Credit Suisse Case Study Starter Project

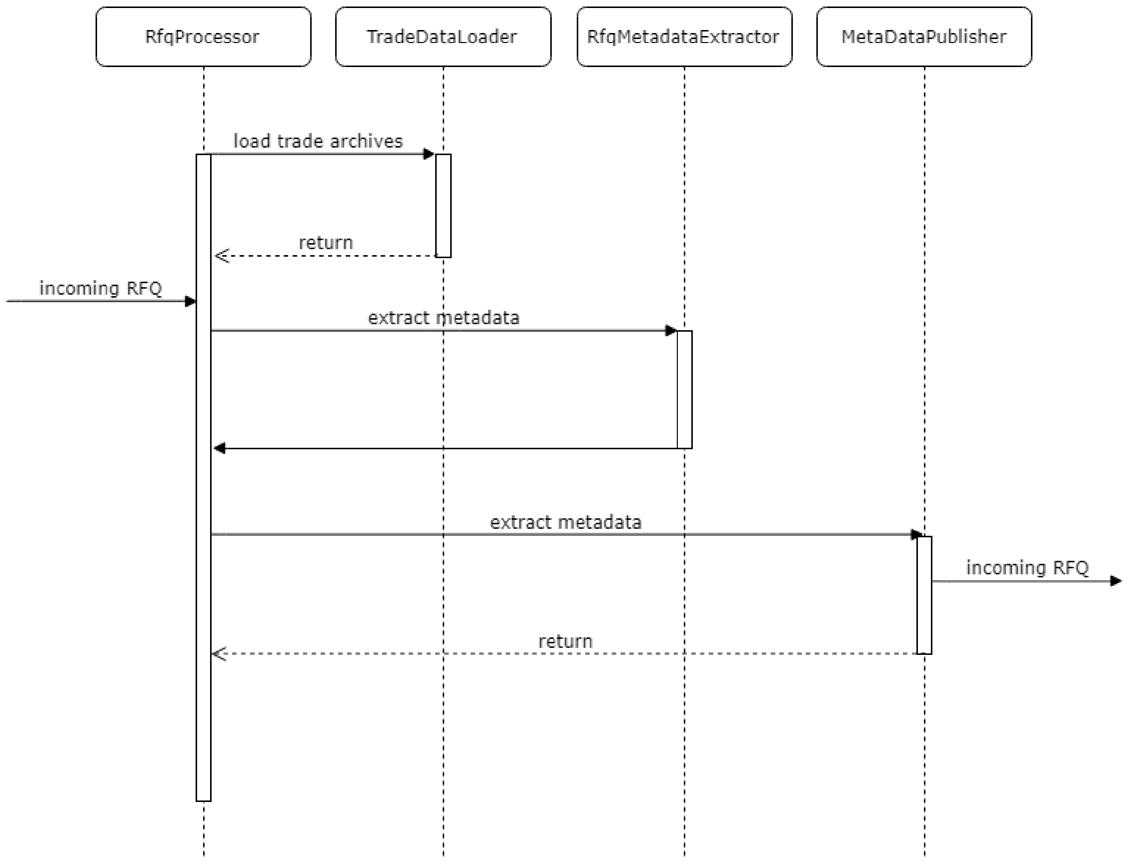
This starter project has been provided to get you going with the case study project. It's entirely your choice whether you use it or not. You are free to start from a blank project if you prefer.

Download and Build the Sources

Clone the starter project from the Git repository your instructor has provided.

From IntelliJ, open the project by opening the file pom.xml. Select **Build -> Build** project from the menus. IntelliJ may take a while to download and index any missing dependencies before compiling the sources.

Take a few minutes to examine the code. The key interaction between the main classes/interfaces in the project is shown in this sequence diagram:



The RFQ processor will load the trade archive data into Spark RDDs, it will then listen for incoming messages using a Spark stream listener. When it receives an RFQ, it converts it into a plain old java object (POJO) before passing it on to any number of extractors, which are responsible for building the metadata. Finally, all the metadata is passed to a publisher for forwarding to downstream components.

Complete the RFQ Class

Run the following unit test, and make the required changes to get it working:

```
\src\test\java\com\cs\rfq\decorator\RfqTest.java
```

Hint: The following code will convert a JSON literal map to a Java collection map:

```
String json = ...;
Type t = new TypeToken<Map<String, String>>() {}.getType();
Map<String, String> fields = new Gson().fromJson(json, t);
```

Complete the TradeDataLoader Class

Run the following unit test, and make the required changes to get it working:

```
\src\test\java\com\cs\rfq\decorator\TradeDataLoaderTest.java
```

Complete the Server Code

Open the following files and complete the sections marked TODO:

```
\src\main\java\com\cs\rfq\decorator\RfqDecoratorMain.java
\src\main\java\com\cs\rfq\decorator\RfqProcessor.java
```

Run the RfqDecorator Server

Start by running our feed simulation server program

Right-click this class and select the **Run** option from the drop-down menus.

```
\src\main\java\com\cs\rfq\utils\ChatterboxServer.java
```

The command line shown by this program allows you to send single lines of text to any process that connects on port 9000. You should have configured your RfqDecorator program to listen on this port.

Now you can start the RfqDecorator server

Right-click this class and select the **Debug** option from the drop-down menus.

```
\src\main\java\com\cs\rfq\decorator\RfqDecoratorMain.java
```

Send a test RFQ

Paste a single line of JSON into the Chatterbox console and press **ENTER**. This will send the RFQ JSON to the Spark process which should process it and generate a JSON message with the required metadata fields.

Back to the Case Study

This starter should meet many of the objectives of the case study, but there are many more to complete. Make sure you understand which objectives have been met with this starter project, and take a moment to prioritize any remaining tasks before proceeding.