**[https://jira.litepoint.com](https://jira.litepoint.com/projects/SYSTEMTEST/issues/SYSTEMTEST-4716?filter=allissues)**[/projects/SYSTEMTEST/issues/SYSTEMTEST-4716?filter=allissues](https://jira.litepoint.com/projects/SYSTEMTEST/issues/SYSTEMTEST-4716?filter=allissues)

[**https://lpbrcm.atlassian.net**/jira/software/c/projects/B89/issues/?jql=project%20IN%20(%22B89%22)%20ORDER%20BY%20created%20DESC](https://lpbrcm.atlassian.net/jira/software/c/projects/B89/issues/?jql=project%20IN%20(%22B89%22)%20ORDER%20BY%20created%20DESC)

1. **https://hevodata.com/learn/jira-api/**

**Jira**is a one-stop-shop for all of your Project Management requirements. It started as a **Software Development bug tracking tool** and now has matured into an **issue tracking** and **Agile Project Management** software. It was created by **Atlassian**, a leading Australian software company that specializes in software products for software development teams.

The **Jira**dashboard has a lot of handy features and capabilities that make dealing with issues a breeze. Jira, on the other hand, has evolved into a robust **task management solution** for a number of use cases, ranging from requirements and test case management to agile software development. The Jira family of products assists teams in effectively managing their work. The Jira Product Suite is designed for a variety of teams, including IT, Operations, Software, Business, and others.

Jira’s software is offered as a cloud-hosted SaaS i.e. **Jira Cloud** as well as a **self-managed solution i.e. Jira Server**. Jira Cloud is designed for speed and ease of use, whereas **Jira Server offers more autonomy and customizable options**. Jira’s cost is fairly affordable for small teams, but it skyrockets as the number of users grows.

**Jira REST APIs** are used to connect with the **Jira Server applications** **remotely**. For basic features like issues and workflows, the Jira Server platform provides a REST API. **Jira APIs** can be used to create apps, automate Jira interactions, or build any other form of integration.

Jira REST APIs use URI routes to give users access to resources (data entities). Your application makes an HTTP request and parses the answer to use a REST API. The Jira REST API communicates in JSON and supports conventional HTTP methods such as **GET**, **PUT**, **POST**, and**DELETE**. REST APIs are available for application-specific capabilities in Jira Software and Jira Service Management, such as **sprints**(**Jira Software**) and **client requests** (**Jira Service Management**).

When accessing data on Jira, there are two key security concepts to consider:

* **Authentication:**It determines the identity of the caller.
* **Authorization:** It determines which options are available to the caller. The actions that the app or integration can perform are limited by the user’s Jira permissions.

**By querying and updating data in your Jira site via the Jira REST API, you can interact with Jira**. Jira leverages the user used in the login process to authorize requests to the Jira REST API. For instance, if a user has access to Jira and authority to create issues in Jira, **he or she can use the Jira REST API to create an issue**. However, if the calling user does not have the authority to perform a specific action in Jira, they will be unable to do so using the Jira API.

One of the following authentication methods can be used with the **Jira APIs**:

* **OAuth:**To authenticate users, OAuth leverages request **tokens** provided by Jira. However, it requires more time and effort to set up this Authentication than the other two Authentication methods, but it is more flexible and secure.
* **Basic Authentication:**To authenticate, the Basic Authentication method relies on a predefined set of **user credentials**. Except for tools like personal scripts or bots, it is not recommended to use this Authentication method. It’s easier to set up, but it’s also a lot less secure.
* **Cookie-Based Authentication: In the browser, Jira uses Cookie-based Authentication**. You can rely on this when using the browser to call the REST API (for example, via JavaScript).

1. **How to Use the Jira API**

* [**https://dzone.com/articles/how-to-use-the-jira-api**](https://dzone.com/articles/how-to-use-the-jira-api)

Interactive API explorer for Atlassian products

* **https://marketplace.atlassian.com/apps/1211542/atlassian-rest-api-browser?tab=overview&hosting=server**

# Restful Web Services Tutorial in Java

[**https://www.journaldev.com/9170/restful-web-services-tutorial-java**](https://www.journaldev.com/9170/restful-web-services-tutorial-java)

**Java API** for RESTful Web Services (JAX-RS) is the Java API for creating REST web services. JAX-RS uses annotations to simplify the development and deployment of web services. JAX-RS is part of JDK, so you don’t need to include anything to use it’s annotations.

