Why Should You Use Docker?

You never have to waste time setting up your environment again!

We'll cover the following

If you don’t want to put production at risk because of an untimely update or a library upgrade, you should start learning Docker right now. Besides this, there are other reasons to learn Docker, which we will look at below.

Advantages of using Docker

1. **Industry demand**: As Docker promises an equivalent environment in both development and production, companies don’t have to test applications twice in different environments. As a result, Docker adoption by companies increases daily.

2. **Isolation from the main system**: As a developer, you will always experiment with libraries and different versions of programming languages. For example

with libraries and different versions of programming languages. For example, if you are testing asyncio support for one application that needs Python 3.7,

and you decide not to use it, you might need to uninstall Python 3.7 and install the previous version. With Docker, you simply remove the container. That's it. Zero complexities.

3. **Configurations**: There are a lot of different configurations required for every project. Maintaining a list of configurations is very difficult. Docker provides the capability to configure images with different configurations and tag images.

4. **Docker Hub**: Ever imagined sharing your machine like you share the code using Github? Docker's Docker Hub provides access to thousands of images that are configured with the environment so that when your code works in your machine, you can build images and share it all over the internet.

5. **Continuous integration support**: Docker supports CI tools like Travis and Jenkins. Docker images can be built and tagged with specific versions and deployed anywhere.

> **Travis** is a hosted continuous integration service used to build and test software projects hosted at GitHub and Bitbucket. **Jenkins** is a free and open-source automation server. It helps automate the parts of software development related to building, testing, deploying, facilitating continuous integration, and continuous delivery.