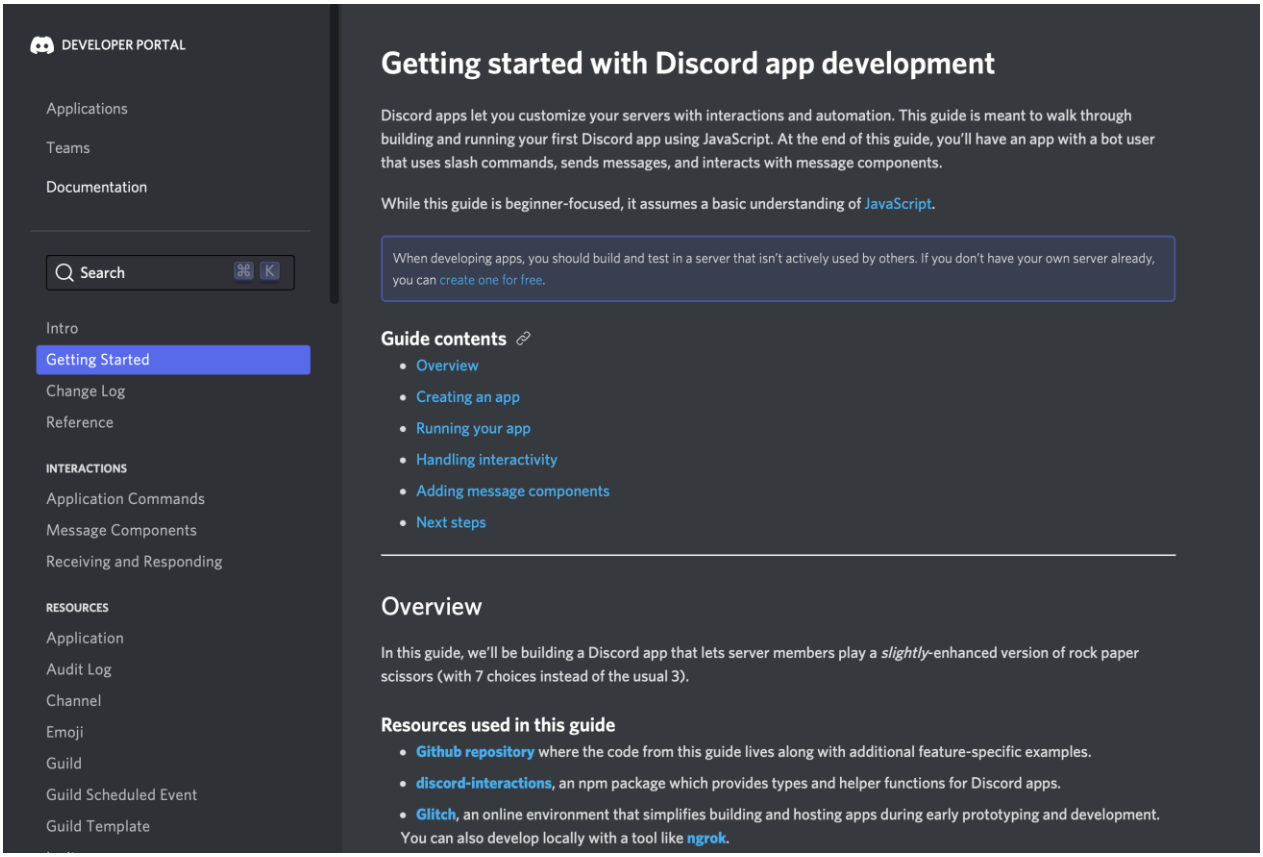


Figure 14: Screenshot of the Discord API Developer Portal with Getting Started Section

Figure 14 shows the Getting Started page which discloses that the user guide is for beginners, yet "assumes a basic understanding of JavaScript." This concise piece of information can save a lot of time and confusion for the user. As aforementioned in the example API documentation section of the literature review, this task design considers "the learning style of the target audience and the API (4.2.2)."

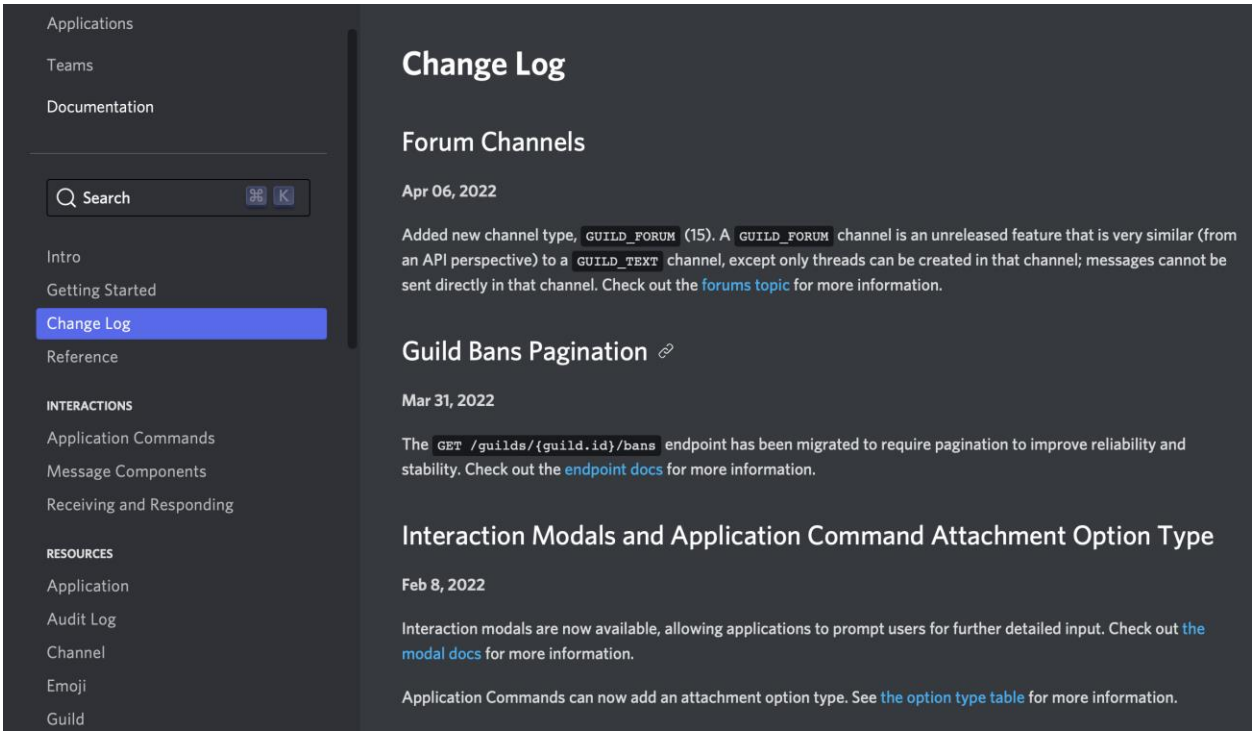


Figure 15: Screenshot of the Discord API Changelog

In Figure 15, Discord API documentation addresses the problem of deprecated or outdated information with a change log, a useful feature for providing transparency on important API updates to users. The result of the presence of this feature is that it represents best practice in documentation. There is a dated list starting with the most recent release to inform the developer community of updates, alleviating outdatedness, one of the ten common documentation problems mentioned in articles and in the survey data. There are several hyperlinks on the page that direct to supporting information that can aid developers in learning how to use the API.

