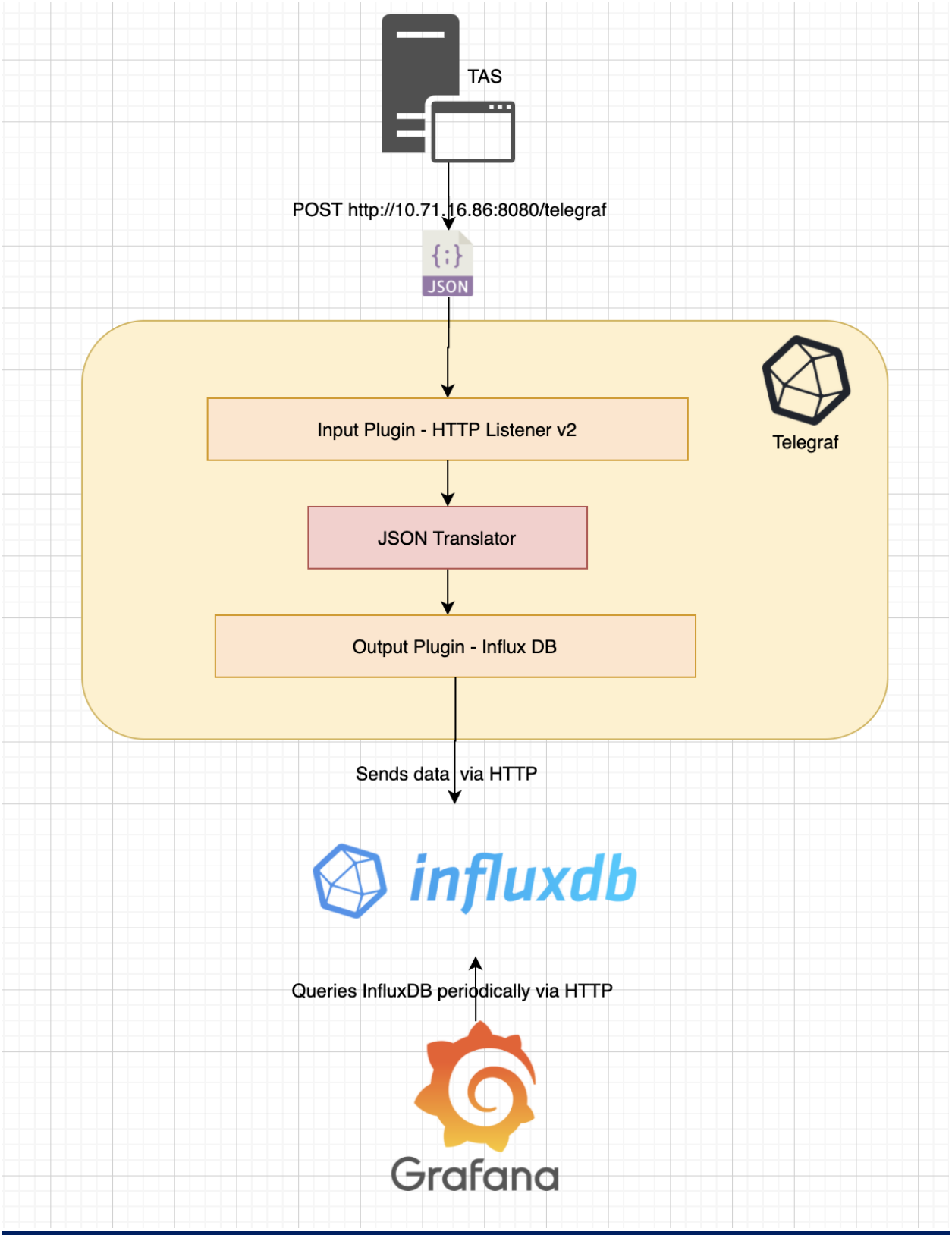
KPI Pipeline to Telegraf/Influx DB/Grafana

Table of Contents

1. Overview

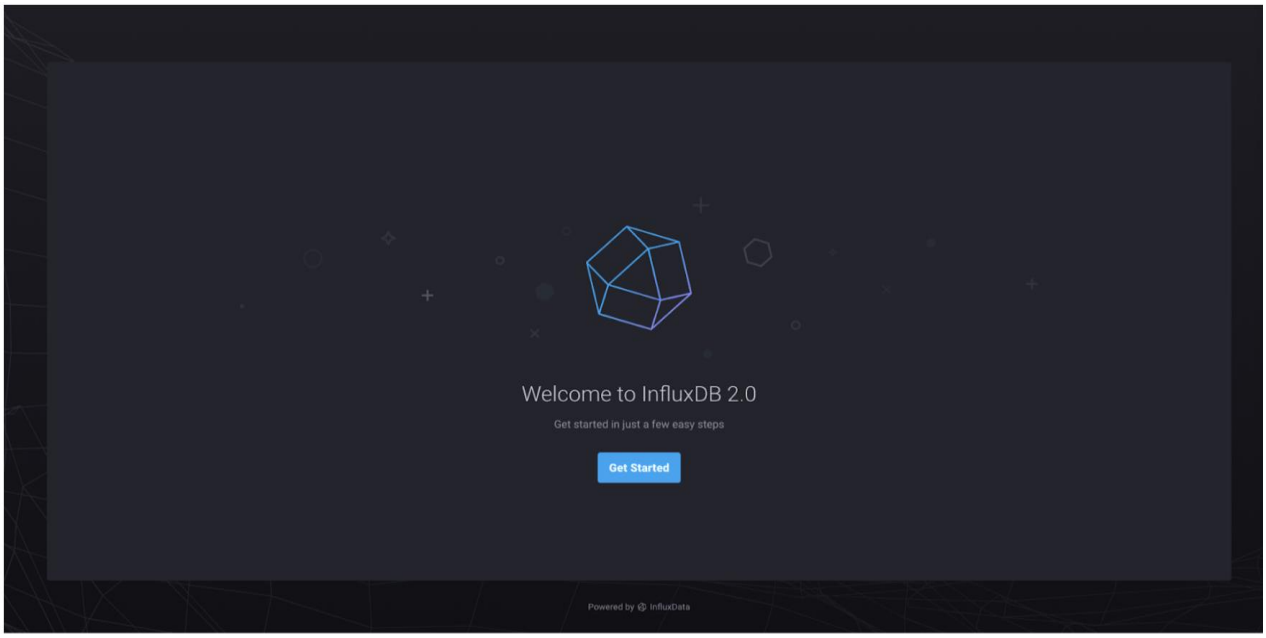
TAS provides a mechanism to HTTP POST a JSON format of the Favorite Measurements to a URL.

