FANALite User Guide

*Author: Neeraj Gupta*

*Creation Date: 30th Sep 2021*

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Changes | Notes |
| 2021-09-30 | Neeraj Gupta | Initial Document |  |

Introduction

# Summary

The FANALite project is the base project for the umbrella project name FANA (Financial Analytics Application). The aim of the Fanalite project is to start small on functionality. The main focus is on the interplay of the various applications that are required to make a solution into a product. A product that can be used by various type of users having different level of requirements. These users use different devices to interact with the system. Some of the users will use mobile, some will use their laptops or desktops and there will be users who will not be using any of these. For last type of users, their data will be collected from their emails, that data will be analyzed by our system, and will be consumed by the consultants of the users. For example, consider a case where our systems analyses the emails received by the user, our system will created General Ledger, Profit Loss Statement, Balance Sheet etc. These will be consumed by the Chartered Accountant hired by the users. This is no touch scenario for the user.

This document captures the deployment details of the project. We will see what are the various parts of the project that need to be deployed and their destinations deployment platforms.

System Overview

# Summary

Mongo Atlas

Node Server

Firebase

This section describes the components of the system which are required for a fully functional system.

# Description

Mobile

Browser

Flink

Kafka

Postman

Curl

For the Users the main focus of the system are the Mobile Applications and Browser Application. The other parts are the responsibility of the deployment team. The details of those have been captured in a separate document.

We have the following application for Users:

# Mobile Application

For Mobile Application, we currently only support the Android Mobile Application.

## Android Mobile Application

The Mobile Application can be installed from Android Studio. The details are captured in the deployment document for the same.

## iOS Mobile Applications

The Mobile Application for iOS systems is not yet supported.

# Browser Applications

## End User Browser Application

This is the application used by regular users to interact with the system. The details of usage are provided in the section dedicated for the same

## System User Browser Application

This is the application used by system users to interact with the system. This provides more access to the system. Also the fuctionality of the Distributed Scaled Backend can be tested. The details of usage are provided in the section dedicated for the same

Android Mobile Application

Summary

This section provides details for using the Mobile Application. We assume that the Mobile Application is installed on your Android phone.

Requirements

1. Android Mobile phone
2. Ensure that the Fanalite Realtime Rest Server is running.
3. Mobile application should be installed on your mobile

The instructions for the same are provided in the ‘Fanalite Deployment Guide’.

Start

Click on the Application Icon or search for Application named ‘**Insights’**

Graphical user interface, application

Description automatically generated

# Rules List

The Rules List is the default screen

Graphical user interface, application

Description automatically generated with medium confidence

# Add Rule

Click on the Fab Button on the Rules List Screen to add a rules

Graphical user interface, application

Description automatically generated with medium confidenceGraphical user interface, text, application, chat or text message

Description automatically generated

Tap Fab Button

Steps:

Tap on the Add Rule Button

Fill the Values

Tap on Save

# Modify Rule

Click on a Rule you want to Modify on the Rules List Screen.

Graphical user interface, application

Description automatically generated with medium confidence Graphical user interface, text, application, chat or text message

Description automatically generated

Tap Rule

Steps:

Tap on the Rule

Modify the Values

Tap on Save

# Delete Rule

Click on a Rule you want to Modify on the Rules List Screen.

Graphical user interface, application

Description automatically generated with medium confidence

Tap Delete

Steps:

Tap Delete Button on the Rule

End User Browser Application

Summary

This section provides details for using the End User Browser Application.

Requirements

1. System with a Browser application.
2. Ensure that the Fanalite Realtime Rest Server is running.
3. Ensure that the Fanalite Frontend Application Server is running
4. The Realtime Rest Server and Frontend Application Server are reachable from the system where you are going to run your Browser application.

The instructions for the same are provided in the ‘End User Application’ in the ‘Fanalite Deployment Guide’.

Start

Open your browser

http://localhost:3000/

# Rules List

The Rules List is the default screen

Graphical user interface, application

Description automatically generated

# 

# Add Rule

Add the title in the Top input box. This ‘ADD REGEX’ button will be enabled.

Add the Regular Expression the lower input box

And Click ‘ADD REGEX’ button

Graphical user interface, application, Teams

Description automatically generated

# Modify Rule

Click the ‘EDIT BUTTON’ next to the rule you want to modify. Refer figure below.

