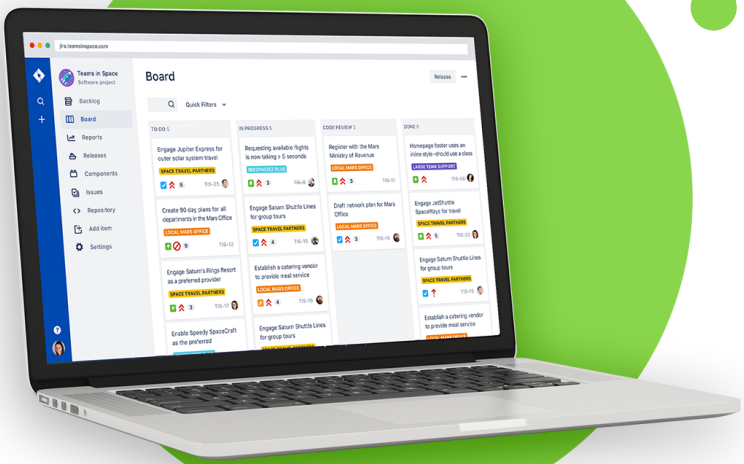
# JIRA ESSENTIALS FOR PRODUCT OWNERS

A Must-Have Project Setup Guide



# TABLE OF CONTENTS

# INTRODUCTION

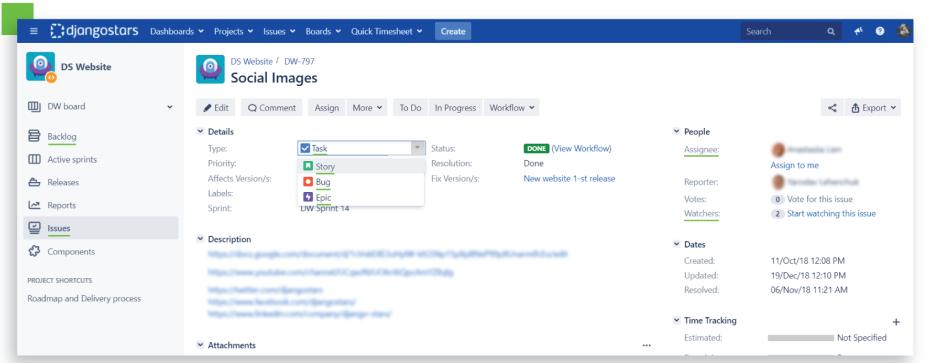
In the Agile world, when you work with any tech company they will surely train you how to work with them, as they may have their own rules. But to reduce the number of unnecessary communication you need to be prepared and know the basic tools you can use to manage and control the development team.

To master Jira, you should try our recommendations in practice. Here we'll show you the minimal set of tools which you as the Product Owner need to know to make effective use of Jira at the development stage of the project. Having them you will be able to manage the work of the development team in a convenient way, set the tasks in a clear for your technical partner manner and follow their execution transparently.

To be clear, we are not talking about what button to press to create an issue. Instead, you'll find out how to customize Jira to avoid common mistakes and get the work done in the most efficient way.

## TERMINOLOGY
## (THE DULL, BUT NECESSARY PART)

Don't worry, this won't take much time. Here, we'll explain a number of basic terms you should know to be "fluent" in Jira and, well, to completely understand what we'll be discussing.




**Assignee —** a person responsible for the performance of a task.

**Backlog** — the list of set-in-advance tasks for future development. It is necessary for the team, so they can know what tasks will come next and plan how to solve them.

**Bug** — a mistake at the production stage (total failure).

**Epic** — the highest-level and the most important task – the one that is so massive that cannot be done in one sprint.

**Issue** — a task in “Jirish”.

**Scope** — the number of tasks that should be done within a defined period of time to develop certain features in product so it can become an MVP.

**Story** — a task handled by several people. Usually more complex than a simple task.

**Story bug** — a mistake revealed during the testing; it is the part of the iteration and it will be fixed by the end of the sprint.

