Conda and pip are too identical yet are too different. Although some of the functionality of these two tools overlap, they were designed for different purposes.

Pip (comes built-in to Python)

Pip is the Python Packaging Authority's recommended tool for installing packages from the Python Package Index (PyPI). Pip installs Python software packaged as wheels or source distributions. The latter may require that the system have compatible compilers, and possibly libraries, installed before invoking pip to succeed.

So, different Python resources are delivered in the form of pip packages. Many times developers make use of different versions of a given library for different projects. In this kind of scenario, it becomes a hassle to organize different python packages. A simple example is doing web development projects and data science projects on the same machine. It will jumble up all the python packages. The solution of this jumble is **anaconda .**

Conda

Conda is an open source package management system and environment management system that runs on Windows, macOS and Linux. Conda quickly installs, runs and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language. Conda package and environment manager is included in all versions of Anaconda.

Quick challenge between conda and pip:

- **Package management**: Pip is a package manager that installs packages from PyPI (Python Package Index), while Conda is a package manager that installs packages from multiple channels, including the official Anaconda repository, the conda-forge community repository, and the PyPI repository. Conda's ability to manage packages from multiple channels gives it an advantage over pip in terms of package availability.

Due to the larger set of packages conda is the winner. ***Conda:1, pip:0***

- **Environment management:** Both pip and Conda allow you to create and manage isolated environments for your projects. However, Conda's environment management is more flexible than pip's, as it can handle dependencies for multiple languages and packages that are not available on PyPI.
  Conda has the ability to create isolated environments that can contain different versions of Python and/or the packages installed in them. This can be extremely useful when working with data science tools as different tools may contain conflicting requirements which could prevent them all being installed into a single environment. Pip has no built in support for environments but rather depends on other tools like *virtualenv* to create isolated environments.

  Due to flexibility, conda is the winner. ***Conda:2, pip:0***

- **Dependency management:** Conda has a powerful dependency solver that can automatically resolve complex dependency conflicts, while pip relies on the user to manage dependencies manually. This makes Conda more suitable for managing complex projects with many dependencies.
  Due to better dependency management, conda is the winner. ***Conda:3, pip:0***

- **Cross-platform compatibility**: Conda is designed to work on different operating systems, including Windows, macOS, and Linux, while pip is a Python-specific tool that may not work as well on other platforms.
  Due to better cross-platform performance, conda is the winner. ***Conda:4, pip:0***

- **Ease of use:** Both pip and Conda are relatively easy to use, but Conda has a more user-friendly command-line interface and documentation, which can make it easier for beginners to get started. *Due to similar commands and user picking pip as initial point despite good conda documentation, both are the winner.* ***Conda:5, pip:1***
- **Community support**: Pip has a larger community of users and developers, as it is the standard package manager for Python. This means that there are more resources available for troubleshooting and support. However, Conda also has a large and active community, particularly in the scientific computing and data science communities. *Bigger community is always a plus point, thus pip is the winner*. ***Conda:5, pip:2***

Tabular comparison of Conda and Pip

	conda	pip
manages	binaries	wheel or source
can require compilers	no	yes
package types	any	Python-only
create environment	yes,built-in	no, requires virtualenv or venv
dependency checks	yes	no
package sources	Anaconda repo & cloud	PyPI

Conclusion

- If you are building a simple project handled by very few to a single person for a short span of time. It is good to go with a simple pip.
- For Larger projects with many people working on it or one that uses different versions of dependencies, It is advisable to use conda over pip to make the process smoother.