# An example of very lightweight RESTful web services in Java

**https://opensource.com/article/20/7/restful-services-java**

* Restful services build upon existing infrastructure and protocols, in particular, web servers and the HTTP/HTTPS protocols. An organization that has HTML-based websites can **readily add web services for clients interested more in the data and underlying functionality than in the HTML presentation**. Amazon, for example, has pioneered making the same information and functionality available through both websites and web services, either SOAP-based or Restful.
* The **HttpServlet** is at the center of Java's web technologies. A website or web service, such as the novels service, extends this class, overriding the do verbs of interest. A **Restful framework such as Jersey (JAX-RS) or Restlet** does essentially the same by providing a customized servlet, which then acts as the HTTP(S) endpoint for requests against a web application written in the framework.
* A servlet-based application has access, of course, to any Java library required in the web application. If the application follows the separation-of-concerns principle, then the servlet code remains attractively simple: the code checks a request, issuing the appropriate error if there are deficiencies; otherwise, the code calls out for whatever functionality may be required (e.g., querying a database, encoding a response in a specified format), and then sends the response to the requester. The HttpServletRequest and HttpServletResponse types make it easy to perform the servlet-specific work of reading the request and writing the response.
* Java has APIs that range from the very simple to the highly complicated. If you need to deliver some Restful services using Java, my advice is to give the low-fuss HttpServlet a try before anything else.

# Creating a REST Web Service With Java and Spring (Part 1)

<https://dzone.com/articles/creating-a-rest-api-with-java-and-spring>

In the modern world of interconnected software, web applications have become an indispensable asset. Foremost among these web applications is the Representational State Transfer (**REST**) web service, with **Java becoming one of the most popular implementation languages**. Within the Java REST ecosystem, there are two popular contenders: [Java Enterprise Edition (**JavaEE**)](http://www.oracle.com/technetwork/java/javaee/overview/index.html) and [**Spring**](https://spring.io/). While both have their strengths and weaknesses, this article will focus on Spring and create a simple order management [RESTful](https://stackoverflow.com/a/29648972/2403253) web application using Spring 4. Although this management system will be simple compared to the large-scale RESTful services found today, it will nonetheless demonstrate the basic thought process, design decisions, and implementation tests required to create a [Level 3 (hypermedia-driven)](https://martinfowler.com/articles/richardsonMaturityModel.html) Spring REST web service.

By the end of this article, we will have created a fully functional Spring REST order management system. While the source code illustrated in this article covers the essential aspects of the order management system, there are other components and code (such as test cases) that support the main service that are not shown. All of the source code, including these supporting aspects, can be found in the following GitHub [repository](https://github.com/albanoj2/order-rest-backend):