This documentation explains on how to publish “Favorites” KPIs into Grafana via Telegraf -> Influx DB -> Grafana Pipeline.

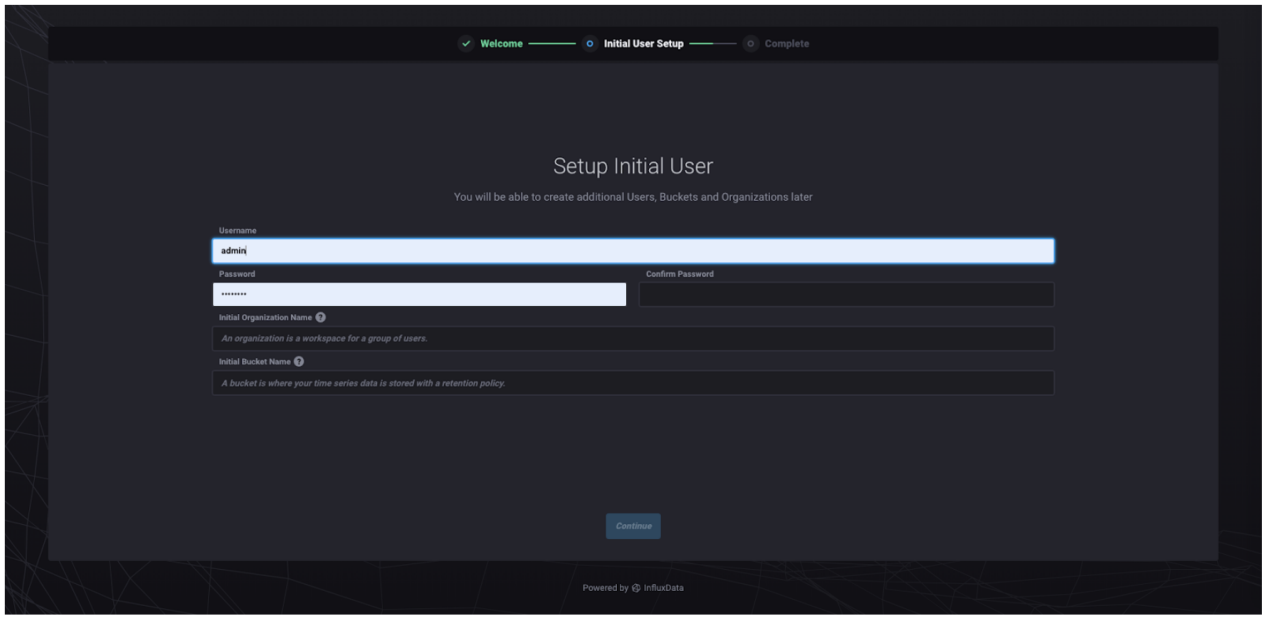

TAS
POST http://10.71.16.86:8080/telegraf
JSON
Input Plugin - HTTP Listener v2
JSON Translator
Output Plugin - Influx DB
Telegraf
Sends data via HTTP
influxdb
Queries InfluxDB periodically via HTTP
Grafana

2. Setup/Configure Influx DB

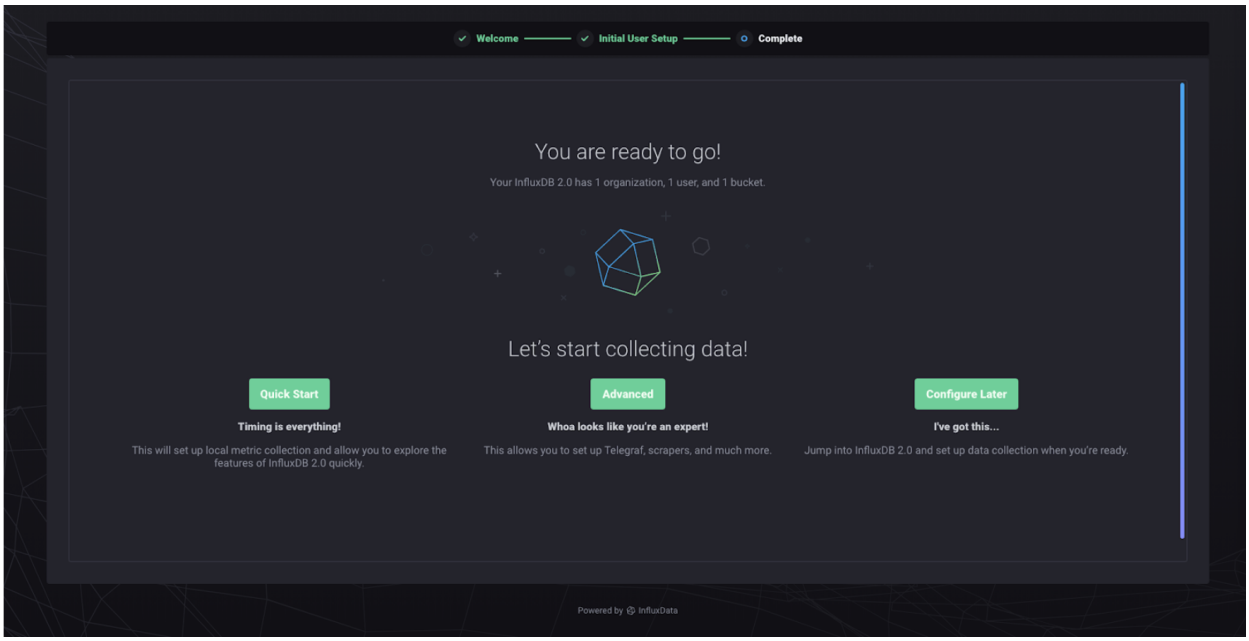
Influx DB is a time series database designed to handle high write and query loads.
Once the Influx DB is running, navigate to http://localhost:8086 and user should be greeted with this page:



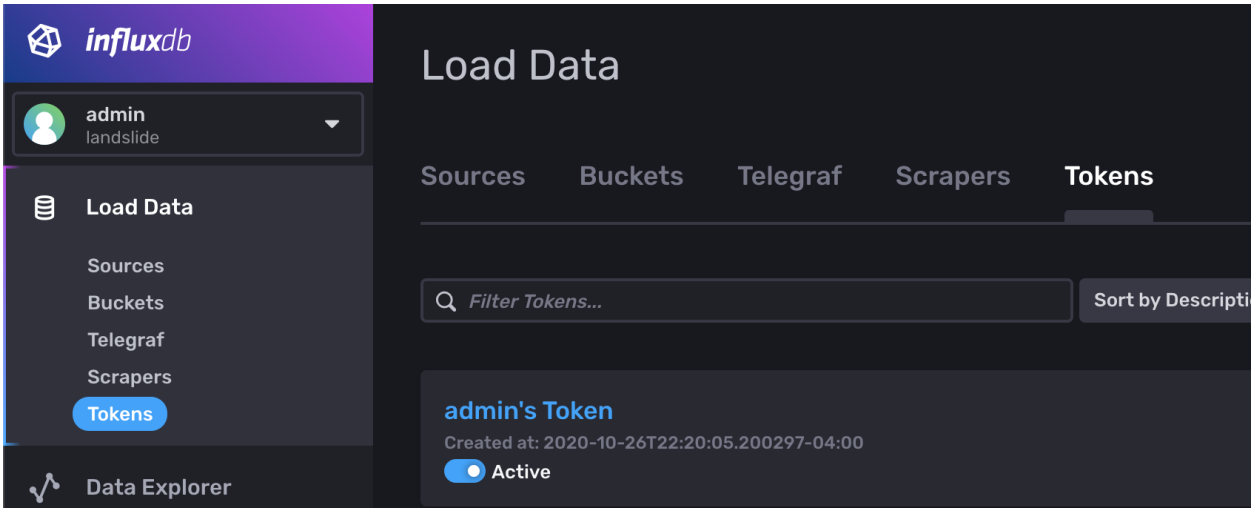
Clicking “Get Started” will take you to this page:



Organizations are an umbrella under which your data and queries are organized in 2.0. A bucket is the equivalent of a database. You can choose whatever organization, bucket name, username, and password you like. Clicking “Continue” will take you to this page:



Influx DB tokens for authentication can be found here:



3. Setup/Configure Telegraf

Telegraf is an agent written in Go for collecting, processing, aggregating, and writing metrics.

Telegraf is plugin-driven and has the concept of 4 distinct plugins. Out of which we are using the below 2 plugins for our use case:

- Input Plugins collect metrics from the system, services, or 3rd party APIs.
- Output Plugins write metrics to various destinations (Influx DB, MySQL etc.)

- **Service Input Plugin**

  In our use case, TAS is providing a mechanism to HTTP POST a JSON format of "Favorites" measurements to a URL. So, Telegraf is configured to use http_listener_v2 service input plugin.

  The configuration changes can be set in the Telegraf configuration file as below:

```
# Generic HTTP write listener
[[inputs.http_listener_v2]]
  ## Address and port to host HTTP listener on
  service_address = ":8080"

  ## Path to listen to.
  path = "/telegraf"

  ## HTTP methods to accept.
  methods = ["POST", "PUT"]
```

  Data format to consume is JSON format in our use case scenario. Each data format has its own unique set of configuration options, read more about them here:
  https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_INPUT.md

```
data_format = "json"
tag_keys = ["_run_","_test_"]
tagexclude = ["url", "host"]
json_name_key = "_target_"
json_time_format = "unix_ms"
json_timezone = "America/New_York"
```

Enable HTTP Basic Authentication by adding the below configuration in the configuration file. Also, the user can enable HTTPS by adding Service certificate and Key.

```
## Set one or more allowed client CA certificate file names to
## enable mutually authenticated TLS connections
# tls_allowed_cacerts = ["/etc/telegraf/clientca.pem"]

## Add service certificate and key
#tls_cert = "/usr/local/etc/telegraf/localhost.cer.pem"
#tls_key = "/usr/local/etc/telegraf/localhost.key.pem"

## Optional username and password to accept for HTTP basic authentication.
## You probably want to make sure you have TLS configured above for this.
basic_username = "admin"
basic_password = "password"
```