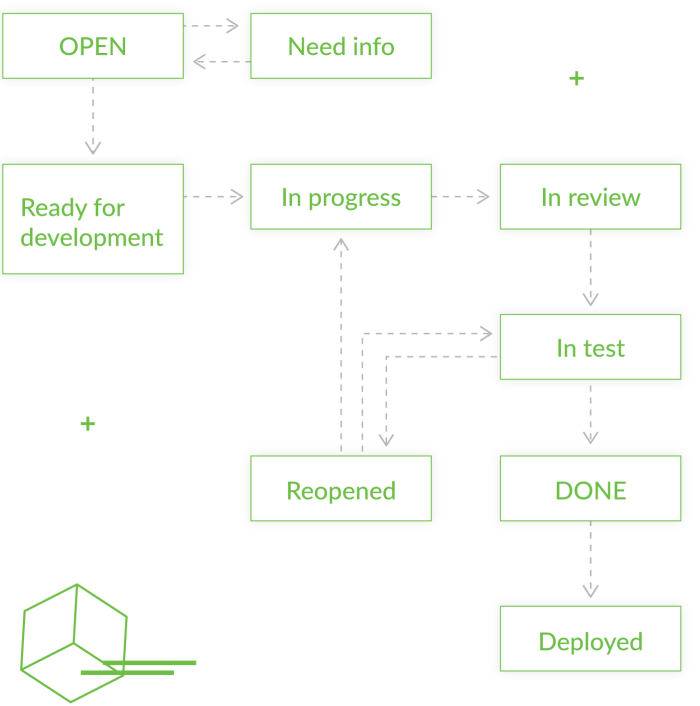
**Task** — the meaning is direct enough, but the thing is that a task is handled by one person.

**Watcher** — a person who controls or follows the execution of the task; the one who receives notifications about everything related to the task.

## HOW TO START WORKING WITH JIRA

First and foremost, the central concept of Jira is that of **workflow.** This means that your project and its parts will progress through several stages of the development process, with the Jira workflow guiding you through it and to measure completion.

Issue Workflow Example



To make Jira work properly, **you need to identify the status of all tasks to create the major workflow, which will cover all the situations in the project.** The workflow will not act properly without planning, and that's the first thing you should do.

## STEP 1. FORM THE PROJECT DOCUMENTATION

To make sure that nothing will be missed in the development process you need to document — in detail — the project requirements, features, and functions the MVP must include. You should provide the team with as more information about the project as possible. Surely, the details will be discussed in the course of the tasks' analysis. But if you want to predict more or less precise deadline and minimize the risks of the development, you should define the MVP frameworks.

> *Tip: For a mature organization, the best approach to this will be to use the Confluence add-on connected to Jira. Confluence is a cloud collaboration tool where you can create massive descriptions of project requirements. It allows you to edit, organize and structure data in any way. Thus, when you have written down everything concerning some big issue, you can link the task in Jira to the description in Confluence. It*

*allows you to keep all the project documentation in one place. If you keep all this directly in Jira tasks, sooner or later you won't be able to find the most up-to-date information.*

**What you must not do is to keep requirements for complex product features in Jira tasks, as it won't be long before you're unable to form a single picture out of it. It's better to keep this in one place, like Confluence.** If you're a newbie and don't have Confluence, you can write everything down in a PDF doc and link it to Jira tasks. Of course, you'll need to track the current activity in Jira and mark in the PDF doc what has been done, what is in progress, and what should be done.

## STEP 2. CREATE A BACKLOG AND SCOPE OF THE PROJECT

The scope of the project is needed so the team can know what they have to do to make the first version of the product. Be specific in your definition of the MVP (minimal viable product) and what features it must have. In the backlog, you set the tasks so the team can know what's coming next. To make this properly, you must be familiar with the main types of tasks in Jira.

## Types of tasks in Jira

**Epic.** The most convenient way is to write down the task in Confluence and link it to Jira ticket using the “Epic” tag.

> *Example of an epic: “Marketing Tracking and Analytics”*

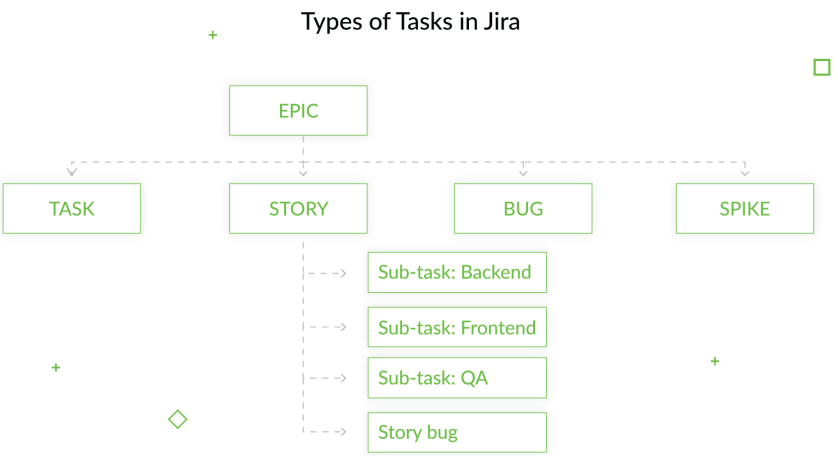
Using epics is also important, as they are displayed in Jira like labels for issues in a list view in backlog, so you can easily track which tasks or stories are related to which epics.

**Tasks, stories, bugs and spikes.** Issues that must be done. These are executed in accordance to prioritization, which will be discussed below.

> *Example of a task: “Update field’s naming on property evaluation page.”*

**Bug/Story bug.** While a bug is an issue that requires fixes at the production level, a story bug is a part of the development process during the sprint. It is an error found by QA engineers in the increment under current development.

*Tip: While reporting bugs and story bugs, you should be extremely careful and remember about ethical communication. You should be precise about which type of bug you are speaking about, as you may communicate this information to all levels of your company and any misunderstanding may strongly affect the team. While the difference between bug and story bug may not mean much to you, it means a lot to developers. And reporting to the administration about a story bug by naming it a bug (which is a much more serious type of mistake, as we've defined above) may strongly demotivate the team and even be grounds for a financial penalty.*



## Creating the backlog

As PO, you have to describe the epic — the super-task, in which you must define your MVP, since the MVP may not require the full execution of the epic. Afterwards, divide the epic into tasks according to your vision of the MVP. However, it's better if the team divides the epic into tasks, because they usually have their own approach of how to split the big item into tasks/stories for the development, while the PO may not know about it. In simple words, let the team do their work.

The Project's Backlog Example


### Creating the scope

When creating the scope, you have to keep in mind that the team plans the sprint in advance. So, if the sprint takes two weeks, it means that while they are working on the current sprint, they are already planning the next one. **So, provide the scope of tasks at least two weeks before they plan to finish the sprint.**

> *Tip: Agile is cool. It's not hard to build agile on the development side. But on the business side, with its complex decision-making process, quick creation of tasks may be a*

*problem. Thus, so as not to be a bottleneck, always be 2 weeks ahead of the team. Think globally.* ***If you don't know what to give to the team for the future development, don't panic. Take your time, and let the team take the engineering sprint.*** *While they work on the code upgrade to improve the speed of the product, its optimization and flexibility, you'll have time to think about the future strategy.*

## STEP 3. PRIORITIZE TASKS

**Prioritization plays a major role in the organization of the process.** Priority gives the team an understanding of what's most important to do and which problem is best to solve at any given moment. They won't need to bother you to make this decision for them.

Do not organize the priority of tasks in a list mode, like in Excel. Each PO as an Excel monster is used to this type of prioritization. However, in Jira, it may cause misunderstandings, as someone can accidentally move the task down the list, for instance. Use the label "priority" instead — critical, medium or low.

*Tip: Do not take all the critical tasks into the sprint, because the team can't do them all. You have to adequately evaluate the intensity of the work. The team has to know with what task to start with, what is the priority. A good practice is to allow the team to take the tasks into the sprint optionally. You still may ask them to take on particular tasks, but don't choose only high-priority ones. If you put low-priority tasks into the sprint, it doesn't mean that the team won't treat them attentively. It just means that they will know what to start with.*

## STEP 4. USE SPLIT-LEVEL LABELS

Usually, the development team includes members with different roles and competencies — backend, frontend, quality assurance, UI/UX design, etc. The problem is that imbalances can happen in the number of tasks assigned to different team members; either most of the tasks are for the backend or for the frontend. And when you scroll through the list of unlabeled sprint tasks, it's difficult to count how many tasks are for each role, as it's hard to see who the assignee is from the name of the task (it may be duplicated several times).