# Restful web services example in java

<https://java2blog.com/restful-web-services-example-java/>

# Creating a REST Web Service With Java and Spring (Part 1)

<https://java2blog.com/restful-web-services-example-java/>

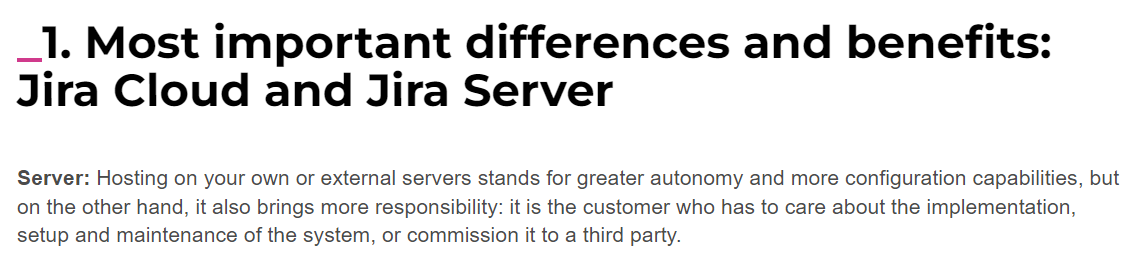
## Welcome to Apache Maven

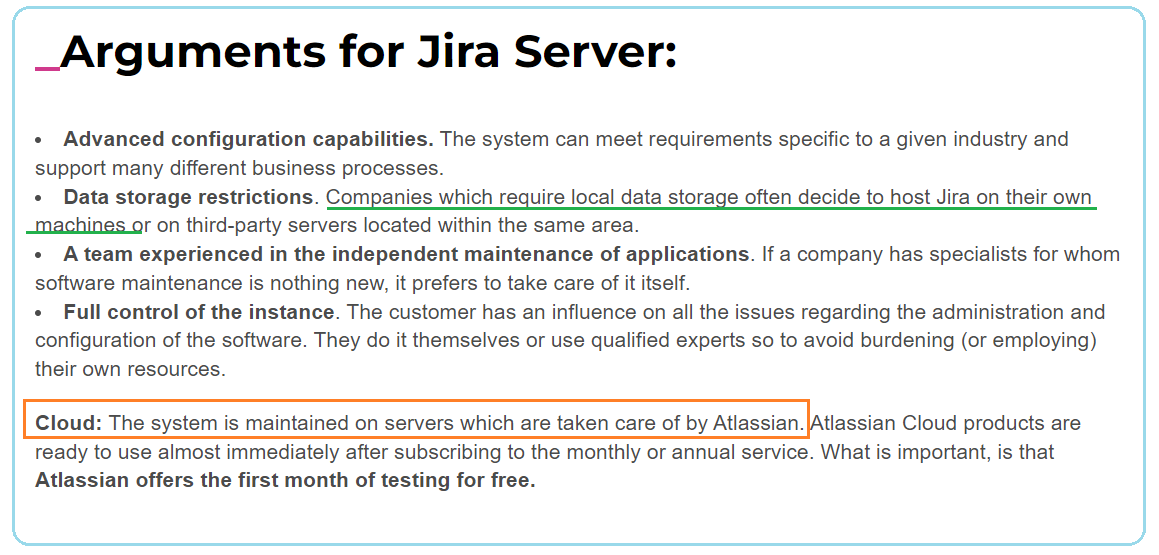