- **Output Plugin**

To start pushing Telegraf data to the Metrics platform, user just need to add an Influx output plugin for writing metrics to a specific bucket in Influx DB as described below:

```
###############################################################################
#                              OUTPUT PLUGINS                                 #
###############################################################################


# Configuration for sending metrics to InfluxDB

   [[outputs.influxdb_v2]]
  ## The URLs of the InfluxDB cluster nodes.
  ## Multiple URLs can be specified for a single cluster, only ONE of the
  ## urls will be written to each interval.
  ## urls exp: http://127.0.0.1:8086
  urls = ["http://localhost:8086"]

  ## Token for authentication.
  token = "CTQScxAZ0Vhwm_7vFYf0KYYMPskVV4weachMn2hu7TRH1JMVNUJuT9JlFOk3oU1_UuIP_CqAfzz6bECGwFz-MQ==="

  ## Organization is the name of the organization you wish to write to; must exist.
  organization = "landslide"

  ## Destination bucket to write into.
  bucket = "user1"
```

4. Configure and Run Tests on TAS (Test Administration Server)

TAS provides a mechanism to HTTP Post (JSON format) “Favorites” measurements to URL (Telegraf/Influx DB/Grafana)

By clicking “HTTP POST” on Test Session, User can configure below HTTP POST Output parameters which sends metrics on each interval to Influx DB via Telegraf:

- HTTP POST URL - The service address, port and path to host HTTP listener, which is the same configuration in Telegraf.

  Also, it allows multiple users to post different URLs based on Telegraf configuration as below example:

  #User 1 - http://10.39.132.97:8080/telegraf
  #User 2 - http://10.39.132.97:8090/telegraf

- JSON_target_ - This is a measurement name user can input which is conceptually similar to a table in Influx DB. Each user can have their own bucket and separate table sharing across the same Influx DB instance

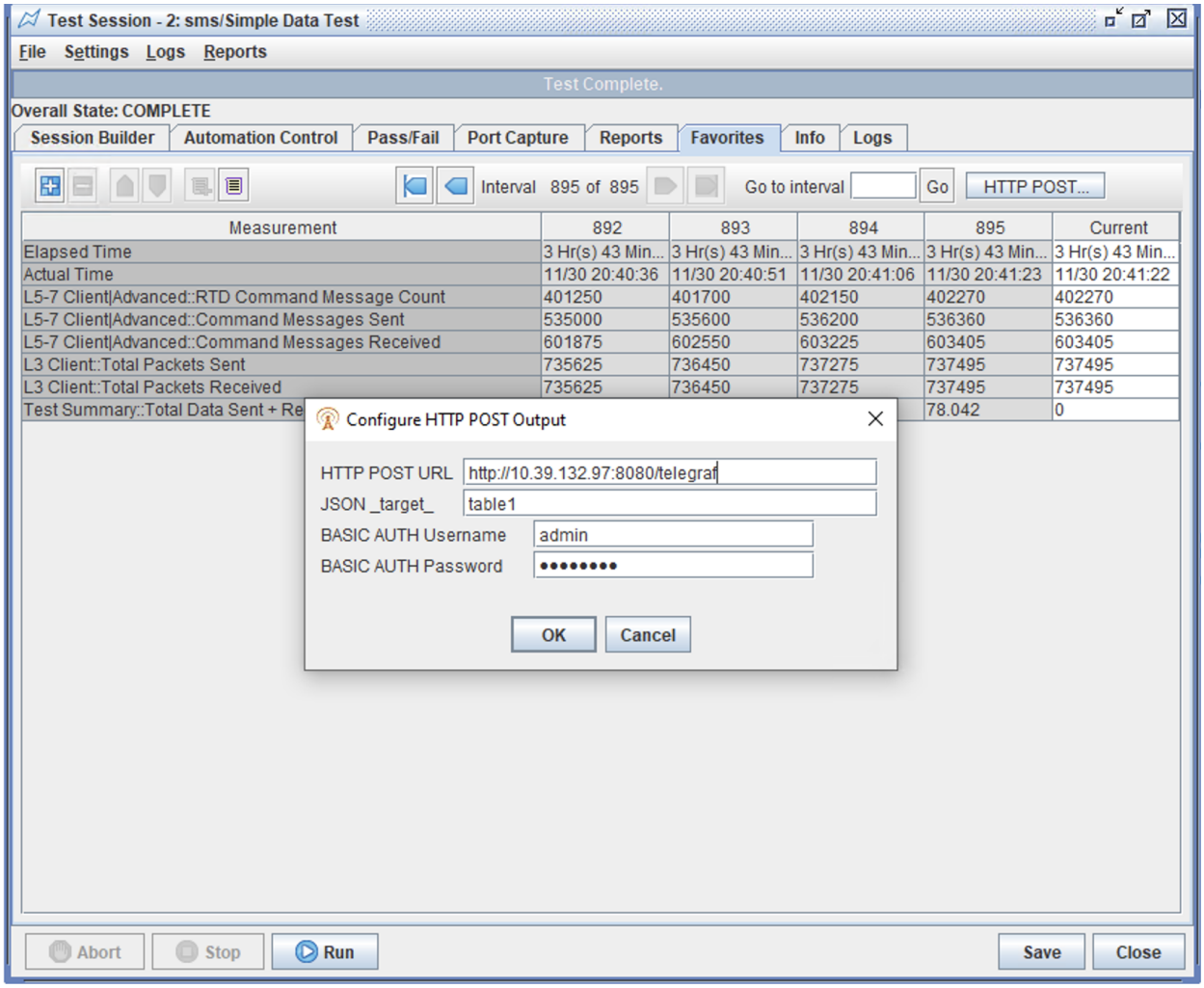
  Note- Target must be 1 to 32 characters only letters, numbers, _, -, or ..

- BASIC AUTH Username – HTTP Basic Authentication Username (configured in Telegraf)

  Note- Username must be 0 to 32 characters only letters, numbers, _, -, or ..

