TAP Hands-On Workshop

Module 2: Visualizing Your Data with an App

Overview

This module uses a Java application to read a dataset from Hadoop Distributed File System (HDFS) that was created in Module 1: Performing Analytics on Your Data. The module uses Dataset-Reader to fetch data that was modified in the first module. The data is then presented to the user in a graphical form.

Estimated Length

This module will take approximately 1 hour to complete.

Setup

The app developer needs to prepare his or her local environment/computer with the tools required to perform the analytics and view the results. **Important**: You need to have access to an instance or to set up an instance of Analytics Toolkit (ATK).

- The software you need includes:
  - **Git** - a widely adopted version-control system for software development. https://git-scm.com/downloads
  - **Apache Maven** - a software project management and comprehension tool which manages a project's build, reporting and documentation. https://maven.apache.org/download.cgi
  - **Java 8** - a programming language expressly designed for use in the distributed environment of the Internet. https://java.com/en/download/
  - **Java IDE** - Java Integrated Development Environment (IDE) software application that enables the writing and debugging of Java programs. http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
  - **Eclipse** - an integrated development environment. It contains a base workspace for customizing the environment. https://eclipse.org/downloads/

Objectives

After completing this module, you will have:

- Demonstrated how to deploy an app to Trusted Analytics Platform (TAP) and viewed the app in a web browser.
- Identified the code that allows the app to access data from HDFS.
- Exhibited the ability to assemble an app using Maven.
- Demonstrated how to use the Cloud Foundry console to create and view a service instance.
- Learned how to access an instance of ATK and deploy apps on TAP.
- Demonstrated the ability to install multiple software tools needed to build and operate domain-specific applications.

Module Steps

1. Install all necessary developer tools listed in Setup.
2. Visit the Dataset-Reader app.
3. Review the Dataset-Reader app flow, the sample folders and the app code with Java IDE. Specifically, check how the app accesses data from HDFS and creates the graphic output.
4. Clone the Dataset-Reader app using *git clone*, compile the app using *mvn compile* and create a Java package using the *mvn package*.
5. Log into the TAP and access the App Developer tool. Set up your environment and edit the manifest.yml file.
6. Deploy the app on TAP. Deploy the app to the platform using the cf push command and confirm the app is deployed on TAP.
7. Try to view the output of Dataset-Reader in a browser. Why are you not able to view the output?
8. Update the environment and set the link to the input dataset. Use the *cf set-env* command to set the link and use *cf restage* to ensure changes take effect.
9. View the output of Dataset-Reader.

Important: The URLs and input commands provided in the workshop will change based on the instance used and the setup of your network. Check with the workshop instructor or consult with your development operations team (DevOps) to obtain the proper information.