<https://maven.apache.org/>

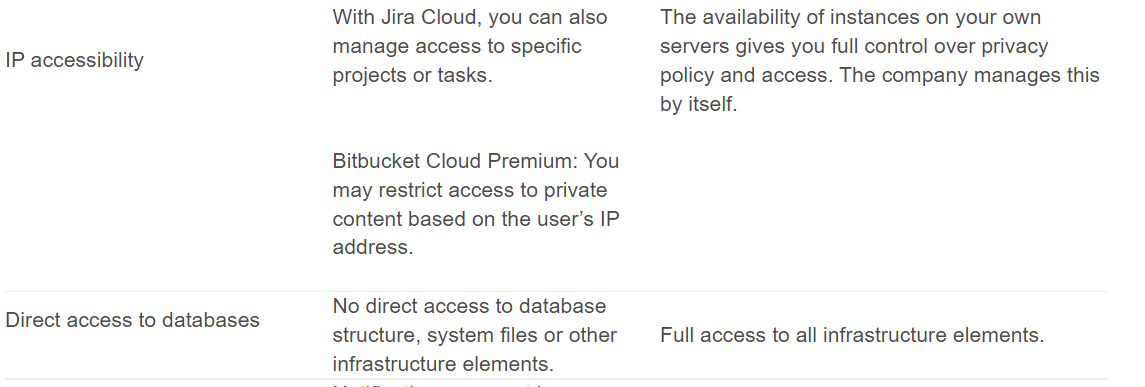
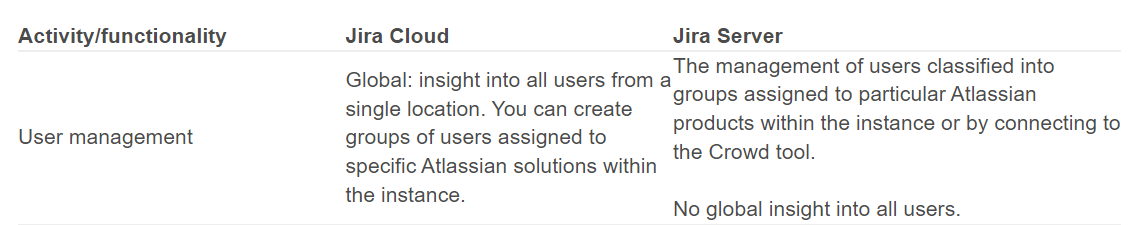
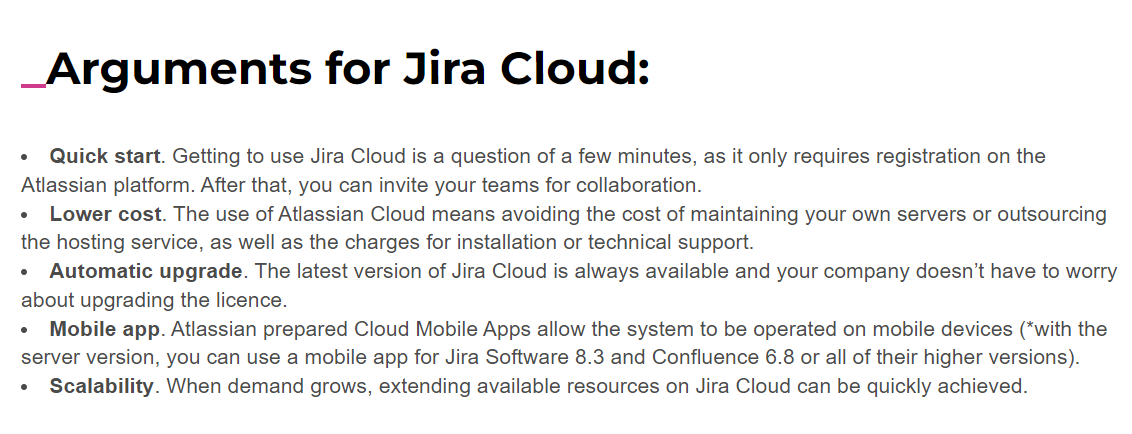
# Building a RESTful Web Service with springUse the internet from the command line with curl

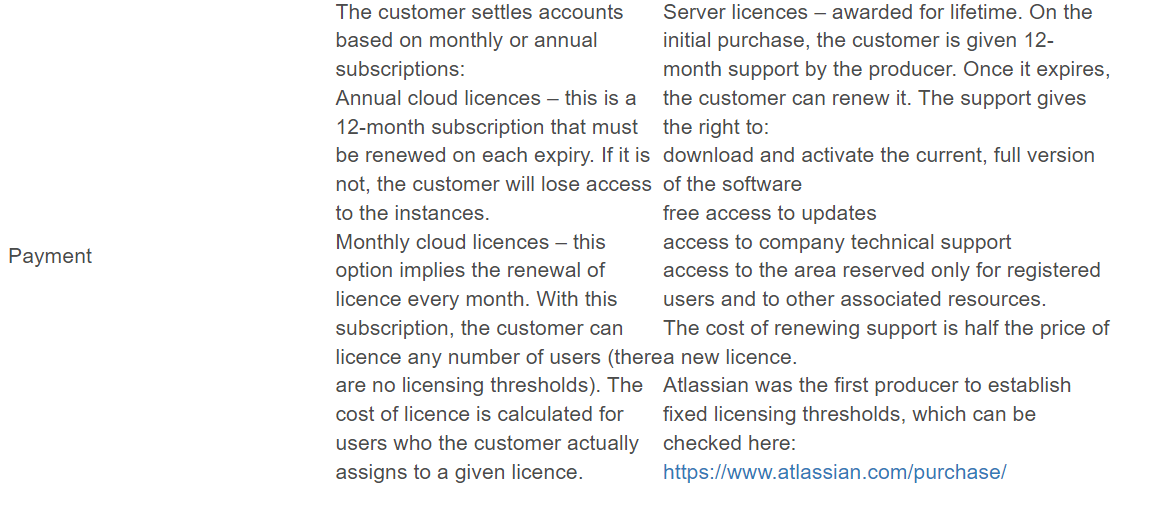
# https://opensource.com/article/20/5/curl-cheat-sheet

# <https://spring.io/guides/gs/rest-service/>









When considering **the installation of Jira** one of the first decisions is choosing Jira Cloud or Jira Server. We’re going to break down the strengths and weaknesses of each, so you can make [the best decision for you](https://www.savvior.com/).

There are massive benefits for cloud based solutions. [The pros of Jira Cloud](https://www.atlassian.com/software/jira/agility?_mid=3edeefd19095ce66d02b1b9e22e18e88&aceid=&adposition=1t1&adgroup=9124375702&campaign=189421462&creative=230411915164&device=c&keyword=jira%20cloud&matchtype=e&network=g&placement=&ds_kids=p19481846897&gclid=Cj0KCQiAus_QBRDgARIsAIRGNGidtUsIKT2LnwhJmojSPq6pUwNXZ-zFwFrxYe3jbJaAsgquLCkgPssaAkvtEALw_wcB&gclsrc=aw.ds) are the fairly universal selling points of all cloud based solutions. The 3 key benefits to Jira Cloud are:

1. No infrastructure requirements
   * Standing up a cloud based solution requires no capital expenditure.  You do not have to deploy virtual machines or servers to get up and running. Also, you do not have the expense or hassle of maintaining these servers.
2. Affordable (especially for small business)
   * Prices start at $10 a month
3. Automatic software updates

## The servers are automatically maintained so you’ll get regular updates to both the software and the servers, including security updates. This means you don’t have to waste time maintaining them, and your team will have more time to focus on things that matter. Where Jira Cloud Loses

The largest con of working with Jira Cloud is **certain Admin restrictions**. For example, at our company, we were unable to sync our project with another remote instance using some popular add-ons as they were not compatible with our Jira Cloud instance. Connectors for Google drive integration were also not available for our cloud installation.

Another con of working with Jira Cloud is **not being able to access databases directly**. Jira’s API is fairly robust, however, it does not expose 100% of all the functions you may need if you plan on doing deep application integration with Jira. For example, currently, there is no way to copy an existing agile board via the API if one were to try to stand up a new board using a previous board’s settings.

## Where Jira Server Wins

If you need to sync multiple Jira installations, need more robust options for customization, or have larger disk space requirements than what is offered out of the box with Jira Cloud, than you may want to [give Jira Server a look](https://www.atlassian.com/software/jira/download). Please bare in mind, however: Jira Server caps its users at 2,000 -- which is more than enough for the vast majority of cases.

Further, you may work for an **organization with sensitive data** where you may not choose to use a SaaS business to **host your data for you**. It is often corporate policy that such information not be exposed beyond the internal networks. In such an instance, Jira Server would be the way to go.

When deciding between installing Jira Cloud or Jira Server, it’s all about what your needs are. Jira Cloud will offer more integration and easier maintenance and use. Jira Server will allow more control and security but at the cost of simplicity. Whatever you decide we’re here to help. Let [our team of Jira experts](https://www.savvior.com/contact-us) work with your company to implement and integrate Jira into your workplace.

### Cloud

Atlassian Cloud allows you to run Jira, Confluence, Bitbucket, and other software while Atlassian hosts and runs your Jira deployment. In addition to the standard benefits that come with any quality cloud hosting service, like scalability and reliability, Atlassian Cloud offers the convenience of a quickly available Jira instance that requires only minimal setup. You can also take advantage of automatic updates to your Atlassian products.

#### **Who should use Atlassian Cloud?**

In general, Atlassian Cloud makes the most sense for organizations that want a software-as-a-service (SaaS) experience and don’t require the customizability or control offered by Atlassian Server and Data Center. It’s also a good choice for organizations whose needs fluctuate wildly from month-to-month, as its tiered pricing solution is based on the number of users who have accessed your instance in the last month.