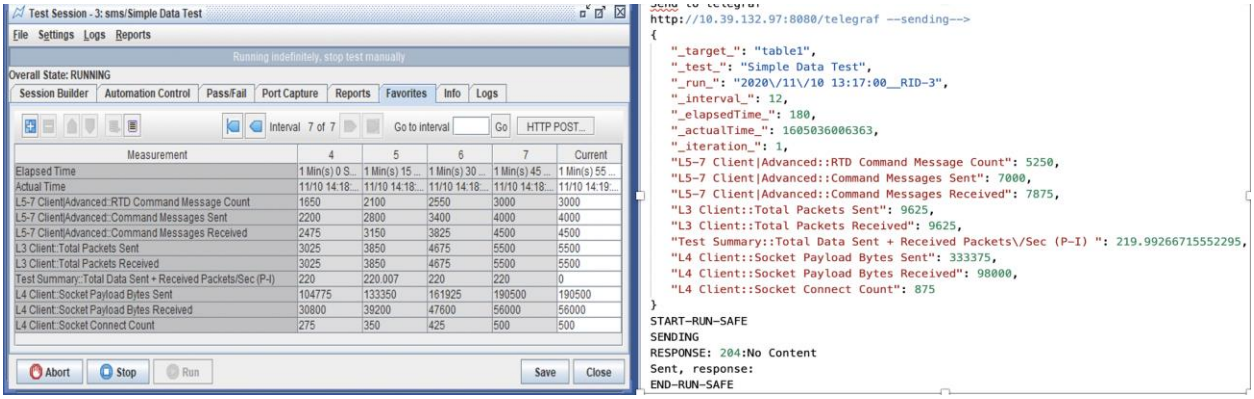
- BASIC AUTH Password – HTTP Basic Authentication Password (configured in Telegraf)

  Note- If username is filled out: Password must be 1 to 32 characters ASCII.



After configuring HTTP POST Output, click “Run” to run the test and send measurements to Telegraf.

The measurements get converted to JSON format in the background as below:



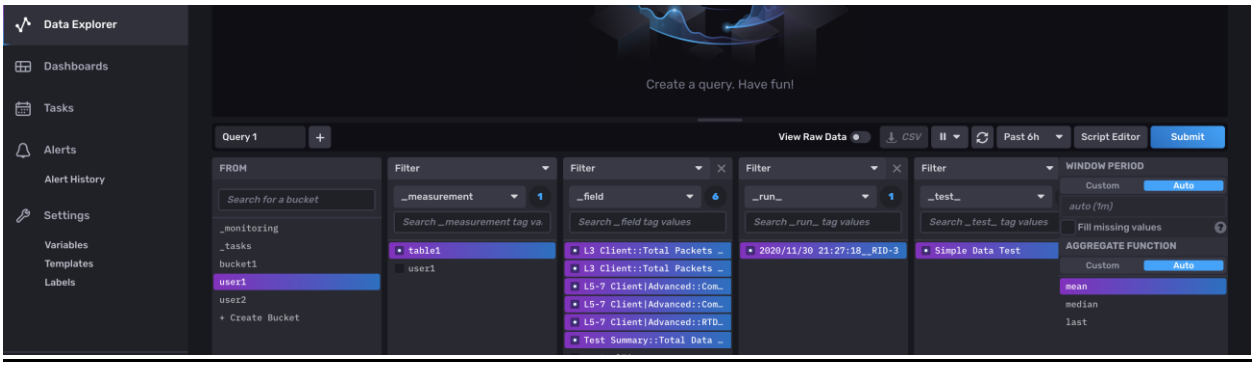
Start Telegraf to verify the live measurements being written to Influx DB in line protocol format.

```
HLMC02Y216BJGH6:telegraf spendyala$ telegraf -config /usr/local/etc/telegraf/telegraf.conf
2020-12-01T04:50:25Z I! Starting Telegraf 1.15.3
2020-12-01T04:50:25Z I! Loaded inputs: http_listener_v2 http_listener_v2
2020-12-01T04:50:25Z I! Loaded aggregators:
2020-12-01T04:50:25Z I! Loaded processors:
2020-12-01T04:50:25Z I! Loaded outputs: influxdb_v2 influxdb_v2 file
2020-12-01T04:50:25Z I! Tags enabled: host=HLMC02Y216BJGH6
2020-12-01T04:50:25Z I! [agent] Config: Interval:10s, Quiet:false, Hostname:"HLMC02Y216BJGH6", Flush Interval:10s
2020-12-01T04:50:25Z I! [inputs.http_listener_v2] Listening on [::]:8080
2020-12-01T04:50:25Z I! [inputs.http_listener_v2] Listening on [::]:8090
table1,_run_=2020/11/30\ 21:27:18__RID-3,_test_=Simple\ Data\ Test _interval_=332,L3\ Client::Total\ Packets\ Received=273625,_iteration_=1,L5-7\ Client|Advanced::RTD\ Command\ Message\ Count=149250,_elapsedTime_=4980,L5-7\ Client|Advance
d:Command\ Messages\ Sent=199000,L5-7\ Client|Advanced::Command\ Messages\ Received=223875,_actualTime_=1606798223616,L3\ Client::Total\ Packets\ Sent=273625,Test\ Summary::Total\ Data\ Sent\ +\ Received\ Packets/Sec\ (P-I)\ =220 1606798
239758030000
table1,_run_=2020/11/30\ 21:27:18__RID-3,_test_=Simple\ Data\ Test L5-7\ Client|Advanced::Command\ Messages\ Received=224550,_interval_=333,L5-7\ Client|Advanced::RTD\ Command\ Message\ Count=149700,Test\ Summary::Total\ Data\ Sent\ +\ Re
ceived\ Packets/Sec\ (P-I)\ =220,_iteration_=1,L5-7\ Client|Advanced::Command\ Messages\ Sent=199600,L3\ Client::Total\ Packets\ Sent=274450,L3\ Client::Total\ Packets\ Received=274400,_elapsedTime_=4995,_actualTime_=1606798238615 1606798
254769969000
table1,_run_=2020/11/30\ 21:27:18__RID-3,_test_=Simple\ Data\ Test L5-7\ Client|Advanced::RTD\ Command\ Message\ Count=150150,L5-7\ Client|Advanced::Command\ Messages\ Sent=200200,L3\ Client::Total\ Packets\ Received=275275,Test\ Summary:
:Total\ Data\ Sent\ +\ Received\ Packets/Sec\ (P-I)\ =220,_iteration_=1,L5-7\ Client|Advanced::Command\ Messages\ Received=225225,_elapsedTime_=5010,_actualTime_=1606798253615,L3\ Client::Total\ Packets\ Sent=275275,_interval_=334 1606798
269027224000
table1,_run_=2020/11/30\ 21:27:18__RID-3,_test_=Simple\ Data\ Test Test\ Summary::Total\ Data\ Sent\ +\ Received\ Packets/Sec\ (P-I)\ =220,_interval_=335,L5-7\ Client|Advanced::RTD\ Command\ Message\ Count=150600,_actualTime_=160679826861
5,_elapsedTime_=5025,L5-7\ Client|Advanced::Command\ Messages\ Received=225900,L3\ Client::Total\ Packets\ Received=276100,L5-7\ Client|Advanced::Command\ Messages\ Sent=200800,L3\ Client::Total\ Packets\ Sent=276100,_iteration_=1 1606798
284724974000
```