*Tip: Use labels like “FE”, “BE” and “QA”. They are visible in list mode, and you won't have to open each task every time you need to know who the assignee is.*

## STEP 5. CONTROL CHANGE REQUESTS AND HOTFIXES WITH LABELS

Hotfix is a change that is important to the PO, and it is normal when someone asks the team to fix something quickly. You may agree with the team that tasks labelled “hotfix” must go directly to development. Also there may be a “due date” field in hotfix (a desired due date).

However, you should ask the team to estimate whether they will be able to complete it in that period of time. It may work for them, but they might make a change request — i.e., substitute the already existing tasks with the hotfix. Or the team may have the buffer of time in the sprint for hotfixes. Using this label means that you don’t have to write to everybody in chats: “Guys, I need a hot fix”. One label saves a lot of time.

*Tip: Coordinate and control the execution of tasks with help from the "assignee" field. Under common practice, if you are in control of the task, you are an "assignee", and Jira will flood you with notifications about every action made to the task, and soon you'll just stop paying attention to these notifications and risk missing something critical. To solve this, agree with the team to handle "assignee" with extreme care. Using this approach, the assignee is the person who has to take action on the task in the current period of time. This means that when one person is finished with the task, the task is reassigned to the one who is supposed to check the work that's just been done, for instance. For you, it means that you can pay attention to the tasks waiting for your participation. You don't have to read thousands of notification e-mails from Jira. Just sign in to Jira and find the tasks in backlog that are assigned to you. It strongly reduces the amount of work, improves the visibility of tasks, and enhances the predictability of what you have to do. And the most useful thing is that you can proactively answer the team's requests.*

## JIRA REPORTS, AND WHY YOU NEED THEM

In the end, after you've read all the information above and mastered the use of Jira tools, you might be pleased to learn that among other convenient features, Jira gives you the possibility to achieve a full transparency of the sprint activity.

Tracking activity during sprint offers you one major advantage: the transparency of the process. In addition, it significantly simplifies conflict resolution and helps you make more accurate estimates for future projects.

During an open sprint, Jira identifies all actions around every task — who moves it, who changes it, everything. That's a report. And, thus, if a conflict arises, you can easily solve it with Jira's tracking system. However, for the report to be optimally clear and accurate, you have to avoid spontaneously adding tasks to the sprint. Because unpredictable and unplanned tasks evoke panic among the team. So just **handle a sprint as if it is something sacred.**

Also, Jira has a time log. Engineers log every task they are working on, and indicate the time they spend on it. Often, the

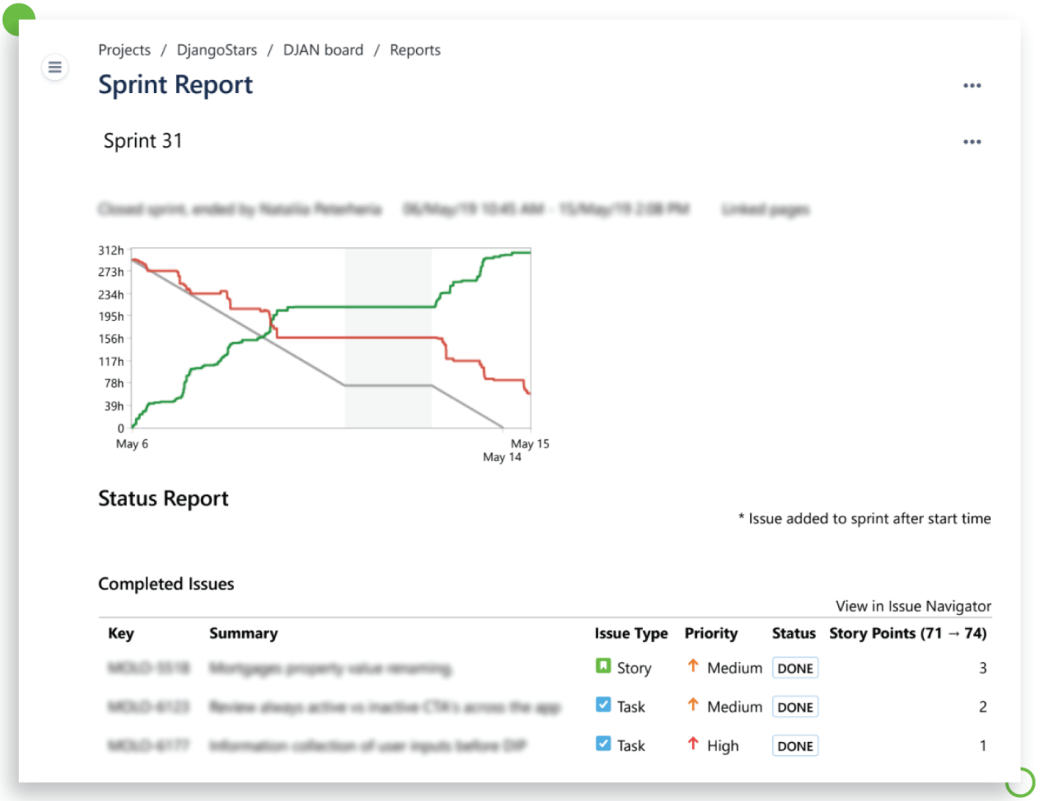
business people don’t understand why the team logs their communication time as well. Here, you should understand that the team needs time to discuss solutions, hold a planning meeting, or just to delve into the tasks in the backlog.