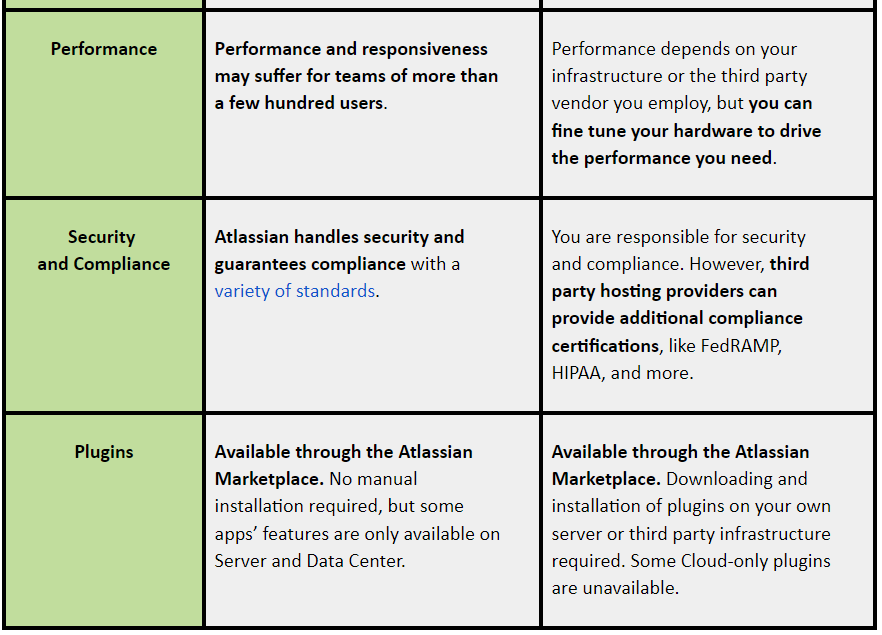
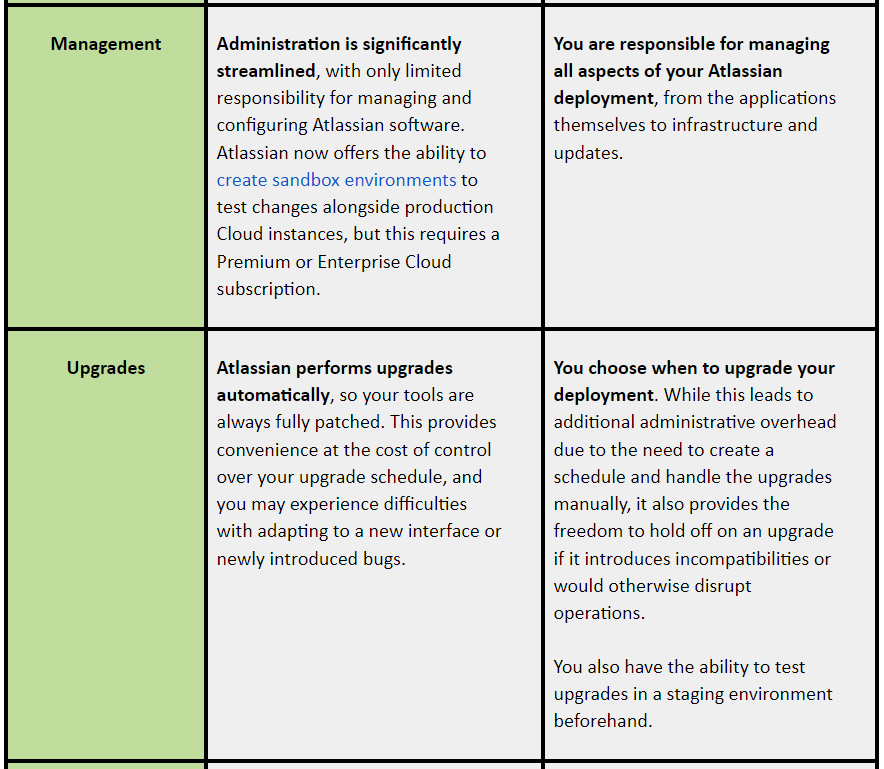
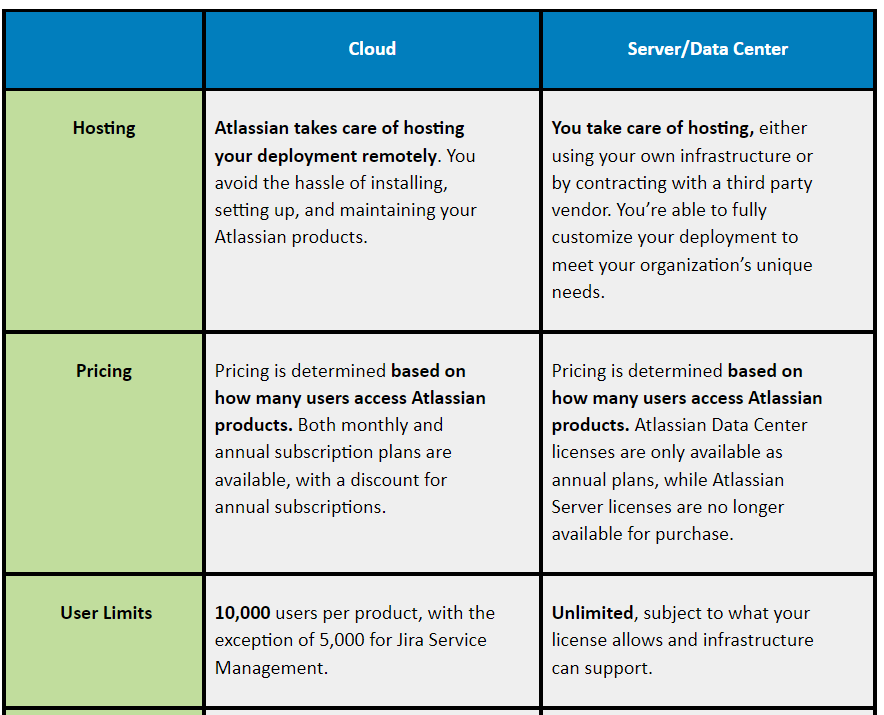
### Server and Data Center