The navigation bar in Discord's API documentation features informative headers and hyperlinks. When you click on a topic, the navigation expands with more subtopics. This saves time for the user and makes it much easier to locate specific actions. The documentation is complete with relevant code samples and parameter tables for easy readability. The user is not bombarded with panels of text, and in instances where redundancy can be avoided and time can

be saved, Discord's documentation points to previous explanations. This technique, seen in Figure 16, avoids tangled and excessive structural information.

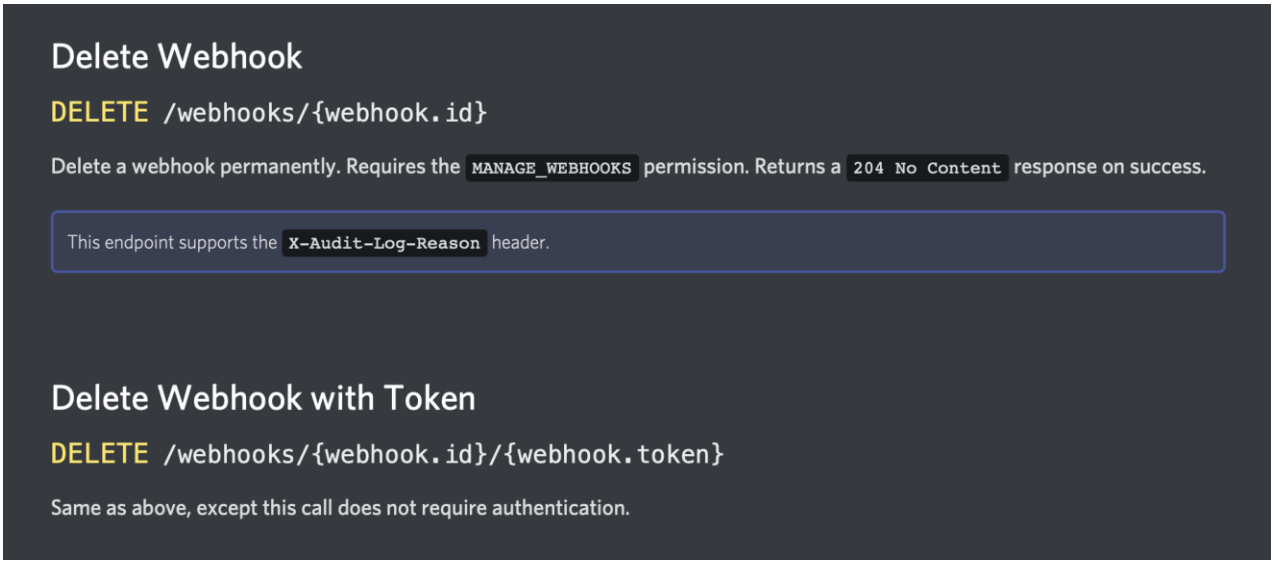


Figure 16: Screenshot of the Create Webhook method

Figure 17 shows documentation for creating a webhook. On the navigation bar, developers can view the different functionalities associated with webhooks. Again, one can visualize helpful links within the text for convenience. The text itself is concise and straightforward. Much of the documentation follows this standard.

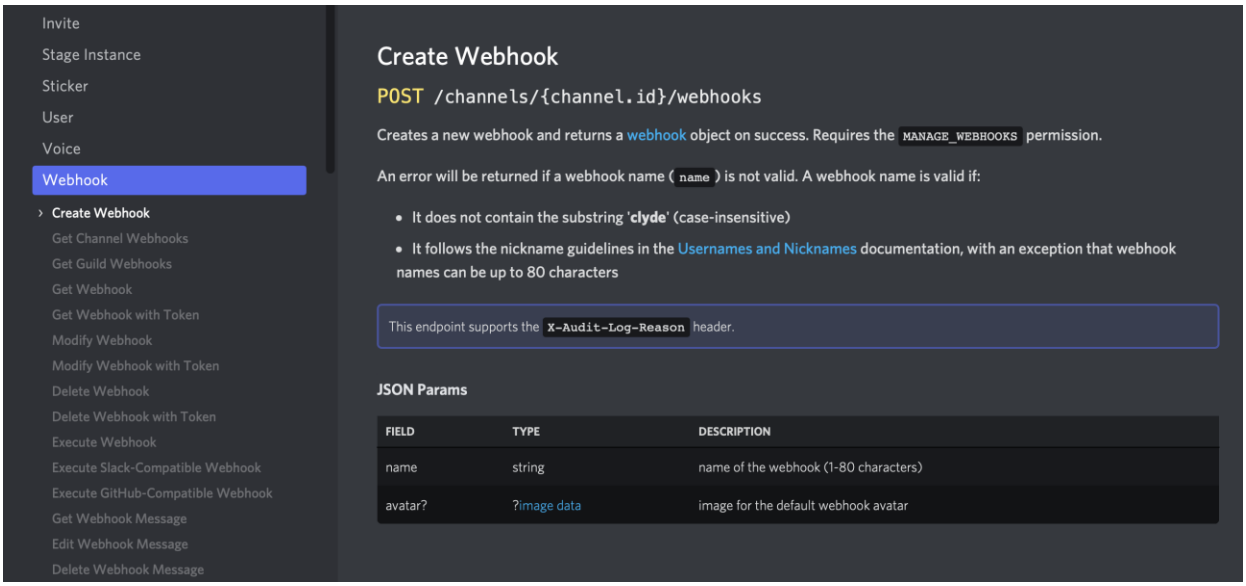


Figure 17: Screenshot of the Create Webhook method

If a resource is deprecated, it is not removed from the documentation entirely, but it is indicated to the developer (Figure 18). This is important for developers who might have heard about a deprecated feature that has been used in developer communities with an expectation that the feature still exists. Any important notes are highlighted like Figure 18.

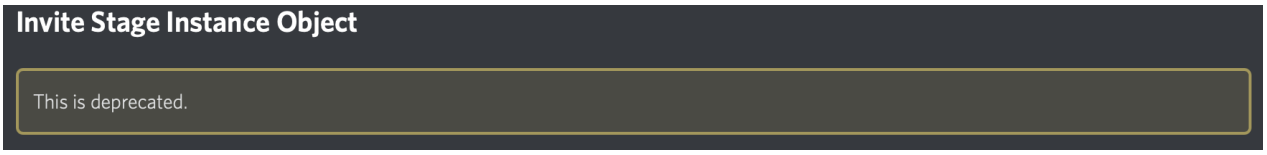


Figure 18: Screenshot of a deprecated feature

The Discord Developer team also has a noticeable presence to be available for help with Discord API. When a developer creates their first Discord app in the Getting Started tutorial, they are encouraged to read up on the documentation for other API features they can integrate. In addition, there is a link to join a Discord server of over 120,000 members for questions about the API and interactions with Discord developers. This option aligns with survey respondents' stance that features of developer forums can be applied to API documentation. While respondents acknowledge that official documentation is a key source to consult, they still have admitted to using outside forums. Discord having their own developer forum on their own platform is impressive and sets a credible example that is appealing to the developer community, especially beginners who are new to the API.

Discord's API documentation can be classified as a good model example based on survey respondents' desire for clarity with useful examples and up-to-date features. The documentation also aligns with best practices conveyed in the literature review. Other documentation examined that aligns with these principles include:

- Robot Operating System (ROS)
- Microsoft FileDialog

- Microsoft Face API

Conclusions

This section includes a synthesis of findings from the surveys, literature review, and rhetorical inquiry documented in the *Results* section. It is divided into topics that are relevant to positive qualities of API documentation determined by primary and secondary research.

