Team process

The following article gives a general overview of the established processes in the **Documentation Team** and is designed for new Technical Writers who join the Company.

Expectations

By default, we consider all Documentation Team writers to be strong professionals who require no excessive supervision and micromanagement. This means you are free to choose your way to create quality documentation as long as it meets the following requirements:

- **Result**: external customers and internal stakeholders are happy with the articles you produce.
- **Transparency**: your work can be tracked via Jira and Git commits.
- **Appropriate timeframe**: try to correctly estimate the time you need to perform a task and keep the deadline in mind.

Another and probably the most important rule is **always ask questions**. It's OK not to know everything but not OK to keep your mouth shut. It's great if you can investigate and find the answer yourself. But if you are feeling lost or spending too much time on this, address your colleagues—they will always help you.

Tools of trade

Before getting to work, set up the following software:

1. Code editor, like Sublime Text, Visual Studio Code (VCS), or Atom.
2. Docusaurus: the static-site generator we use to build documents from the source markdown files.
3. Grammarly for proofreading your documents and finding grammatical, spelling, and punctuation errors. Make sure to use the Premium version provided by the Company.
4. Vale Server for writing your documents in compliance with Google Style Guide.
   If you use Visual Studio Code, you can also install Vale CLI and the corresponding VSC extension without the license.
5. VPN client for connecting to the WorkFusion internal resources. It will be required for accessing the staging documentation at doc-dev.workfusion.com.
6. Cygwin (for Windows users only) for launching the documentation website on a local live server. With the server running, any changes you make in the source files will be instantly reflected on the rendered website in a way that clients will finally see them.
7. Git client (optional). For publishing your updates and changes to the documentation repository, you can use any tool you like, for example, Sublime Merge or SourceTree, Source Control embedded into VCS, or work with it via any command-line tool.
8. Pandoc for converting HTML files exported from Confluence into the Markdown format.

> **Important**
>
> Even technical writers must study the documentation of the tools they use. Before addressing your colleagues for help, try searching for the answer yourself. Our team encourages autonomy and ownership.

Before getting to work, you'd want to study the following resources carefully:

- Docusaurus as it's our primary tool.
- Readme in our repository to learn some tips and tricks.
- Google Style Guide is the primary source of styling truth we use.
- Adobe Acrobat Pro (purchased business-level license) for viewing and advanced editing functionality for PDFs (such as combining single pages into one file). Request from IT help desk.

It is mandatory to scrutinize these resources during your first days in the Company.

Receiving tasks

From Master features

As a service unit, we rarely create assignments for ourselves and describe Product functionality developed by other teams. Workfusion holds a company-wide PI planning session each month to determine the amount of work for the next two sprints, the available capacity, and the possible risks. Documentation acts as a dependency during this session, meaning other teams reach us if they need our help with the documentation. For us, this results in the number of **Stories** related to other teams' Jira **Epics**, which in their turn are created under the Jira Align **Master Features**. Each such **Story** is assigned to a technical writer.

These **Stories** have the highest priority and must be addressed firsthand. They don't usually contain much helpful information in the description, so your first step will be going up to the **Epics** and **Master features** and their authors for details.

From other sources

The new assignments may also come from the following sources:

- Jira tasks that are not tied to initiatives, Epics, or Master features
- Requests in the **#documentation**[redacted] channel
- Personal requests

Of all the above, the requests submitted via Jira are the most preferable and transparent option.

When you receive requests for creating or editing documentation personally or in Slack channels, ask about their priority and assess the efforts required to resolve it. The general rule of thumb here is if the work can take you more than 30 minutes, the Jira ticket is required. In any such situation, ask the requester to create the ticket in the [redacted] project and select **Docs** in the **Scrum Team** box.

To ease your life, the tickets must contain the following:

- **Epic**. The task must not hold in a vacuum but be tied to ongoing initiatives.
- **Priority**. Set **Critical** for most urgent tickets. And don't use **Blocker** so as not to agitate the Release Managers.
- **Fix version**. If your documentation changes relate to several Product versions, create separate tickets and set a specific **Fix version** in each of them.
- **Description** including the problem, required outcome, Subject Matter Experts (SMEs) to address for more information, and the affected Product versions.

The empty tickets are not acceptable. If the above information is missing, insist on completing the ticket as required.

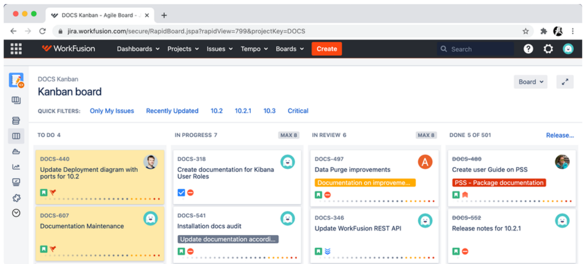
Tickets flow

The created tickets appear at the bottom of the project's Backlog. The **Blocker** and **Critical** tasks are the most important ones and must be taken first. But before that, it will be a good idea to contact a responsible SME (usually, the reporter or a person stated in the description) and ask them for details. Once you receive the confirmation that they will provide you with the information ASAP, set the ticket's status to "**To do**" so that it appears on the Team's board.