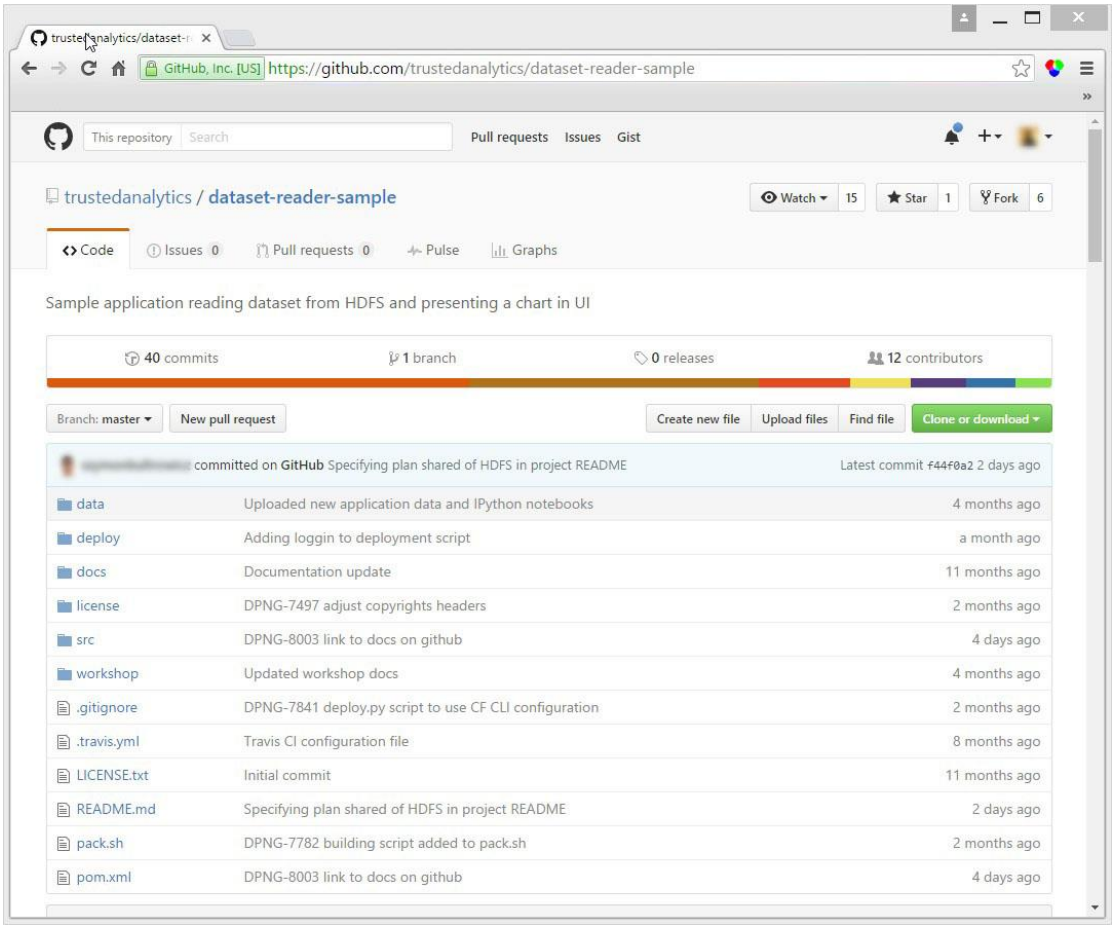
Instructions

1. Install all necessary developer tools listed in Setup

1.1. No additional instructions are provided for this step.

2. Review the Dataset-Reader app

2.1. https://github.com/trustedanalytics/dataset-reader-sample



2.2. App flow review:

- The dataset is uploaded through the Data Catalog into the platform. The file is then stored on the HDFS.
- The app developer performs analysis on the data using ATK within the TAP. The results are also stored on HDFS.
- The Dataset-Reader application is uploaded into the platform and binds to the file.
- The Dataset-Reader app presents the data graphically in a set of charts.

3. Clone, compile and create Java Package

4.1 Clone the Dataset-Reader app

```
git clone https://github.com/trustedanalytics/dataset-reader-sample.git
```

4.2 Compile the Dataset-Reader app using the command

```
mvn compile
```

This step takes a considerable amount of time to complete. Note – If you prefer to run the code on your local computer or server, you will need to provide thepath to your file location. Use this command to establish the path to your file:

```
FILE=<path to file> mvn spring-boot:run –
Dspring.profiles.active=local
```

4. Pushing to the platform

4.1 Make Java package

```
mvn package
```

4.2 Login and set proper organization and space

```
cf api <platform API address>
cf login
cf target -o <organization name> -s <space name>
```

4.3 (Optional) Change the application name and host name if necessary in the manifest.yml

```
name: <your application name>
host: <application host name>
```

 For example, if you set host to "dataset-reader" and your platform URL is "example.com", the application will be hosted under 'dataset-reader.example.com' domain.

4.4 Push dataset-reader to the platform

```
cf push
```

Application will fail to start anything because it doesn't know which file to serve and how to connect to HDFS.

4.5 Create HDFS service instance called hdfs-instance. You can do that using command line or via browser:

- Using CF CLI:

```
cf create-service hdfs shared hdfs-instance
```

- Using WebUI:
  - Go to **>Marketplace**
  - Select **>HDFS** service offering
  - Choose plan **>Shared**
  - Type the name of the instance: hdfs-instance (**Note:** the instance must be called hdfs-instance)
  - Click **>Create new instance**

4.6 Create an instance of **>Kerberos** service, named kerberos-service, using the same steps as above.

4.7 Bind the “hdfs-instance” and “kerberos-service” to application

- Using CF CLI:

```
cf bind-service dataset-reader hdfs-instance
cf bind-service dataset-reader Kerberos-service
```