5.1 Positive Elements

The surveys, literature review, and rhetorical inquiry culminated in suggestions that formed important inferences on positive API documentation elements. Based on the qualitative survey, users acknowledged that, unlike forums where finding specific information is reliant on search keywords, API documentation can have useful navigation. One user mentioned that API documentation is more “niche”, everything one should need to know about the API when done correctly, is presented to developers. Developers particularly voiced that API documentation is effective when it contains well explained code examples and references that are up-to-date and well-organized.

5.2 Negative Elements

Cryptic content is an element that was widely reported by developers in the distributed surveys. It can be concluded that a big reason for reports of material that is hard to understand is due to a lack of prerequisite knowledge. The abundance of respondents in the qualitative and quantitative survey who indicated they have had experiences with lack of prerequisite knowledge in API documentation suggests that API documentation neglects non-experts or assumes competency. Developer respondents of the qualitative survey expressed their concerns with having to learn syntax of code not defined in the documentation as expected knowledge. It can

be deduced that the lack of such information negatively impacts users and costs developers more time.

5.3 Influence of Forums

Survey respondents expressed preferences for forums like Stack Overflow. They generally like forum services because they feel guaranteed to find an answer to questions that official documentation does not address. Respondents tended to rely on forum services since they are used by other developers. Each post on Stack Overflow also has a date on it and a visual voting indicator where developers can vote on whether answers are correct. These statements expressed by respondents are evidence that API documentation typically lacks dates with its content and visuals that communicate the information is correct.

Nearly 69 percent of qualitative survey respondents mention that they think that there are elements of forums that can be adopted or altered to fit API documentation. One respondent shared that “comment and voting sections to amend wrong sections of the docs” should be integrated like forums.

5.4 Ranked Programming Sources

In the quantitative survey, based on responses on sources consulted for programming, participants chose Stack Overflow and Random Googling as the top two source methods. Stack Overflow and Random Googling rely on search engines for finding answers, a convenient way to query hubs that can aid in problem solving and troubleshooting. Participants have mentioned these sources as easy to navigate; in a single search, one can be directed to a multitude of sources that can help. It can be suggested from these results that effective API documentation elements should be easy to navigate to with an intuitive hyperlink, navigation bar, and/or table of contents.

Stack Overflow was consistently ranked as a number one source, with nearly double the responses than official documentation.

5.4 API Documentation vs Forums

The results on whether respondents would use API documentation over forums and other help differs significantly between the qualitative survey and the quantitative survey. The qualitative survey indicates an overall preference for official API documentation with about eight percent more respondents favoring official API documentation. On the other hand, about ten percent more participants in the quantitative survey chose forums and other help over official documentation. With this data, and the nature of it, it is unclear which platform is most effective to users.

Moreover, the results are based largely on personal preference. Based on the tight margin between the two choices, it appears that the decision-making of choosing might fall on one or two differing elements. In the quantitative survey, distributed strategically after the qualitative survey, participants were asked why they would choose one option over another. The top reason for choosing forums stated that forums have more examples and tutorials, while the top reason for vendor documentation is more credibility. In conclusion, documentation being a credible source and directly from the API provider, should consider more examples and tutorials for making sense of code.

Recommendations

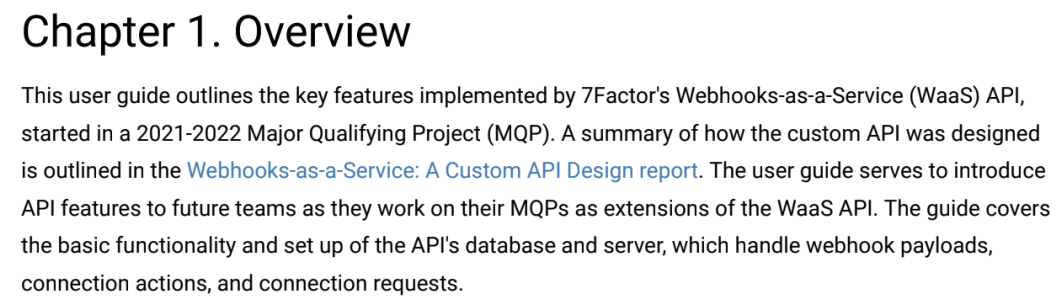
Researching general best practices in technical documentation and referencing example user guides and SOPs helped me to create a user guide and SOP for the API my team has created. I wrote these documents in DITA files generated in Oxygen XML Editor which can be

converted to HTML or PDF form for use. These documents capture what qualities developers look for in APIs such as concise language and organized topics. The recommendations followed by the documentation set I created communicate and promote the need for consistent, outlined technical documentation. In each section, there are samples of the documentation set to support each recommendation; these are derived separately attached supplemental materials that can be fully accessed.

6.1 User Guide

The *7Factor User Guide*, located in the supplemental documents, is an example reference that touches on some of the recommendations listed below.

- **Insert code examples where they make sense.** Show and explain methods by depicting instances in which they would be called.
- **Write concisely.** Showcase brief and thorough content.
- **Leverage supportive media.** Insert pictures and videos where applicable.
- **Revisit and revise documentation regularly.** Schedule meetings with developers and technical writers to edit and update API documentation alongside the API.
- **Include navigation.** Integrate smooth navigation options like a navigation bar or aa table of contents.
- **Add a change log.** Record and post history to keep developers up to date with changes.
- **Include an Overview section.** Specify the abilities of the API.

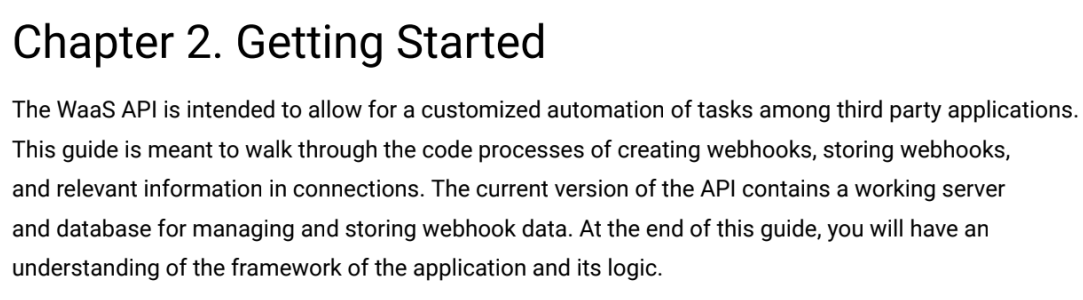


Chapter 1. Overview

This user guide outlines the key features implemented by 7Factor's Webhooks-as-a-Service (WaaS) API, started in a 2021-2022 Major Qualifying Project (MQP). A summary of how the custom API was designed is outlined in the Webhooks-as-a-Service: A Custom API Design report. The user guide serves to introduce API features to future teams as they work on their MQPs as extensions of the WaaS API. The guide covers the basic functionality and set up of the API's database and server, which handle webhook payloads, connection actions, and connection requests.

Figure 19: Screenshot of the 7Factor User Guide Overview

- **Include a Getting Started section.** Outline developer expectations with prerequisite information and/or tutorials.