When you are ready to work on the ticket, set it to **In Progress**. We recommend working on only one assignment at a time, but understandably, the activity may be put on hold due to a change of priorities. In this case, take a new one but avoid having too many in the column.

Move the ticket to the In Review column when you finish the writing, and consider that your work only requires reviewing. Remember to provide links to the created or updated articles (or Pull Requests in the git repo) and tag the required persons with "@" to check the changes. Do not hold tickets in the column for long, and remind the SMEs about the feedback from their side from time to time.

When you finish applying changes after the feedback, and the SMEs explicitly tell you that the documentation is customer-ready, close the ticket by moving it to the **Done** column. If, after that, you receive requests for updating documentation related to the ticket, move it back to the **In Progress** or **In review** column and follow the above flow.



After you finish a task, you shouldn't wait for a manager to approach you with a new assignment. Pick up an unassigned one from the Backlog and ask the manager about its readiness and other details.

Working with Subject Matter Experts

You may want to know who's who in the Company to address the right people for the technical expertise. To avoid getting lost among all these POs, Directors, and Team Leads, refer to the Engineering Structure table. Here, in most cases, you'll need the **Team Lead** column—this person is your technical friend and will tell you about the aspects of their components or at least direct you to the right team member for further explanations.

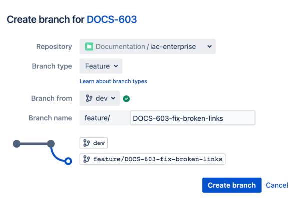
When preparing Release Notes, you can also address persons in the **PO** column, as they may consult you about what's included in the coming release.

Writing docs

When taking a task into work, we recommend creating a separate branch.

The general workflow here is the following:

1. In Jira, in the **Ticket** window, click **Create branch**, and set the following:
   - **Branch type**: usually **Feature**, but you can select other options, like **Bugfix** or **Custom** if you find it more suitable.
   - **Branch from**: always make your branch from **Dev**.



2. Click **Create branch.**
3. Go to the editor or favorite Git client, and check out the created branch.
4. Start working on the task.
5. After finishing it, and before committing and pushing your changes to the remote repository, always make **Pull from the origin/dev** branch so that your update won't affect other writers' work.
6. Commit the changes along with the message and **Push** it to the repository.
7. At https://git.workfusion.com/projects/DOC/repos/iac-enterprise/browse, go to **Pull requests** and create a request from your branch to **Dev**. Remember to specify reviewers on the next step: [redacted] and optionally SMEs.
8. Once any reviewer approves your request, the **Merge** button becomes active. Click it to merge your changes to Dev. In 5-10 minutes, you can check the document updated on the live environment at https://doc[redacted].workfusion.com/enterprise/ (VPN required).
   If the request is marked as **Needs work**, look for the comments in the files in your commit, apply the necessary changes, and notify the reviewers again personally.

For more information on working with the source files and Docusaurus (our static-site generator behind the website), see https://git.workfusion.com/projects/DOC/repos/iac-enterprise/browse/readme.md.

Reviewing

Before submitting your writing to review by pushing changes to the repo, always check your work with provided tools:

- **Grammarly**: grammar and style.
- **Vale**: compliance with the Google style guide and the Company's vocabulary.
- **Local live website**: running a live web server from the **/website** directory and with the `npm start` the command will show you how your work will appear for clients. If the execution fails locally, the same situation will occur in the prod environment. See https://git.workfusion.com/projects/DOC/repos/iac-enterprise/browse/readme.md.

Publishing

Documentation publishing for the Team means merging the latest changes from the **Dev** branch to the **Master** one. We do it in the following cases:

- When we take on the critical mass of current or previous docs changes, the articles must be updated.
  Any writer may initiate this update by creating the Pull Request from Dev to Master. After receiving approval from reviewing writers and merging the PR, the changes appear at https://doc.workfusion.com/enterprise within 15 minutes.
- When we release the new version along with the Product version.
  For more information, see **Cutting new version** at https://git.workfusion.com/projects/DOC/repos/iac-enterprise/browse/readme.md.

Time tracking

Worktime logging is required for transparency of your work and determining current load and capacity. In Workfusion, time tracking is tied to Jira tasks.