- Using Web UI:
  - Go to **>Applications** list
  - Select **>details>>** of the “dataset-reader” application
  - Select the **>Bindings** tab at the top of the page
  - Click **>Bind** button next to the hdfs-instance and keberos-service (you can use filtering functionality to search for the service)

4.8 Pass the path to the file on HDFS (acquired in step Preparing data) as an environment variable called "FILE":

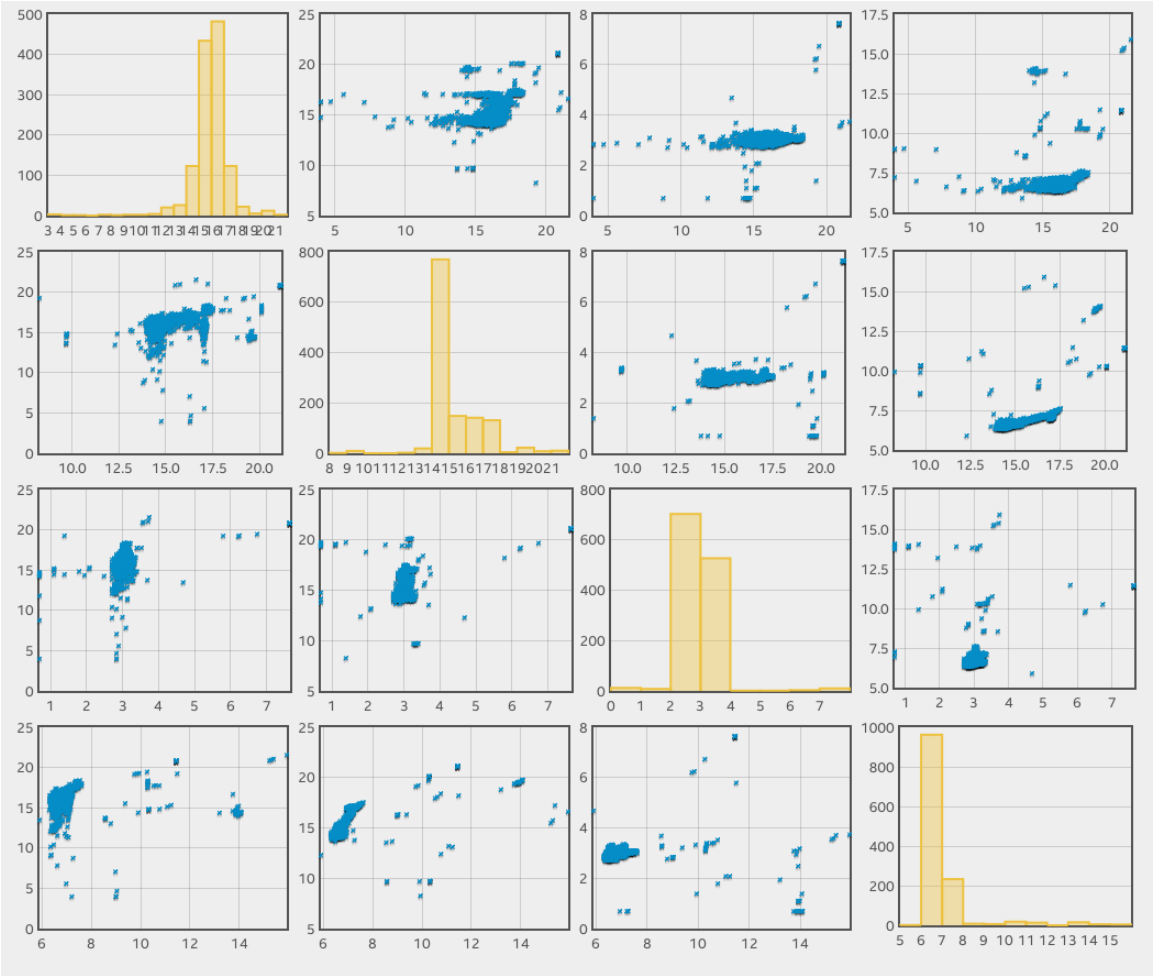
```
cf set-env <application name> FILE <path to file on HDFS>
```

4.9 Restart the application to reload the environment variables

```
cf restart <application name>
```

5. View the output of Dataset-Reader

Upon successful restage of the Dataset-Reader app, you will see the data presented in the following manner:



CONGRATULATIONS!

You have successfully completed the workshop.