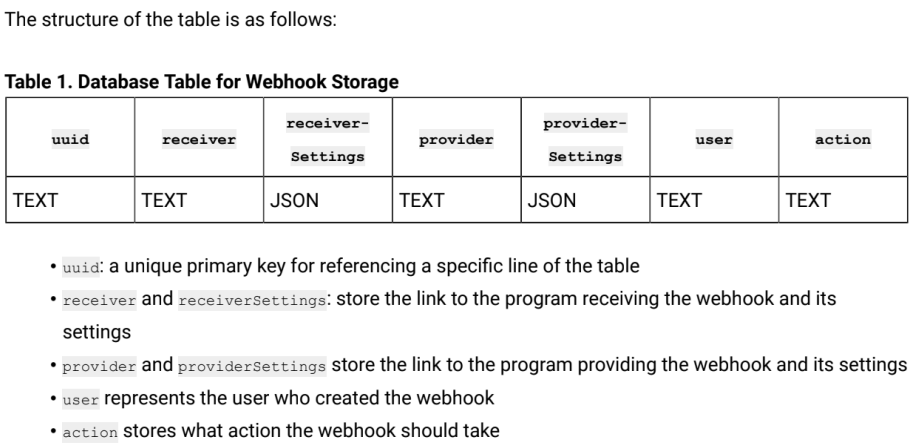
Chapter 2. Getting Started

The WaaS API is intended to allow for a customized automation of tasks among third party applications. This guide is meant to walk through the code processes of creating webhooks, storing webhooks, and relevant information in connections. The current version of the API contains a working server and database for managing and storing webhook data. At the end of this guide, you will have an understanding of the framework of the application and its logic.

Figure 20: Screenshot of the 7Factor User Guide Getting Started

- **Introduce elements of forums such as a help center.** Create a developer community around the API that supports fellow developers.
- **Format code snippets differently from the rest of the text.** Have a design standard for code blocks to make them stand out.
- **Avoid resorting to auto-generated documentation.** Provide context for the solutions an API service delivers.

- **Use tables.** Format parameters into tabular lists



The structure of the table is as follows:

Table 1. Database Table for Webhook Storage

`uuid`	`receiver`	`receiver-Settings`	`provider`	`provider-Settings`	`user`	`action`
TEXT	TEXT	JSON	TEXT	JSON	TEXT	TEXT

- `uuid`: a unique primary key for referencing a specific line of the table
- `receiver` and `receiverSettings`: store the link to the program receiving the webhook and its settings
- `provider` and `providerSettings` store the link to the program providing the webhook and its settings
- `user` represents the user who created the webhook
- `action` stores what action the webhook should take

Figure 21: Screenshot of a table included in the 7Factor User Guide

- **Fragment sections logically.** Separate features into various topics to avoid information overload.
- **Practice consistency.** Stick to uniform design and content.
- **Incentivize contributions to API documentation.** Leverage writing documentation as a task for junior/newcomer developers to work on to better understand how an API works.

6.2 Standard Operating Procedure

- **Know the audience.** Write considering those who complete the outlined tasks.
- **Be concise.** Keep your content short and comprehensive, splitting up content if need be.
- **Divide tasks properly.** List tasks in manageable chunks for easy readability.
- **Provide supportive media.** Use pictures to aid users in tasks.

Figure 22: Screenshot of an image included in the SOP

- **Review and edit regularly.** Make appropriate changes promptly and often.
- **Consult experts for feedback.** Collect input from industry professionals.
- **Include an Overview section.** Specify the purpose of the SOP.
- **Include a Getting Started section.** Outline tasks that users need to complete before getting to the main concepts.

References

Arya, D. M., Guo, J. L. C., & Robillard, M. P. (2020). Information correspondence between types of documentation for APIs. *Empirical Software Engineering*, *25*(5), 4069–4096. https://doi.org/10.1007/s10664-020-09857-0

Badras, C., Lohse, K., & Nüssel, C. (2011). *Linguistic and cultural adaptation of user documentation: Approaches and challenges*. 16th World Congress of Applied Linguistics (AILA 2011), Beijing, China, 23-29 August 2011. https://digitalcollection.zhaw.ch/handle/11475/15868

Bush, T. (2019, May 16). *5 Examples of Excellent API Documentation (and Why We Think So) | Nordic APIs |*. Nordic APIs. https://nordicapis.com/5-examples-of-excellent-api-documentation/

Charney, D., & Porter, J. E. (1991). Current Research in Technical Communication. *Technical Communication*, *38*(1), 137–139.

Discord. (n.d.). *Documentation - Introduction*. Discord Developer Portal. Retrieved April 28, 2022, from https://discord.com/developers/docs/intro

Foote, T. (2020). *Wiki: Documentation*. ROS (Robot Operating System). Retrieved April 28, 2022, from http://wiki.ros.org/

Geiger, R. S., Varoquaux, N., Mazel-Cabasse, C., & Holdgraf, C. (2018). The Types, Roles, and Practices of Documentation in Data Analytics Open Source Software Libraries. *Computer Supported Cooperative Work (CSCW)*, *27*(3), 767–802. https://doi.org/10.1007/s10606-018-9333-1

Grusenmeyer, D. (2003). *Developing Effective Standard Operating Procedures*. https://ecommons.cornell.edu/handle/1813/36910

Inzunza, S., Juárez-Ramírez, R., & Jiménez, S. (2018). API Documentation: A Conceptual Evaluation Model. In Á. Rocha, H. Adeli, L. P. Reis, & S. Costanzo (Eds.), *Trends and Advances in Information Systems and Technologies* (pp. 229–239). Springer International Publishing.

Jacobson, D., Brail, G., & Woods, D. (2012). *APIs: A Strategy Guide*. O'Reilly Media, Inc.

Ko, A. J., & Riche, Y. (2011). The role of conceptual knowledge in API usability. *2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. https://doi.org/10.1109/vlhcc.2011.6070395

Lucidchart Content Team. (n.d.). *How to Write a Standard Operating Procedure | Lucidchart Blog*. Retrieved April 28, 2022, from https://web.archive.org/web/20200908074819/https://www.lucidchart.com/blog/how-to-write-a-standard-operating-procedure

MaintainX. (n.d.). *Two Types of Standard Operating Procedures: Technical and Management*. Retrieved April 28, 2022, from https://www.getmaintainx.com/blog/two-types-of-standard-operating-procedures-technical-and-management/

McLellan, S. G., Roesler, A. W., Tempest, J. T., & Spinuzzi, C. I. (1998). Building more usable APIs. *IEEE Software*, *15*(3), 78–86. https://doi.org/10.1109/52.676963

Mehrotra, N. (2021). *Face rest API reference - azure cognitive servicesNitin*. Microsoft. Retrieved April 28, 2022, from https://docs.microsoft.com/en-us/rest/api/face/

Microsoft. (n.d.). *Filedialog class*. Microsoft. Retrieved April 28, 2022, from https://docs.microsoft.com/en-us/dotnet/api/microsoft.win32.filedialog?view=windowsdesktop-6.0

Myers, B. A., Jeong, S. Y., Xie, Y., Beaton, J., Stylos, J., Ehret, R., Karstens, J., Efeoglu, A., & Busse, D. K. (2010). Studying the Documentation of an API for Enterprise Service-Oriented Architecture. *Journal of Organizational and End User Computing (JOEUC)*, *22*(1), 23–51. https://doi.org/10.4018/joeuc.2010101903

Novick, D. G., & Ward, K. (2006). What users say they want in documentation. *Proceedings of the 24th Annual Conference on Design of Communication - SIGDOC '06*, 84. https://doi.org/10.1145/1166324.1166346

