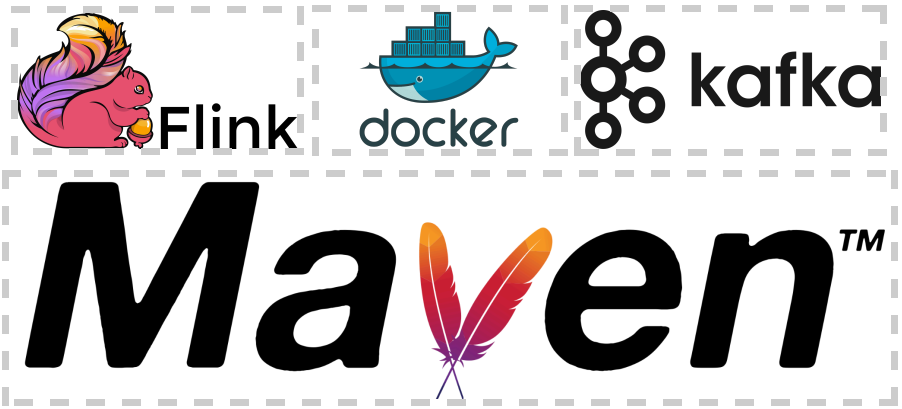
1. Install Requirements

This is the first out of the five guiding pdf files that aims to make it easier for the developer to get started with the project. In five sections, we will dive into the essentials of setting everything up successfully:

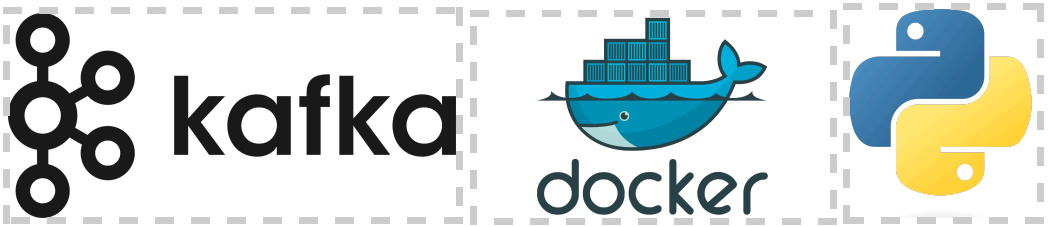


1) **Automation:** The information in this pdf file is based on the shell script "/instructions-executions/executionary/1. install-requirements.sh". Use this script as a reference to the explanations provided in this pdf. To be clear, you can just run "1. install-requirements.sh" and everything is taken care of automatically. With that said, this file is just providing elaborated explanations of the installing procedure.

2) **Setup Kafka Producer:** In the "/application" folder, there is a subdirectory dedicated for the production of real-time data using the Kafka technology. Fortunately, its Dockerfile takes care of installing the dependencies as the image is being built. Furthermore, in 1. "install-requirements.sh" two options are provided that are represented with a boolean variable:

```
SETUP_AND_RUN_CONTAINERS="True"
```

- **False:** Build the docker image without running the container
- **True)** Build the docker image and run the container



3) **Setup Flink Processor:** In the "/application" folder, there is a subdirectory dedicated for the processing and aggregation of real-time data using Flink technology. Since the module is written in Java with Maven, the code's dependencies must be updated each time a new addition is added by the coder or in the initial setup execution. This is done through a Maven-clean command through the CLI. You don't need to worry about this since it is automated with another shell script inside of "install-requirements.sh":

```
setupFlinkProcessor() {
    cd ../../application/flink-processor
    ./mvn-cmd.sh # Refresh the maven-dependencies
```

Just like the Kafka producer, the same variable is used as a developer configuration in "install-requirements.sh" to specify whether to only build the image or to run it as a container as well:

```
SETUP_AND_RUN_CONTAINERS="True"
```

4) **SQL Debugging:** To conclude this get-started tutorial, we will install the necessary dependencies so that you can debug and view the data instances being produced from the Kafka module and then processed into the PostgresSQL database via Flink. In "install-requirements.sh"

```
setupSqlDebugging() {
    pip install -r ../../debug-api/apis/requirements.txt
}
```

5) **The next step:** In the next get-started pdf, we will dive into the conceptual aspect of what is happening in the automated shell script "2. real-time-streaming.sh".