Modify the values

# Graphical user interface, application Description automatically generated

Click Edit

Modify the values

Click the ‘SAVE’ button.

Graphical user interface, application, Teams

Description automatically generated

# Delete Rule

Click the ‘DELETE BUTTON’ next to the rule you want to modify. Refer figure below.

Modify the values

# Graphical user interface, application Description automatically generated

Click Delete

System User Browser Application

Summary

This section provides details for using the System User Browser Application. This application gives us direct access to the API server. The API server also has a web client support. This web client does not focus on being user friendly but it provides access to more features of the API Server.

Requirements

1. System with a Browser application.
2. Ensure that the Fanalite Realtime Rest Server is running.
3. Ensure the Apache Kafka messages queues is running.
4. Ensure the Apache Flink stream processing is running.
5. The Realtime Rest Server is reachable from the system where you are going to run your Browser application.

The instructions for the above steps are provided in the ‘System User Application’ in the ‘Fanalite Deployment Guide’.

Start

Open your browser

<http://localhost:3030/>

# Start Screen

## Create User

Fill input boxes as shown in figure and click ‘Sign up and log in’

[neeraj@abc.com](mailto:neeraj@abc.com)

Neeraj123

## Login User

Fill input boxes as shown in figure and click ‘Log in’

[neeraj@abc.com](mailto:neeraj@abc.com)

Neeraj123

Graphical user interface, application

Description automatically generated

# Dashboard Screen

Graphical user interface, text, application

Description automatically generated

## Stream Processing Commands

We can enter the stream processing commands in the System User Browser Application to configure our stream processor.

The format of the command is given below:

<UserId>, <Command>, <StreamName>, <StreamText>

UserId: String

Command: Choices [Rule|Text]

StreamName: String

StreamText: String

Given below are the example commands:

1, Rule, CheckNumber, .\*(?<Date>\d{2}/\d{2}/\d{4}).\*(?<Number>\d+).\*

1, Text, Payment, Dated 10/09/2020 and Rs 455 complete

## Rule Command

<UserId>, Rule, < RuleName>, <RuleText>

Output

**<SystemUser>** <Timestamp>

New Rule:<RuleName>, <RuleText>, {}

## Text Command with Matching Rule

<UserId>, Text, <StreamName>, <Text>

Output in Case Match

**<SystemUser>** <Timestamp>

<RuleName>, <Text>, <JSON>

## Text Command with No Matching Rule

<UserId>, Text, <StreamName>, <Text>

Output in Case Match

**<SystemUser>** <Timestamp>

No Match, <Text>, {}

# Add Stream Processing Rule

1, Rule, CheckNumber, .\*(?<Date>\d{2}/\d{2}/\d{4}).\*(?<Number>\d+).\*

We can see that the system has received and processed our command and we get the following result

**system@abc.com** Sep 30th, 12:48:54

New Rule:CheckNumber, .\*(?<Date>\d{2}/\d{2}/\d{4}).\*(?<Number>\d+).\*, {}

Graphical user interface, application, Teams

Description automatically generated

This signifies that a rule to identify future text has been added.

It will become clear shortly.

# Send Matching Stream Message

After the rule we will add the Text message

1, Text, Payment, Dated 10/09/2020 and Rs 455 complete

We can see that the system has identified the message using the rule specified earlier

**system@abc.com** Sep 30th, 12:51:57

CheckNumber, Dated 10/09/2020 and Rs 455 complete, {Number=5, Date=10/09/2020}

The Number and Date field in the json are corresponding the group names specified in the previously added rule.

Graphical user interface, text, application

Description automatically generated

# Send Unmatching Stream Message

Now we will add a text message which matches no rule

1, Text, Payment, Rs 455 and Dated 10/09/2020

We get the following message from the system

**system@abc.com** Sep 30th, 01:01:27

No Match:Payment, Rs 455 and Dated 10/09/2020, {}

Graphical user interface, text, application, email

Description automatically generated

Conclusion

This is a very basic application which supports CRUD (Create, Retrieve, Update Delete) for Rules. The Guide shows how to achieve these operations on the multiple User applications to interact with the system.

We have covered the Android Mobile Application. The iOS Mobile Application is not supported yet.

We have covered the End User Browser Application and the System User Browser Application.

LESSONS

The document should be written while we are creating the system. This saves us time in terms of recreating the environment etc. Also the context switch is not much. The downside could be that we might have to change the document if the feature changes.