Rocha, Á., Adeli, H., Reis, L. P., & Costanzo, S. (Eds.). (2018). *Trends and Advances in Information Systems and Technologies* (Vol. 746). Springer International Publishing. https://doi.org/10.1007/978-3-319-77712-2

Roza Albareta, A., & Mursanto, P. (2019). Design of Standard Operating Procedure for Requirement Engineering in Software Development: Case Study Data Processing Integration Subdirectorate Statistics Indonesia. *Journal of Physics: Conference Series*, *1175*, 012081. https://doi.org/10.1088/1742-6596/1175/1/012081

Sohan, S. M., Maurer, F., Anslow, C., & Robillard, M. P. (2017). A study of the effectiveness of usage examples in REST API documentation. *2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 53–61. https://doi.org/10.1109/VLHCC.2017.8103450

Souza, L. B. L., Campos, E. C., Madeiral, F., Paixão, K., Rocha, A. M., & Maia, M. de A. (2019). Bootstrapping cookbooks for APIs from crowd knowledge on Stack Overflow. *Information and Software Technology*, *111*, 37–49. https://doi.org/10.1016/j.infsof.2019.03.009

Top 10 Standard Operating Procedure Software and How to Choose the Best for Your Company. (2019, November 19). *SweetProcess*. https://www.sweetprocess.com/standard-operating-procedure-software/

Uddin, G., Khomh, F., & Roy, C. K. (2021). Automatic API Usage Scenario Documentation from Technical Q&A Sites. *ACM Transactions on Software Engineering and Methodology*, *30*(3), 1–45. https://doi.org/10.1145/3439769

Uddin, G., & Robillard, M. P. (2015). How API Documentation Fails. *IEEE Software*, *32*(4), 68.

van Loggem, B. (2013). User Documentation: The Cinderella of Information Systems. In Á. Rocha, A. M. Correia, T. Wilson, & K. A. Stroetmann (Eds.), *Advances in Information Systems and Technologies* (pp. 167–177). Springer. https://doi.org/10.1007/978-3-642-36981-0_16

van Loggem, B., & Lundin, J. (n.d.). *Interaction with user documentation: A preliminary study*. 6.

Voskoglou, C. (n.d.). *APIs Have Taken Over Software Development | Nordic APIs |*. Retrieved April 28, 2022, from https://nordicapis.com/apis-have-taken-over-software-development/

Watson, R. (2012). Developing best practices for API reference documentation: Creating a platform to study how programmers learn new APIs. *2012 IEEE International Professional Communication Conference*, 1–9. https://doi.org/10.1109/IPCC.2012.6408606

Watson, R., Mark Stamnes, M., Jeannot-Schroeder, J., & Spyridakis, J. H. (2013). API Documentation and Software Community Values: A Survey of Open-Source API Documentation . *Proceedings of the 31st ACM International Conference on Design of Communication - SIGDOC '13*. https://doi.org/10.1145/2507065.2507076

Watson, R. (2014). Applying the Cognitive Dimensions of API Usability to Improve API Documentation Planning. *Proceedings of the 32nd ACM International Conference on The Design of Communication CD-ROM*, 1–2. https://doi.org/10.1145/2666216.2666239

Watts, S., & Liao, W. (n.d.). *How to Write an SOP (Standard Operating Procedure)*. BMC Blogs. Retrieved April 28, 2022, from https://www.bmc.com/blogs/sop-standard-operating-procedure/

Whitaker, J. C., & Mancini, R. K. (2018). *Technical Documentation and Process*. CRC Press. https://doi.org/10.1201/9781315217147

Y. Wu, L. W. Mar, & H. C. Jiau. (2010). CoDocent: Support API Usage with Code Example and API Documentation. *2010 Fifth International Conference on Software Engineering Advances*, 135–140. https://doi.org/10.1109/ICSEA.2010.28

Appendix A

Informed Consent Agreement for Participation in a Research Study

Investigator: Carley Gilmore

Contact Information: cngilmore@wpi.edu

Title of Research Study: Effectiveness of Computer Programming Resources and API Documentation

Introduction

You are being asked to participate in a research study. Before you agree, however, you must be fully informed about the purpose of the study, the procedures to be followed, and any benefits, risks, or discomfort that you may experience because of your participation. This form presents information about the study so that you may make a fully informed decision regarding your participation.

Purpose of the study: The purpose of this study to assess the effectiveness of API documentation that is currently available to the public, specifically computer programmers. The results of this study will be used to measure to what extent existing forums and API documentation are useful to WPI students. With WPI student feedback, the issue of inconsistent API documentation can be brought to light. Developer opinions on positive and negative experiences with API documentation will be organized and communicated to aid technical writers in writing more consistently and efficiently for their audience of developers.

Procedures to be followed: This survey is the single, primary means of data collection for this research study. In the survey, there are questions that have to do with programmers' experience with API documentation. Some of the questions have follow-up questions that analyze details about specific experiences with API documentation and web forums. The participants' identities will not be requested, as all responses are anonymous.

Risks to study participants: There are no foreseeable risks. If a participant does not feel comfortable participating in the survey, they can exit the survey and forgo submission at any time.

Benefits to research participants and others: The benefit of this research project is that participants will have a reinforced understanding of what qualities they have found in experiences with API documentation. Participants can reflect on qualities in documentation and forums that they prioritize and what they would look for in sources that aid in technical development. The goal of the research is to indicate that there are common mistakes in technical writing and showcase examples that follow the best practices in writing API documentation.

Record keeping and confidentiality: Records of your participation in this study will be held confidential so far as permitted by law. However, the study investigators, the sponsor or it's designee and, under certain circumstances, the Worcester Polytechnic Institute Institutional Review Board (WPI IRB) will be able to inspect and have access to confidential data that identify you by name. Any publication or presentation of the data will not identify you

2

Compensation or treatment in the event of injury: The research conducted by this project will not result in any physical harm or event of injury. You do not give up any of your legal rights by signing this statement.

For more information about this research or about the rights of research participants, or in case of research-related injury, contact:

Researcher:

Carley Gilmore, Email: cngilmore@wpi.edu

IRB Manager:

Ruth McKeogh, Tel. 508 8316699, Email: irb@wpi.edu

Human Protection Administrator:

Gabriel Johnson, Tel. 508-831-4989, Email: gjohnson@wpi.edu

Your participation in this research is voluntary. Your refusal to participate will not result in any penalty to you or any loss of benefits to which you may otherwise be entitled. You may decide to stop participating in the research at any time without penalty or loss of other benefits. The project investigators retain the right to cancel or postpone the experimental procedures at any time they see fit.

By signing below, you acknowledge that you have been informed about and consent to be a participant in the study described above. Make sure that your questions are answered to your satisfaction before signing. You are entitled to retain a copy of this consent agreement.