Both Atlassian Server and Data Center let **you host Atlassian’s products on your own infrastructure or by using a third party cloud provider**. Running and maintaining Atlassian software yourself does require more work — and technical expertise — than using Atlassian Cloud, but **it provides control over all aspects of your deployment, from updates to security**. In addition, depending on your setup, these options may perform better than Cloud.

As Atlassian Server has been deprecated as of February 2, 2021, and new licenses are no longer sold, Data Center is now the only option for organizations who are new to the Atlassian ecosystem. Thankfully, Atlassian Data Center provides superior security, customizability, and reliability compared to Server, though at a higher cost. Data Center is the most performant, robust option available, and it can meet the needs of even the largest enterprises.

#### **Who should use Atlassian Data Center?**

Organizations that require **control over their infrastructure due to performance, security and compliance, or integration needs should look to Data Center**. And if disruptions to service have a serious impact on your company, Data Center’s multiple node instances and high availability are great ways to maximize uptime.

You have a lot knowledge to process when you enter the Atlassian ecosystem. From deciding on the pricing model that makes sense, choosing which Atlassian platforms to adopt, to deciding what pricing model is best. You must make the right decision when choosing between Jira’s available hosting options, Jira Cloud and Server (now Data Center).

The main difference between Jira Cloud and Jira Server is **where you store your data**. Cloud solutions generally allow you to store your data on other servers. Your data is transferred from your computer to the operator’s servers. They are then stored and backed up. **Jira Server can be used as a self-hosting service**. Your instance is kept behind your firewall and security on your servers. **Your data is not sent anywhere**. **You store it in your own storage with backup.** It is a good idea to have additional storage off-site. This article will discuss Jira Cloud and Jira Server, as well as their key differences. 