5. Exploring “Favorites” measurements on Influx DB

As the user can see, the “Favorites” measurements are being written to Influx DB via Telegraf.

In the Influx DB Data explorer, the first column is Bucket which is equivalent to database. The second column is measurement conceptually similar to table. The third column shows fields where metrics are stored. And the other columns are tag values where strings are stored.

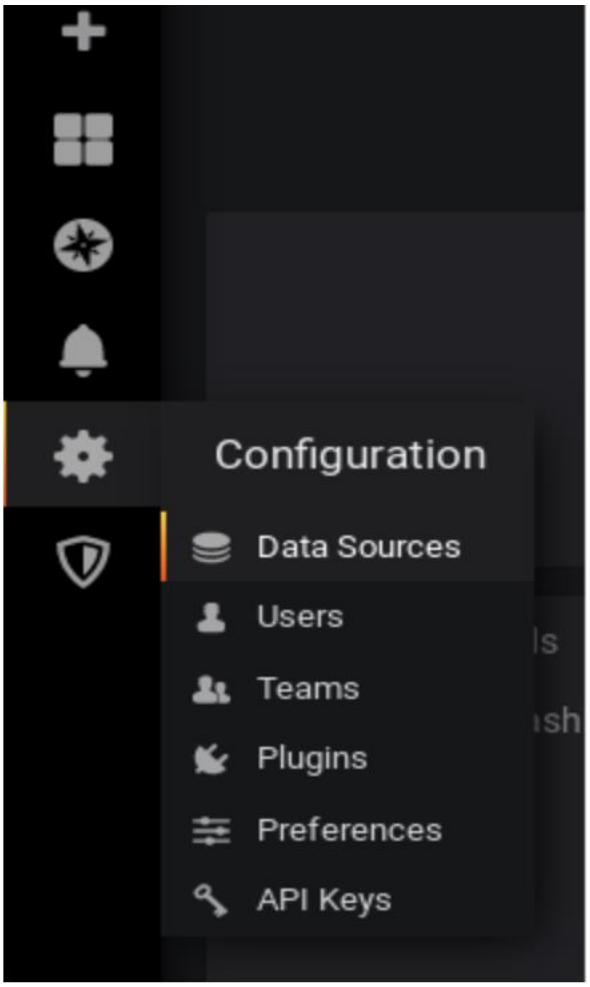


6. Setup Grafana

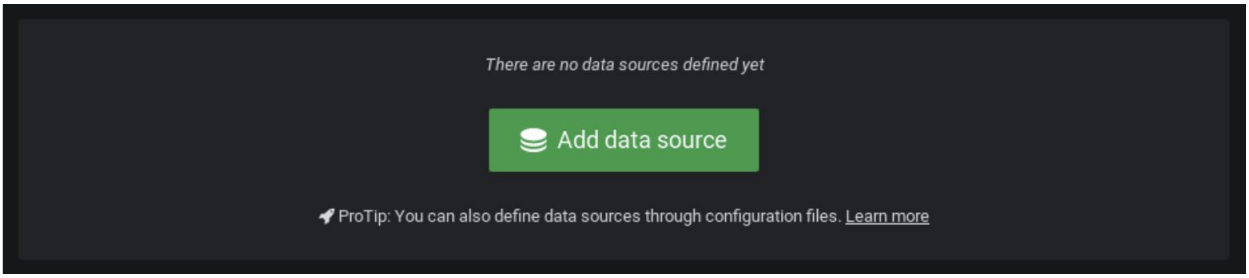
Grafana is a multi-platform open source analytics and interactive visualization web application. Each user can have their own pre-configured Grafana dashboard which periodically refreshes (configured to 10 secs) showing live measurements through visualizations.

- **Add Influx DB as a data source on Grafana**

  In the left menu, click on the Configuration > Data sources section.