_____ **Date:** _____

Study Participant Signature

_____

Study Participant Name (Please print)

_____ Date: _____

Signature of Person who explained this study

Appendix B

Survey for WPI Students on Programming Resources and API Documentation

1) What sources do you use for integrating technology/programming errors/troubleshooting? Ex. Stack Overflow, GeeksforGeeks, etc.
    a) What do you like about these experiences?
2) Have you referred to any API documentation for projects? If so, please describe your experience with the documentation.
    a) If the experience was positive, what about it did you like? Complete, Well Explained Examples, Up-to-date, Consistent, Correct, etc.
    b) If the experience was negative, what about it did you dislike? Ex. Incompleteness, Ambiguity, Obsoleteness, Inconsistent, and/or Incorrect information.
3) If you have used API documentation, do you think there are elements of sources like StackOverflow and other forums that could be adopted?
4) When using API Documentation, as a developer, can you recall any instances where the prerequisite knowledge was vague or not mentioned?
5) Have you ever created your own API?
    a) If yes, what sources did you refer to in creating your own API?
    b) If no, do you feel that there are adequate resources/documentation to get started with creating an API?
6) Do you find it easy to locate sources that can help with using specific technologies?
7) Are you likely to use official API documentation to integrate or create your own APIs or do you rely on forums or other help?
    a) If you use documentation more, what are your reasons?
    b) If you rely on forums or other help, what are these sources and what are your reasons?

Appendix C

Qualitative Survey Data Report
Effectiveness of Computer Programming Resources and API Documentation

Q1 - What sources do you use for integrating technology/programming errors/troubleshooting? Ex. StackOverflow, GeeksforGeeks, etc.

#	Answer	%	Count
1	StackOverflow	38.18%	21
2	GeeksforGeeks	16.36%	9
3	Official Documentation	32.73%	18
4	Other	12.73%	7
	Total	100%	55

Q1_4_TEXT - Other

Other - Text
articles about specific problems
Sometimes I'll try and find the repository and read the source code. Developers are slightly more inclined to write good comments than good documentation.
Tutorialspoint
StackExchange Github issues

in the past, yahoo answers
literally anything I can get my hands on, including but not limited to scraping developers emails from git commits and bothering them
random googling

Q2 - What do you like about these experiences? Please list each source and what you like about it.

What do you like about these experiences? Please list each source and what you like about it.

sof: practical doc: full functionality and behaviour description, satisfying my weird need
official documentation was probably the most reliable way to find information. StackOverflow was occasionally helpful for other types of issues, however, everyone has a slightly different application they need the problem solved for, so sometimes the answers. aren't relevant to me, or it seems to be more of a scavenger hunt. Though it does help give ideas of how I could solve something and sometimes sparks inspiration. Same deal with yahoo answers.
These resources provide sample code to better understand the solution and how it works.
Stackoverflow: Green checkmark or lack thereof quickly indicates whether to keep looking or actually read the page Official documentation: Usually exhaustive and thorough GeeksForGeeks: Examples and generally easy to follow
Stackoverflow: A ridiculous amount of information. The problem I'm having has usually been answered. GeeksforGeeks: Good as a tutorial resource, which will show multiple examples of an algorithm implementation or use of a library. Official Documentation: When I'm deep into a project and trying to get a library to work in a very specific way, official docs are my best bet. GitHub: When nothing else gives the results I need, I'll read through source code.
Stackoverflow is good for common problems that have easy solutions. Official documentation usually works for everything else, except if the documentation is very bed (which it sometimes can be)
Stackoverflow has consistently good answers for edge cases and explanations as to why things work or don't. GeeksforGeeks feels much less curated and SEO'ed but shows up and has decent answers sometimes. Official documentation can be cumbersome but is known good. (probably SEO'ed) articles are decent for answering specific questions about a language or a problem
StackOverflow: lot of knowledge in one place and it is easy to digest and understand also usually when I am on StackOverflow I am looking for an example of how to do something and usually the

responses give multiple ways of doing something that helps me understand
StackOverflow is useful in that it enables developers to ask particular questions may have, or resolve specific problems. Often times I find that responses to questions will involve answers or issues not mentioned in official documentation. GeeksForGeeks is a source I tend to stay away from. I usually find it useful for reading brief overviews of concepts. But in terms of learning how to use an API, they lack official detail that would most likely be clarified in documentation. Official documentation is often times a great resource if written correctly. The best documentation must include the purpose of a specific implementation, examples of how to use it, common problems, and parameter details. Without these core pieces, the official documentation can become less relied upon, forcing you to ask questions and gain clarification from sources like Stack Overflow.
StackOverflow has a wide spread in terms of what it handles. GeeksforGeeks is very informative and explains itself very well, helping understanding. Official documentation is more niche but highly focused and effective when done well.
StackOverflow has a vast repository of other's experiences and issues that they've run into, so it provides a wealth of knowledge on any issue I could possibly run into. The official documentation however, for example, Apple's developer documentation which happens to be my favorite is also vital to understanding how different API's work as it goes into detail as to the different use cases, and other possible methods for each one
StackOverflow - it really seems like if you have a problem, someone else has already had it, and someone *else* has taken the time to provide a helpful answer. In very rare cases this isn't true, but it's usually a very helpful resource. Official documentation - It's like the textbook of a project. In theory, everything you need to know will be in one place, nicely organized. In practice this isn't always true, but it *usually* is in my experience.
Stack Overflow gives direct, concise answers-- useful for learning "tricks" and solutions to small problems. Official Documentation varies project by project, but can be useful for learning new capabilities of the software. This helps planning future projects, and deepening engineering of larger projects.
Stack Overflow - other people have the same kind of specific question I do and other people answer and provide working solutions GeeksforGeeks- provides multiple solutions for multiple languages and explains the topic at the top of the page TutorialsPoint- lots of slides for lots of basics. Provides not only multiple solutions, but multiple variations to fit more to what the programmer specifically needs to do.
SOF - Moderated platform, nice and easy and answers are easily integrate-able Official Docs - Their existence means they're useful.. Random googling - Bound to find someone with the same error eventually
SO: Filtered answers so the most probable solution is higher in the answer listings. GFG: tutorial style for basic knowledge questions Docs: The source.
Nothing beats the official documentation in terms of completeness and accuracy. However,

stackoverflow is useful for learning the basics about how to use new languages, or working with new APIs.
I liked how stackoverflow gives working examples as it shows how to solve a problem. I dislike how official documentation can frequently get abstract and cryptic
I like how stackoverflow usually has direct answers to issues and examples. I dislike how official documentation can be cryptic and doesnt show examples.
For all execpt GeeksForGeeks, there are real application with real fixes instead of theoretical issues.
Finding examples, and people who have had the same errors I have

Q3 - Have you referred to any API documentation for projects? If so, please list and describe your experience with documentation in the following prompts.

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Have you referred to any API documentation for projects? If so, please list and describe your experience with documentation in the following prompts.	1.00	2.00	1.06	0.24	0.06	17

#	Answer	%	Count
1	Yes	94.12%	16
2	No	5.88%	1
	Total	100%	17

Q4 - If the experience was positive, what about it did you like? For example: Complete, Well Explained Examples, Up-to-date, Consistent, Correct, etc.

If the experience was positive, what about it did you like? For example: Complete, Well Explained Examples, Up-to-date, Consistent, Correct, etc.
Documentation that is Complete and up to date is the most important in my opinion. After that, examples are very nice to have and general consistency is also nice.
Well Explained Examples for sure. These help so much with common use cases and uptake.
Using Stack Overflow I’m almost always guaranteed to find some sort of solution or link to a solution somewhere
Complete, Well Explained Examples, Up-to-date, Consistent, Correct
Very helpful guide to show the sheer scope of the project and its abilities
I think that good documentation that I've used before is always complete, correct, up to date, and consistent. Good explanations are nice, but without being able to trust that the docs are accurate, it becomes very difficult to use.
Consistently displays the correct signatures for how to use different functions/calls.
All of the above. I've never had issues. In fact only issues when it comes to IDEs is that things are too up-to-date, and I have an older version of the software.
The organization of the documentation was clear and easy to follow (Discord Botting API documentation).
Apple Developer’s Documentation: Complete, Well-explained examples, readable, consistent.
Video examples, little prerequisite knowledge, examples right in the docs
I mean, they’re correct and concise and make my life easier.