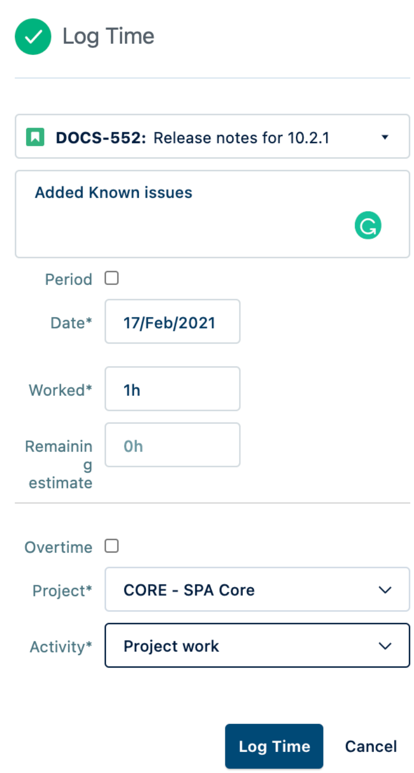
While there are no strict requirements for this operation, you must have your work hours added to the system at the end of each day so you won't miss any performed tasks.

If you have forgotten to log time till the end of the week, you can still do it the following Monday. After that, the previous period is closed, and you will have to ask Jira Admins to open the period again.

You can log time in the following ways:

- Via ticket: open the required ticket, and click **More** (...) > **Log time**.
- Via Tempo: in the upper menu, click **Tempo** > **My work**, and move tickets from the **Issues** section to the required Day column.

For the details on each **Activity** type, see Time logging in Jira Tempo in GTO.



Syncs and teamwork

The Documentation team meets for team updates and synchronization on Wednesdays. We plan our work for the coming week and share news, concerns, or risks during this session.

Before or during the meeting, estimate the tasks you plan for the week and move them from the **Backlog** to the **To do** column. Choose wisely so as not to have too much or little of them in progress.

Writers on duty

The Documentation team has recently introduced the Duty Writer practice. This initiative is created to deal with the documentation debt ad hoc requests and ensure no tasks remain in the Backlog for too long. It also helps non-duty fellow writers focus on the Major task and avoid dispersing attention into minor ones.

When on duty, the writer devotes 40% of their time (~16 hours per week) to:

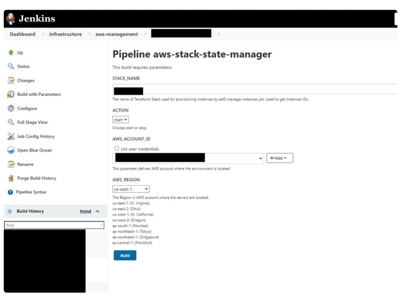
- Feedback in the **documentation_i_i** channel. Respond to requests and inform on the progress.
- Backlog tasks (non-assigned, trivial, minor).
- Tasks from the Support team.
- Fixing the broken documentation links.
- Hotjar feedback (https://insights.hotjar.com/sites/[redacted]208. Remember to remove the card after fixing the issue.

IA Cloud instance for Documentation and Enablement

For your Documentation or Enablement activities, you may use a live IA Cloud instance.

To run the instance:

1. Activate WorkFusion VPN.
2. Go to Jenkins job https://jenkins.workfusion.com/view/[redacted]/. If you don't have access to Jenkins, request it from IT Ops.
3. On the left, click **Build with parameters** (Собрать с параметрами).
4. In the **Pipeline aws-stack-state-manager** window, specify the following parameters:
   - STACK_NAME = [redacted]1169
   - ACTION = Start or Stop (to run or shut down the instance accordingly)
   - AWS_ACCOUNT_ID = [redacted]077
   - AWS_REGION = us-east-1



5. Click **Build**.
   Once the instance is started or stopped, the **Slack** bot notifies you.

After you finish working with the environment, make sure to stop it with the same Jenkins job.

After the instance starts, insert the URLs below in your browser's line to access the respective IA Cloud components. To log in, use the credentials listed below the URLs.

```
PROJECT_URL:
ct_url:            'https://wfinst11[redacted]
workspace_url:     'https://wfinst11[redacted]
bot_manager_url:   'https://wfinst11[redacted]
sqc_url:           'https://wfinst11[redacted]
kibana_url:        'https://wfinst11[redacted]
s3_endpoint_url:   'https://wfinst11[redacted]
nexus_url:         'https://wfinst11[redacted]
rabbitmq_api_url:  'https://wfinst11[redacted]
vault_url:         'https://wfinst11[redacted]
ocr_url:           'https://wfinst11[redacted]
keycloak_url:      'https://wfinst11[redacted]
mesos_url:         'https://wfinst11[redacted]
marathon_url:      'https://wfinst11[redacted]
ags_url:           'https://wfinst11[redacted]
ams_url:           'https://wfinst11[redacted]
amms_url:          'https://wfinst11[redacted]
wms_url:           'https://wfinst11[redacted]
tds_url:           'https://wfinst11[redacted]
haproxy_stats_url: 'https://wfinst11[redacted]
zoonavigator_url:  'https://wfinst11[redacted]
elasticsearch_url: 'https://wfinst11[redacted]
CREDS:
CT:                [redacted]
```