READ: Why Project Communication Is Important and Worth Your Money

> *Tip: To get control over the time spent on communication by the team, you can just ask them to describe what they've discussed in each log. This way, you will not only have control over cost distribution, but know what problems the team has, how you can help them, and what questions arise.*

Thus, if you use Jira the right way, it can provide accurate reports of how the sprint has gone.

On the horizontal axis of the chart are the sprint dates. As we practice 7-day sprint, here the sprint runs May 6–May 15 (with day-offs on the 10–12 of May). On the vertical axis we see the total number of hours in the sprint for the whole team.




- the time spent — the sum of all the hours logged against the tasks in the sprint
- the time left — the total estimated time in the sprint minus all the hours that have been logged—

Two color lines demonstrate the dependency between what was planned in advance and what has been done. The smooth and gradual burn down and burn up indicated that estimates were performed accurately and the team's performance matched them.

Namely, *the green line* reflects the actual time spent on the

tasks. To be clear, it is the sum of the hours logged for tasks during the sprint. That is why it grows towards the end of the sprint.

*The red line* displays the time left - the total time estimated for the sprint without the hours that have been already logged. If everything goes well, this line has to burn down completely by the end of the sprint.

You may wonder why the lines go straight in the middle of the chart. Those are days off.

This is a very short and basic description of what the report can show you. Go deeper to plan the next sprint more accurately, analyze whether planned tasks have been replaced with the new ones, find out if the members of the team logged more time than was planned.

Once you have set up and mastered the use of the mentioned tools you will have at your disposal an irreplaceable instrument that makes the management of all parts of the development process much easier. If handled right, Jira can give you the answer to any question, and reduce the number

of difficult and controversial situations. This makes Jira a transparent tool for tracking sprint activity and history.

## SUMMARY

To make Jira your effective helper in your next project just remember these five main steps.

1. **Do not ignore the project documentation.** Form it in advance and keep it in one place.
2. **Create a backlog and scope of the project.** The project scope and backlog will help your team to plan the development process.
3. **Use task prioritization.** If the team knows what is more important, they can do their work more independently.
4. **Use split-level labels.** Labels as “BE”, “FE” and “QA” will improve the tasks’ list organization.
5. **Use labels to control hotfixes.** The team will know what you want them to do without excessive discussion in chats.

And with the help of Jira reports you will be able to track all the work, money distribution and time spent. Using them in a right way you’ll be able to make more accurate estimates for future projects. So Jira gives you not only the control and transparent tracking of the active projects, but the tool to predict the future ones.

## ABOUT THE AUTHOR

# Nataliia Peterheria

PROJECT MANAGER @ DJANGO STARS

For more than 3 years at Django Stars, Nataliia has been responsible for multiple fintech projects with all levels of complexity. She believes that agile and technology rule the world and with her team she creates complex digital products that are easy to use. As an experienced Project Manager, she often shares her insights on project management and process organization. Find out more here.

## ABOUT DJANGO STARS

Django Stars is a technical partner with a business vision. For over 10 years we have been transforming business ideas into successful digital products, building them from scratch and supporting them as long as needed. Working with markets including UK, US, Switzerland and Singapore we are highly experienced in product development for fintech, banking, travel, and transportation industries and were recognized as one of the TOP-3 Python & Django Developers by Clutch.co. You may find out more information at djangostars.com. Have a project idea? Get in touch!

START NOW