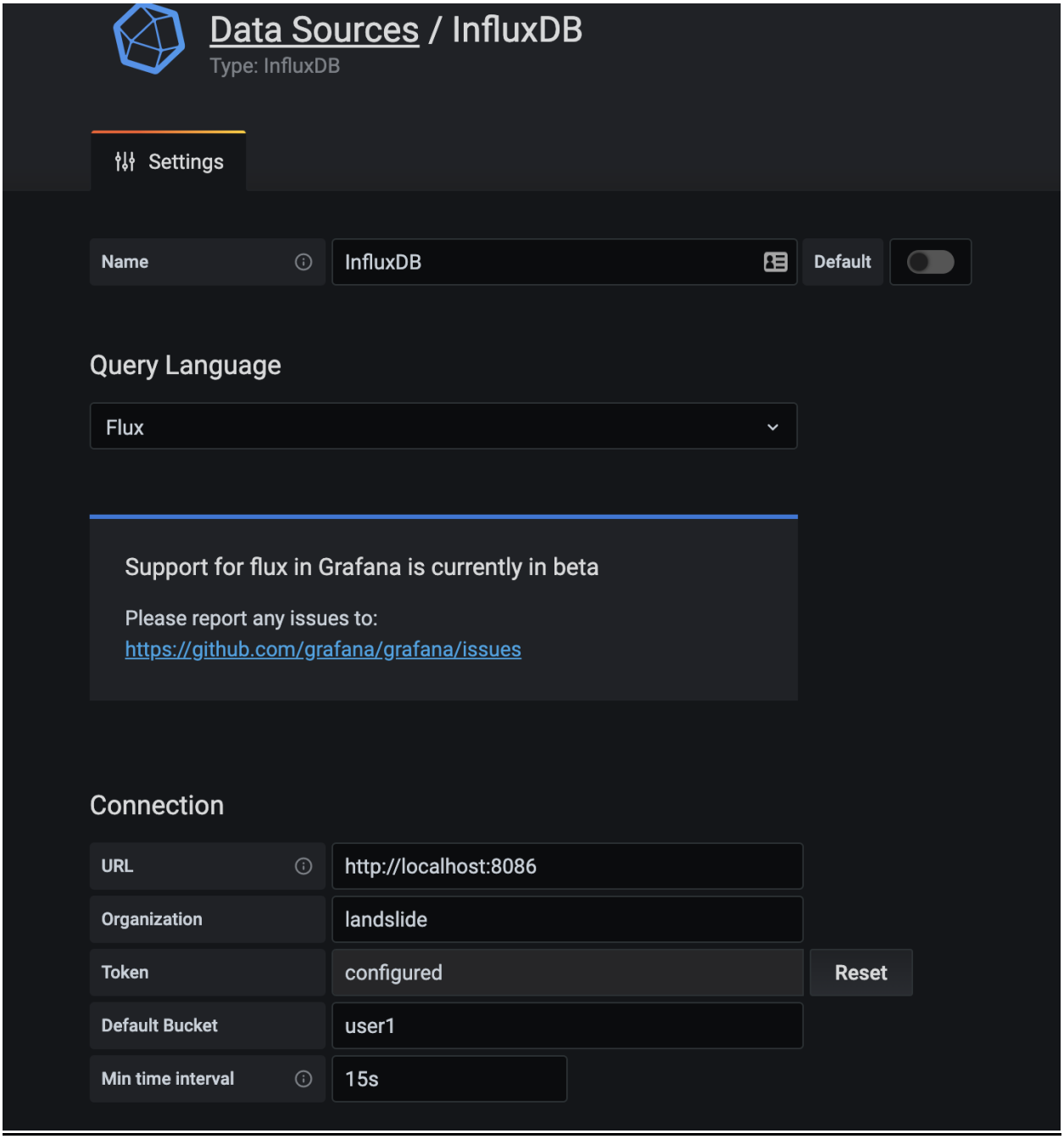


In the next window, click on “Add datasource”.



In the datasource selection panel, choose Influx DB as a datasource.

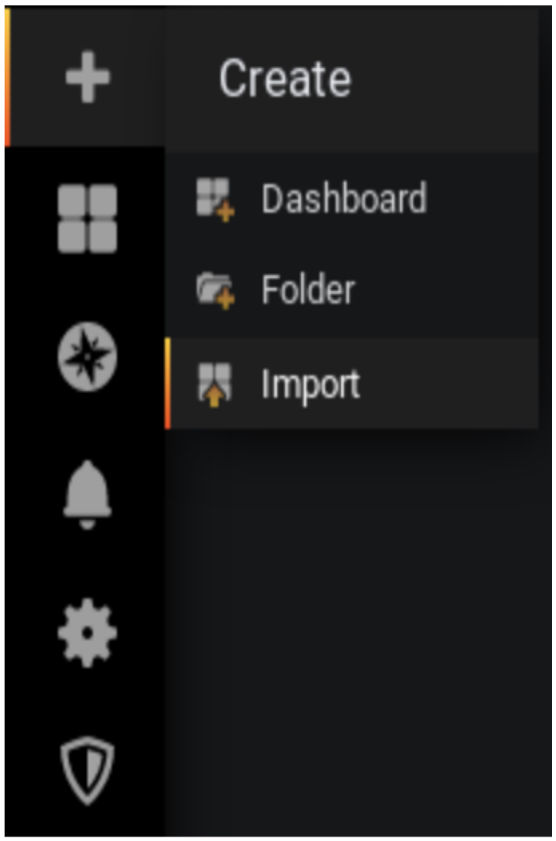
Here is the configuration you have to match to configure Influx DB on Grafana.