Easy to follow. Easy to get necessary data
Methods, events, options etc all laid out with great examples and ways for me to learn how to use them clearly defined
Complete, Well Explained Examples, Up-to-date, Correct

Q5 - If the experience was negative, what about it did you dislike? For example: Incomplete, Ambiguous, Obsolete, Inconsistent, Incorrect, etc.

If the experience was negative, what about it did you dislike? For example: Incomplete, Ambiguous, Obsolete, Inconsistent, Incorrect, etc.
Incompleteness or incorrectness are usually my biggest complaints about certain project's documentation.
Incomplete, obsolete, not always sure what version documentation is valid for, poor/no examples
Sometimes the format of the documentation is just hard to read, especially notation used for parameters, etc.
Some of the official documentation doesn’t always explain or give examples of how to use certain functions.
A lot of inconsistencies between the code and the docs
Out of date documentation is annoying, especially when a service hosts multiple pages for documentation, one of which is incorrect (or for a deprecated version).
I disliked how it was cryptic; hard to understand, etc.
I haven't had much negative experiences with documentation
The organization was very repetitive and hard to follow. (Microsoft documentation, namely when researching FileDialog windows)
Lack of examples -- providing a list of functions and their parameters is a reference, not complete documentation.
Sometimes examples are very broad or use coded language like FOO and BAR and make leaps like :everyone should know how to use our API
Incomplete, Obsolete, Incorrect

Q7 - Do you think there are elements of sources like StackOverflow and other forums that could be adopted in official API documentation? Please give examples if applicable.

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Do you think there are elements of sources like StackOverflow and other forums that could be adopted in official API documentation? Please give examples if applicable. - Selected Choice	1.00	2.00	1.31	0.46	0.21	16

#	Answer	%	Count
1	Yes	68.75%	11
2	No	31.25%	5
	Total	100%	16

Q7_1_TEXT - Yes

Yes - Text

Examples are always nice to have, and stackoverflow usually has them. However, generally stackoverflow is a first resort to see if someone else has the same problem so that you don't have to read the documentation.
On the old pygame docs, I think there used to be a way to add a comment underneath API sections. These areas often had discussion about how to use a specific feature of the library, which was useful.
Yes and No. I mean it's already there: major software projects already have good documentation AND active Stack Overflows. I think these are good to keep separate. Imagine the nightmare of a community driven documentation (heavy on Stack Overflow).
Examples and situations of not just how, but when to use certain functions
Comment and voting sections to amend wrong sections of the docs
I think official documentation should incorporate more examples.
Possibly. Having Q and As under sections of documentation.
Example code. That's it. And not snippets, like an entire method or short program that I can copy/paste and actually run.
Perhaps more examples or different ways to do things
I think a 'common errors' or known issues type thing could be helpful. Like just calling out the things people commonly dont get right with the API
Community contributions should be embraced, especially if nobody's getting paid to keep the official documentation maintained.

Q8 - When using API documentation, as a developer, can you recall any instances where prerequisite knowledge was vague or not mentioned? Please elaborate.

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	When using API documentation, as a developer, can you recall any instances where prerequisite knowledge was vague or not mentioned? Please elaborate. - Selected Choice	1.00	2.00	1.56	0.50	0.25	16

#	Answer	%	Count
1	Yes	43.75%	7
2	No	56.25%	9
	Total	100%	16

Q8_1_TEXT - Yes

Yes - Text

All of the time. I do most of my work in a framework called ROS, which is an amalgamation of official packages and many third party packages and libraries. The assumed or prerequisite knowledge is often implied, and rarely explicitly stated.
Oh all the time. Knowing promises patterns in JS, for example.
For some extensively large projects you will need to be familiar with its infrastructure to find needed class.
All the time. It seems like developers assume people know the domain that they are working in
Many times (especially with microsoft stuff, C#, etc), it just kind of says taht the method exists without providing context for non-experts to understand.
I can't remember specifics but it's definitely happened.
There are times where something will reference another library, or standard lib as if one should just know it. Or it will reference things like "use try catch with the following block" but their block doesnt have that, so you spend twenty minutes looking up syntax and realizing you need to add a null return because they use bluebird, or some silly thing like that

Q9 - Have you ever created your own API?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Have you ever created your own API?	1.00	2.00	1.47	0.50	0.25	15

#	Answer	%	Count
1	Yes	53.33%	8
2	No	46.67%	7
	Total	100%	15

Q10 - What sources did you refer to in creating your own API?

What sources did you refer to in creating your own API?
Generally I just try to maintain a consistent style and comment code well.
OpenAPI 3.0 Spec
n/a
Experiences.
Example projects and pre-existing code bases.
Swaggerhub
SOF/official docs
I just made them based on needs

Q11 - Do you feel that there are adequate resources/documentation to get started with creating an API?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Do you feel that there are	1.00	2.00	1.43	0.49	0.24	14

	adequate resources /documentation to get started with creating an API? - Selected Choice						

#	Answer	%	Count
1	Yes	57.14%	8
2	No	42.86%	6
	Total	100%	14

Q11_1_TEXT - Yes

Yes - Text

I believe there are resources and books out there.
While I said yes here, I think that it also depends on which service and what you are trying to accomplish.
I would assume there are, but I haven't looked into it. So many resources for everything. Though I've been interested in making one in the past and probably should start looking into it.
Apple provides a lot of information on how to use DocC to create documentation for your own API and how you could even go about writing your own API complete with tutorials and everything
For established libraries, yes

Q11_2_TEXT - No

No - Text

I honestly don’t know. I never looked up how to
Not sure why there's a text box here with no text prompt
Basically I just wing it all the time
I saw a freecodeacademy video but I don't have the need to create one now

Q12 - Do you find it easy to locate sources that can help with using specific technologies?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Do you find it easy to locate sources that can help with using specific technologies? - Selected Choice	1.00	2.00	1.15	0.36	0.13	13

#	Answer	%	Count
1	Yes	84.62%	11

2	No	15.38%	2
	Total	100%	13

Q12_1_TEXT - Yes

Yes - Text

It really depends on the specific technology and how commonly used it is. The more people use it, the more likely I am to find the information I need.
Search these days is pretty good. Even Youtube can be helpful
Usually a quick google search with your question and the programming language or device will get an answer
Not sure why there's a text box here with no text prompt
Google is a good place.
Yep, SOF and docs always have what I want.

Q12_2_TEXT - No

No - Text

The only way to know how to use specific technologies is to already know them and then assume everyone else does too

Q13 - Are you more likely to rely on official API documentation to integrate or create your own APIs or forums and other help?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count

1	Are you more likely to rely on official API document ation to integrate or create your own APIs or forums and other help?	1.00	2.00	1.43	0.49	0.24	7

#	Answer	%	Count
1	Official API Documentation	57.14%	4
2	Forums and Other Help	42.86%	3
	Total	100%	7

Q14 - What are your reasons for relying on official API documentation more?

What are your reasons for relying on official API documentation more?
Generally it is more complete and reading it gives the up to date answers quickly
(Why am I referring to other documentation when I am creating my own API??)
They’re generally up to date and accurate
Because it is generally the MOST correct and updated when changes or deprecations happen

Q15 - What are your reasons for relying on forums or other help? What are these sources and how do they help more than API documentation?

What are your reasons for relying on forums or other help? What are these sources and how do they help more than API documentation?
They are often more up to date than official docs. Also, the Q&A format helps answer frequently asked questions that documentation may miss.
Primarily because they give examples that are more easy for me to be able to use complete my given task/solve my issue.
Forums cater to people asking a question because they don't know and need an explanation. Docs are more often than not just an info dump that is unhelpful to someone who's trying to learn. Having a complete reference is important, but useless if I can't figure out how to work any of it.

Appendix D

Quantitative Survey on the Effectiveness of Computer Programming Resources and API Documentation (Revised)

Q1 - What sources do you use for integrating technology/programming errors/troubleshooting?

#	Answer	%	Count
1	Stack Overflow	23.56%	49
2	GeeksforGeeks	15.87%	33

3	Official Documentation	18.27%	38
4	Github source code	14.90%	31
5	Random googling	22.60%	47
6	Directly consulting developers	2.88%	6
7	Other	1.92%	4
	Total	100%	208

Q2 - Please rank these sources based on the likelihood that they will be able to effectively help you with integrating technology/troubleshooting. (The topmost ranking represents the source most likely to help.)

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Stack Overflow	1.00	5.00	1.90	1.04	1.09	52
2	GeeksforGeeks	1.00	6.00	3.90	1.64	2.70	52
3	Official Documentation	1.00	6.00	2.60	1.40	1.97	52
4	GitHub source code	2.00	7.00	4.31	1.01	1.02	52
5	Random googling	1.00	6.00	3.81	1.37	1.89	52
6	Directly consulting developers	1.00	7.00	4.69	1.90	3.60	52

7		Other		1.00		7.00		6.79		0.99		0.97		52	

#	Question	1		2		3		4		5		6		7		Total
1	StackOverflow	46.15%	24	28.85%	15	15.38%	8	7.69%	4	1.92%	1	0.00%	0	0.00%	0	52
2	GeeksforGeeks	5.77%	3	21.15%	11	17.31%	9	13.46%	7	17.31%	9	25.00%	13	0.00%	0	52
3	OfficialDocumentation	26.92%	14	28.85%	15	17.31%	9	13.46%	7	11.54%	6	1.92%	1	0.00%	0	52
4	GitHub sourcecode	0.00%	0	3.85%	2	15.38%	8	38.46%	20	32.69%	17	7.69%	4	1.92%	1	52
5	Randomgoogling	5.77%	3	11.54%	6	25.00%	13	23.08%	12	23.08%	12	11.54%	6	0.00%	0	52

6	Directly consulting developers	13.46%	7	5.77%	3	7.69%	4	3.85%	2	13.46%	7	51.92%	27	3.85%	2	52
7	Other	1.92%	1	0.00%	0	1.92%	1	0.00%	0	0.00%	0	1.92%	1	94.23%	49	52

Q3 - Have you referred to any API documentation for projects?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Have you referred to any API documentation for projects?	1.00	2.00	1.10	0.29	0.09	52

#	Answer	%	Count
1	Yes	90.38%	47
2	No	9.62%	5
	Total	100%	52

Q4 - In your positive experiences, what applicable qualities in official API documentation have you encountered and like?

#	Answer	%	Count
4	Complete	21.13%	41
5	Well Explained Examples	22.16%	43
6	Up-to-date	18.56%	36
7	Consistent	15.46%	30
8	Correct	21.13%	41
9	Other	1.55%	3
	Total	100%	194

Q4_9_TEXT - Other

Other - Text
Sandbox demos / prefilled editor that allows for a demo return call
Parameters and their types, if applicable
Easy to search

Q5 - In your negative experiences, what applicable qualities in official API documentation have you encountered and dislike?

#	Answer	%	Count
4	Incomplete	20.00%	31
5	Ambiguous	27.74%	43
6	Obsolete	21.94%	34
7	Inconsistent	15.48%	24
8	Incorrect	11.61%	18
9	Other	3.23%	5
	Total	100%	155

Q5_9_TEXT - Other

Other - Text
Out-of-date
Only including one example for complex/multi-use features, making them seem unusable any other way
sometimes they’re way over my head, and I feel overwhelmed with information
Auto-generated
Lack of Examples

Q7 - Do you think that there are elements of sources like Stack Overflow and other forums that could be adopted in official API documentation?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Do you think that there are	1.00	2.00	1.19	0.39	0.16	52

	elements of sources like Stack Overflow and other forums that could be adopted in official API document ation?						

#	Answer	%	Count
1	Yes	80.77%	42
2	No	19.23%	10
	Total	100%	52

Q8 - When using API documentation, as a developer, can you recall any instances where prerequisite knowledge was vague or not mentioned?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	When using API document ation, as a developer , can you	1.00	2.00	1.18	0.38	0.15	51

	recall any instances where prerequisite knowledge was vague or not mentioned?						

#	Answer	%	Count
1	Yes	82.35%	42
2	No	17.65%	9
	Total	100%	51

Q9 - Have you ever created your own API?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Have you ever created your own API?	1.00	2.00	1.61	0.49	0.24	51

#	Answer	%	Count
1	Yes	39.22%	20
2	No	60.78%	31
	Total	100%	51

Q10 - What sources did you reference to create your own API?

#	Answer	%	Count
4	Stack Overflow	31.58%	12
5	Example projects and pre-existing code bases	39.47%	15
9	OpenAPI 3.0 Specifications	13.16%	5
10	SwaggerHub	13.16%	5
11	Other	2.63%	1
	Total	100%	38

Q10_11_TEXT - Other

Other - Text

Language docs (such as Flask’s official site)

Q11 - Do you feel that there are adequate resources/documentation to get started with creating an API?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Do you feel that there are adequate resources /documentation to get started with creating an API?	1.00	2.00	1.33	0.47	0.22	48

#	Answer	%	Count
1	Yes	66.67%	32
2	No	33.33%	16
	Total	100%	48

Q12 - Do you find it easy to locate sources that can help with using specific technologies?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Do you find it	1.00	2.00	1.22	0.41	0.17	50

	easy to locate sources that can help with using specific technologies?						

#	Answer	%	Count
1	Yes	78.00%	39
2	No	22.00%	11
	Total	100%	50

Q13 - Are you more likely to rely on official API documentation to integrate/create your own APIs or forums and other help?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Are you more likely to rely on official API documentation to integrate/ create	1.00	2.00	1.55	0.50	0.25	49

	your own APIs or forums and other help?						

#	Answer	%	Count
1	Official API Documentation	44.90%	22
2	Forums and Other Help	55.10%	27
	Total	100%	49

Q14 - What are your reasons for relying on official API documentation more?

#	Answer	%	Count
4	More reliable	32.65%	16
6	More examples and tutorials	10.20%	5
7	More credible	36.73%	18
8	Other	6.12%	3
9	Easier to read	14.29%	7
	Total	100%	49

Unable to export widget. Please contact Qualtrics Support.

Q15 - What are your reasons for relying on forums or other help over official API documentation?

#	Answer	%	Count
5	More reliable	13.79%	8
6	More examples and tutorials	41.38%	24
7	More credible	3.45%	2
8	Easier to read	34.48%	20
9	Other	6.90%	4
	Total	100%	58

Q15_9_TEXT - Other

Other - Text
up to date
Easier to find similar examples to what you are struggling with
Mostly because it’s easier to understand, and doesn’t go over my head.
Up to date