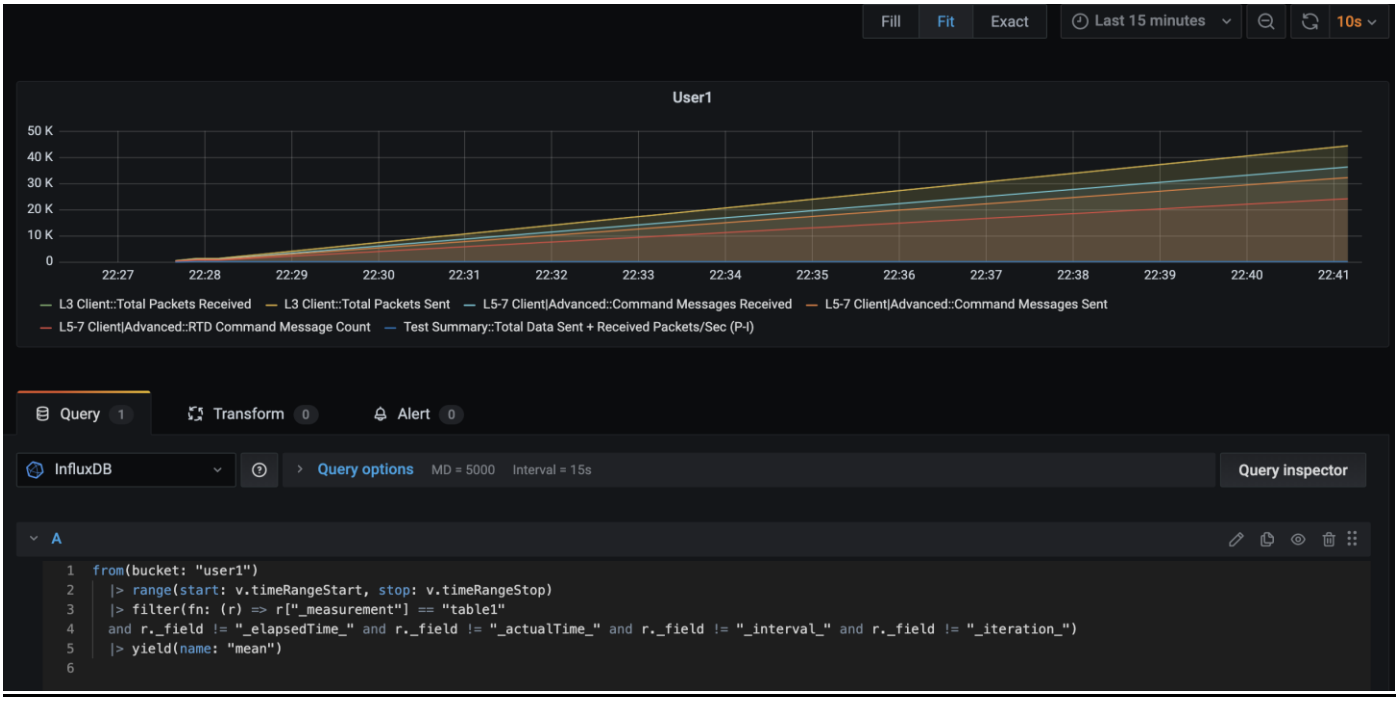
Click on “Save and Test”, and make sure that you are not getting any errors.

- **Add Flux queries in Grafana query explorer & Build a Grafana dashboard**

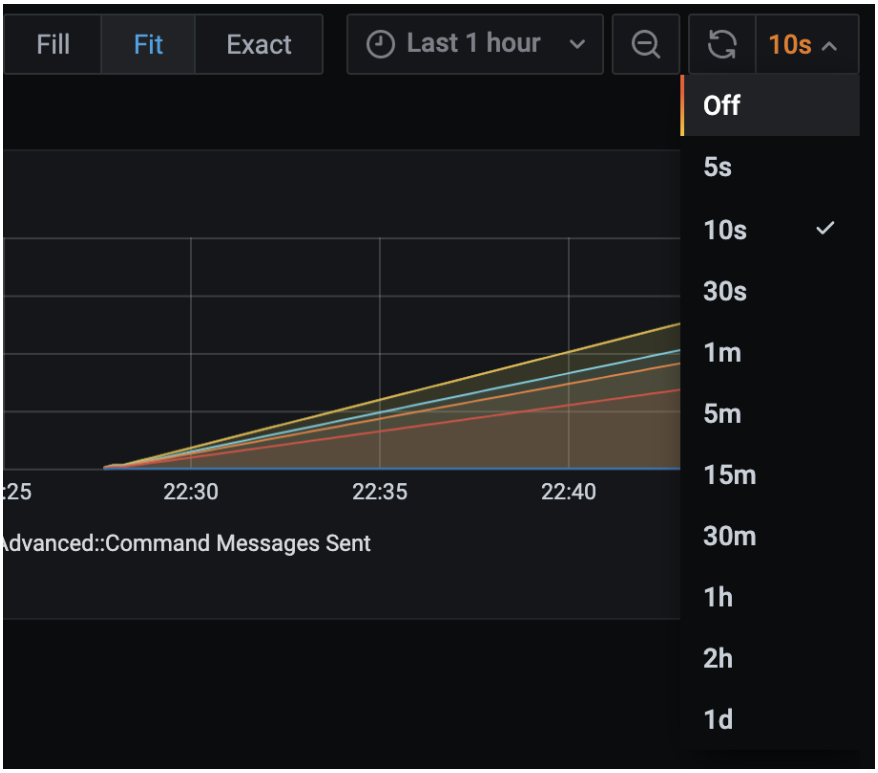
Click on “Create dashboard” and click on “Add new Panel”



Select Influx DB in the Query explorer and write Flux QL to create a Grafana dashboard



And configure the time period (say Last 1 hour) and interval (say 10 seconds) on the top right corner to refresh periodically and display live measurements every 10 seconds on the Grafana dashboard.



Live “Favorites” measurements on Grafana dashboard: