Sales-force Streaming Events and Kafka Integration

# **Document History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Created By** | **Reviewed By** |
| 2021-11-24 | V-0.1 | Saurabh Dadhich | Kamlesh Kumar Pant |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Contents**

[**Solution Architecture** 3](#_Toc88497729)

[**Solution Design/Approach** 3](#_Toc88497730)

[**Setting up Salesforce for Streaming Events:** 4](#_Toc88497731)

[ Create Connected App: 4](#_Toc88497732)

[ Acquire Consumer Key and Secret Key: 6](#_Toc88497733)

[ Acquire refresh token: 7](#_Toc88497734)

[ Enable Salesforce for CDC: 9](#_Toc88497735)

[ Enable Salesforce for PushTopic: 9](#_Toc88497736)

[**Application Development:** 11](#_Toc88497737)

[ Software Requirements: 11](#_Toc88497738)

[ Installation Details: 12](#_Toc88497739)

[ Start Gitbash 13](#_Toc88497740)

[ Start Zookeeper 14](#_Toc88497741)

[ Start Kafka Server 15](#_Toc88497742)

[ Create Topics 15](#_Toc88497743)

[ Application Configuration Details: 16](#_Toc88497744)

[ Application Development Details: 21](#_Toc88497745)

[**Application Deployment and Testing:** 21](#_Toc88497746)

SALESFORCE - CDC & KAFKA CONNECTOR

This solution is primarily intended for Citibank, where Citi bank requires all the changes related to users -any addition/modification or role changes etc should be reflected in real time to consuming applications.

To Achieve this, application capitalizing salesforce event bus mechanism to capture the CDC Data.

# **Solution Architecture**

Diagram

Description automatically generated

# **Solution Design/Approach**

To capture CDC data and storing or redirecting them to HDFS/Databases (Mysql) or Kafka Topics, application (Python) needs to capitalize Salesforce Streaming Events service for capturing real time changing data via SF event bus.

Following steps to be executed across platforms

# **Setting up Salesforce for Streaming Events:**

## Create Connected App:

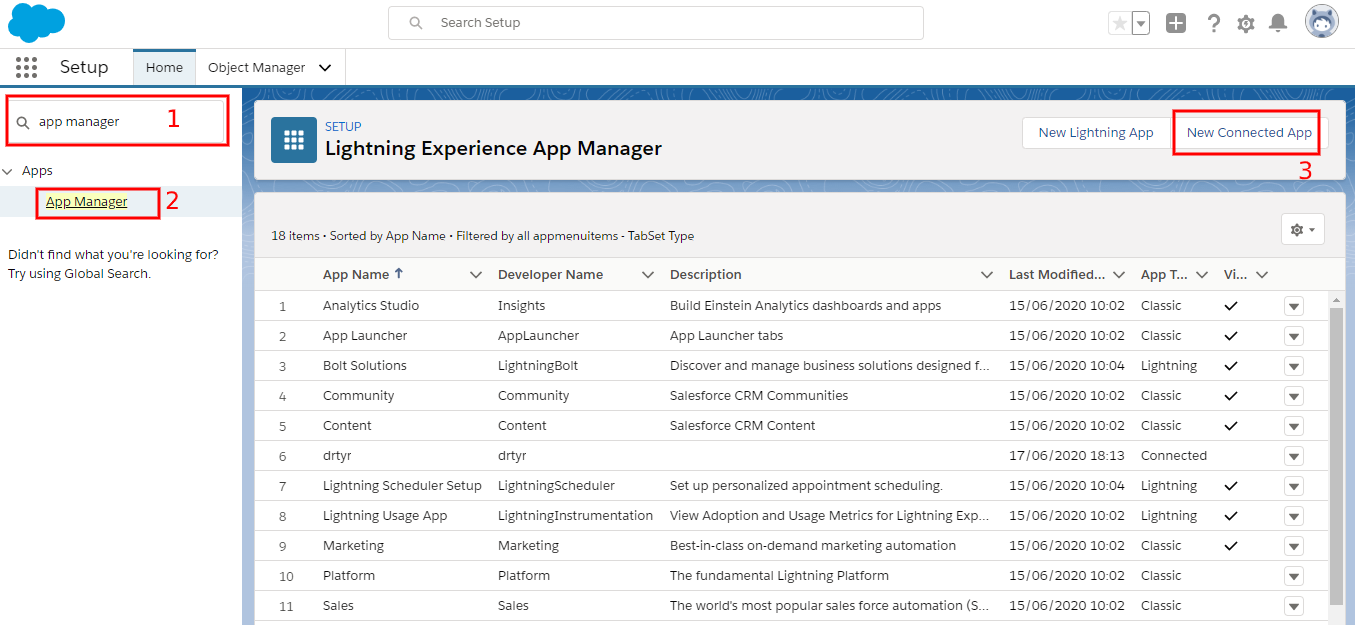
- To create a connected Salesforce app, follow these steps:

- Log in to Salesforce with your developer account

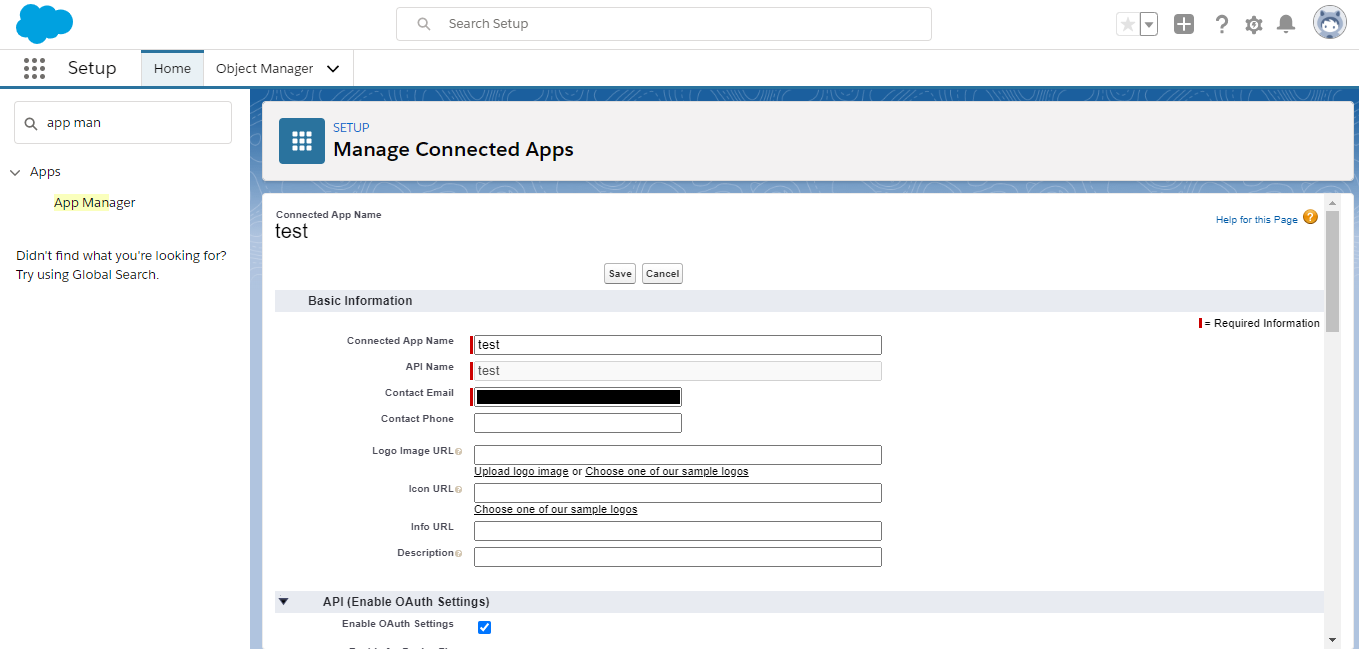
- In the drop-down list of the account (in the upper-right corner), select **Setup**

- In the left-hand pane, search for: **App Manager** and open it

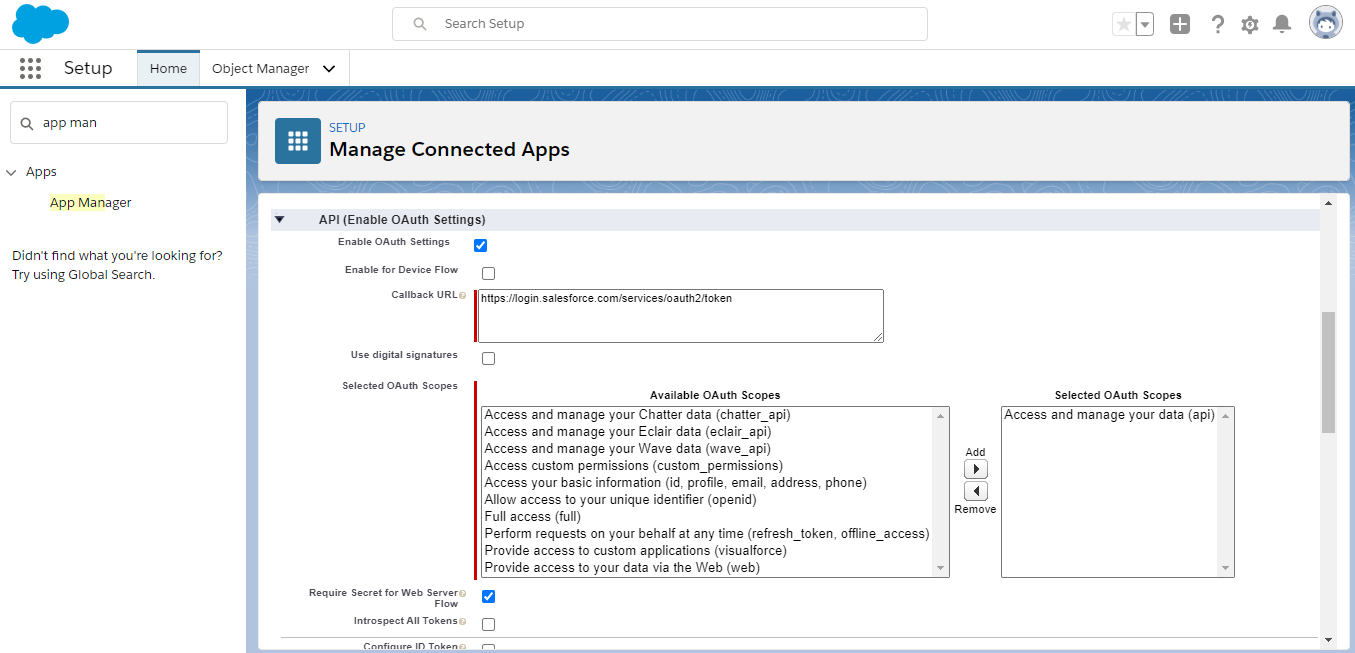
- Click on **“New connected App”**

[](https://soft-builder.com/wp-content/uploads/2020/06/screenshot-3.png)

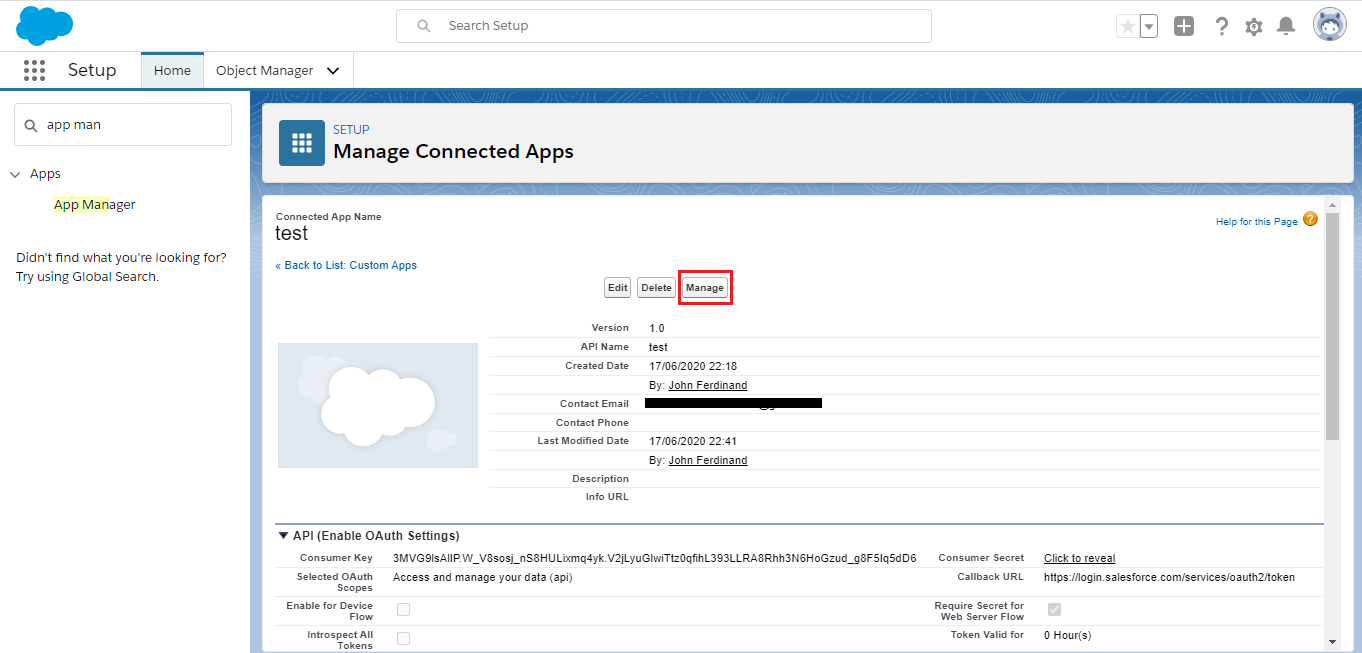
* On the New Connected App page, fill the following required fields under Basic Information:
* **Connected App Name.** For example, "test". The connected app name must be unique within your org. If the connected app was created in the Spring ‘14 release or later, you can reuse the name of a deleted connected app. Only letters, numbers, and underscores are allowed, also avoid using names which ends with "builder".
* **API name.** For example, "test".
* **Contact Email.** your email.

[](https://soft-builder.com/wp-content/uploads/2020/06/screenshot4.png)

* Go to **API (Enable OAuth Settings)** and select Enable OAuth Settings.
* In the **Call back URL** field, enter **https://login.salesforce.com/services/oauth2/token** for a production Org and **https://test.salesforce.com/services/oauth2/token** for a Sandbox Org.  
  If you have a custom domain enabled on your org, you must use the custom URL [**https://yourcustomdomain.my.salesforce.com/services/oauth2/token**](https://yourcustomdomain.my.salesforce.com/services/oauth2/token)
* In the **Selected OAuth Scopes** field, select **Access and manage your data (api)**, and then click **Add**.
* Click the **Save**button to save the new Connected App.

[](https://soft-builder.com/wp-content/uploads/2020/06/screenshot5.png)

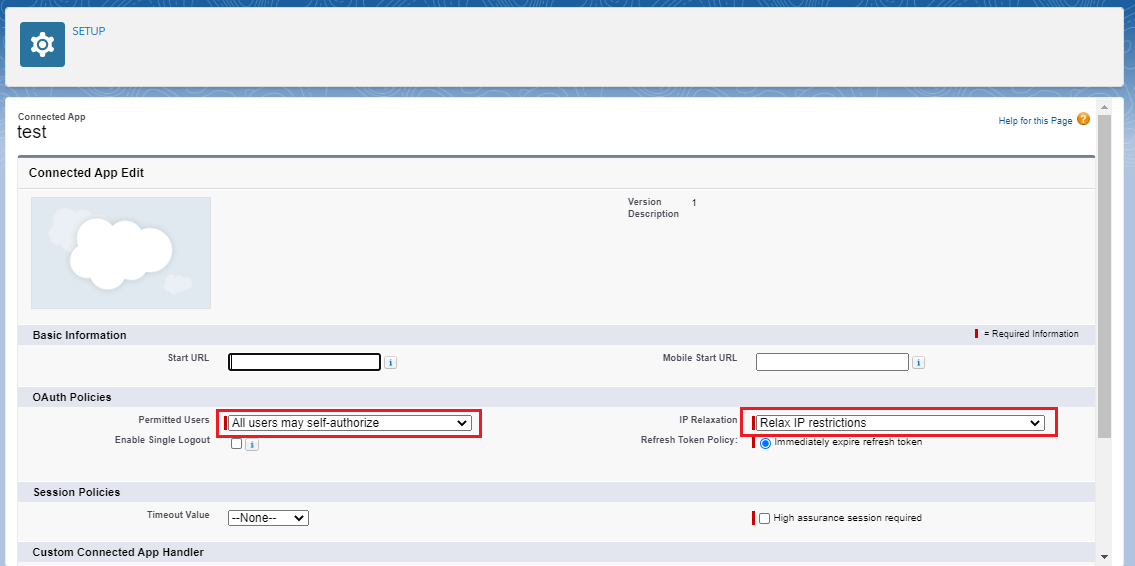
* On the new app that you just created, click **Manage**

[](https://soft-builder.com/wp-content/uploads/2020/06/screenshot6.png)

- On the page that opens, click the Edit Policies button.

- Set**IP Relaxation** to: Relax IP Restrictions

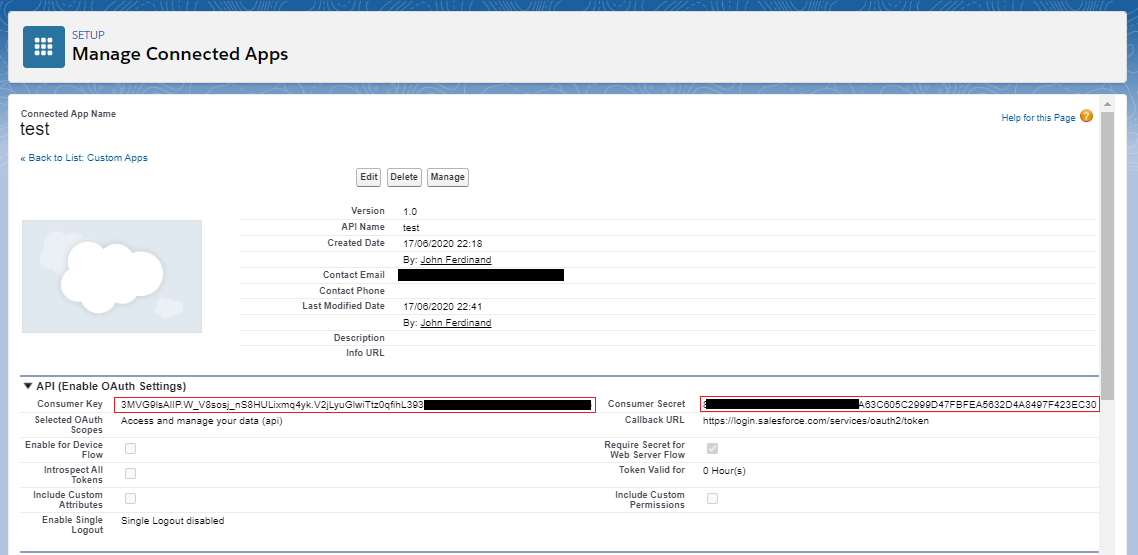
- Set **Permitted Users** to: All users may self-authorize

[](https://soft-builder.com/wp-content/uploads/2020/06/screenshot7.png)

## 

## Acquire Consumer Key and Secret Key:

* Go to the **API (Enable OAuth Settings)** section, and note down the **Consumer Key** and **Consumer Secret**.

[](https://soft-builder.com/wp-content/uploads/2020/06/screenshot8-1.png)

* Get Your Consumer Key and Client Secrete and Save it in Notepad.

## Acquire refresh token:

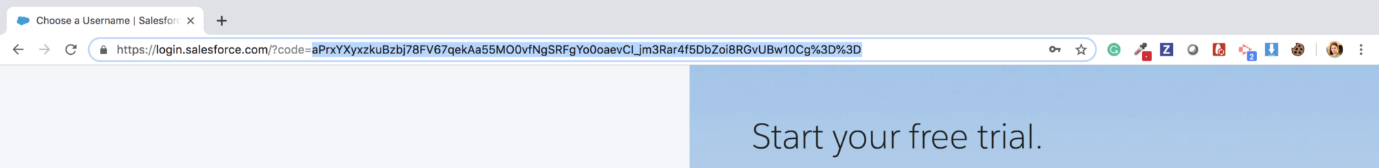
* Log in to Salesforce using your favourite browser, then enter the following request Url in a new tab to get the code. <CONSUMER\_KEY> should be replaced with the obtained Consumer Key in the above step. <YOUR\_INSTANCE> should be replaced with your instance name, in my case it is **ap15**.

https://**<YOUR\_INSTANCE>**.salesforce.com/services/oauth2/authorize?response\_type=code&client\_id=**<CONSUMER\_KEY>**&redirect\_uri=https://login.salesforce.com/



* Enter request Url in the browser
* Allow access if any alert popup. Then you will see browser is redirecting to a Url like this. You can obtain the code using that Url.

[https://login.salesforce.com/?code=**aPrxYXyxzkuBzbDGdwv67qekAQredtrsWqty38LsdhfREyTRbvdjvTqdbvxPVC\_\_4Cb9xGKDGErtOw%3D%3D**](https://login.salesforce.com/?code=aPrxYXyxzkuBzbj78FV67qekAQpAHyzh9Ry38LsyinKxPVC__Wzov6j6OBFx74Cb9xGKI60AOw%3D%3D)



- The browser will redirect to a Url with the code

- Get Access token & Refresh token

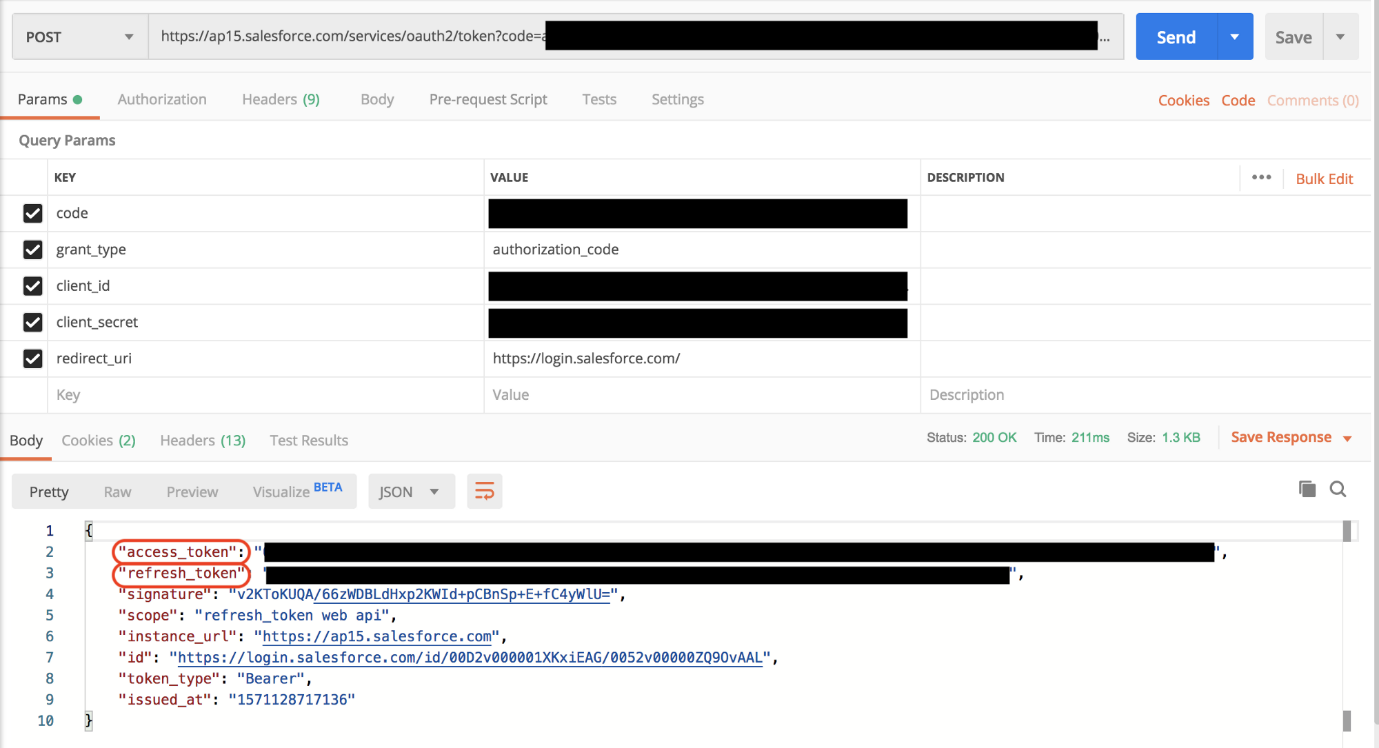
- Send the following curl request to obtain the tokens. <CODE> should be replaced with the code you obtained in the above step. <CONSUMER\_KEY> and <CONSUMER\_SECRET> should be replaced with obtained keys with the created Connected App.

curl -X POST https://**<YOUR\_INSTANCE>**.salesforce.com/services/oauth2/token?code=**<CODE>**&grant\_type=authorization\_code&client\_id=**<CONSUMER\_KEY>**&client\_secret=**<CONSUMER\_SECRET>**&redirect\_uri=https://login.salesforce.com/

* You can obtain the **access token** and **refresh token**from the response.

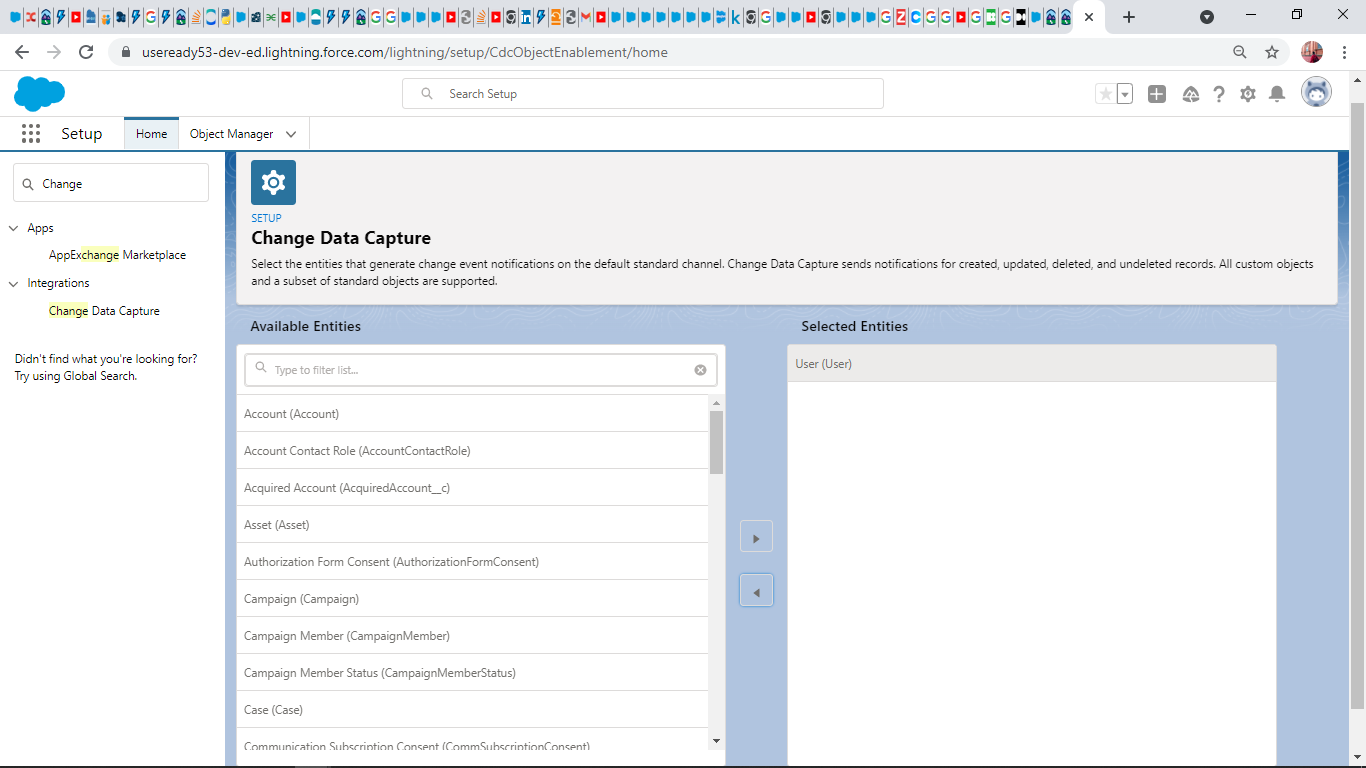
{  
**"access\_token":"00D2v000001XKxi\_\_SOMETHING"**,  
**"refresh\_token":"5Aep861dlMxAL.LhVTuPRa\_\_SOMETHING"**,  
"signature":"MK/YGMNQhPSSnKtYicXlaU\_\_SOMETHING",  
"scope":"refresh\_token web api",  
"instance\_url":"[https://ap15.salesforce.com](https://ap15.salesforce.com/)",  
"id":"[https://login.salesforce.com/id/00D2vKxiEAG/0045Q09vAAL](https://login.salesforce.com/id/00D2v000001XKxiEAG/0052v00000ZQ9OvAAL)",  
"token\_type":"Bearer",  
"issued\_at":"1570030000198"  
}

* If you are not familiar with curl you can use Postman to send the request.



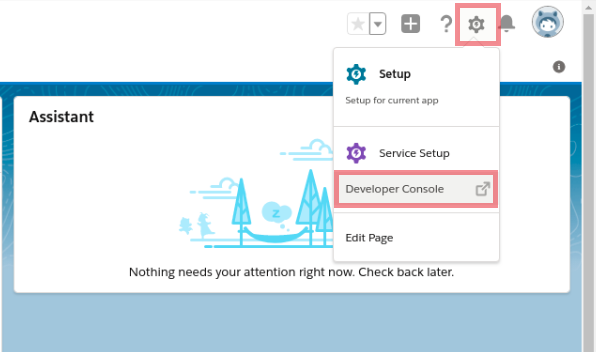
## Enable Salesforce for CDC:

* Go in your Salesforce Dev Org. Click on set up.
* Then search Change Data Capture in Quick Find Box.
* Click on change data capture and You will find two boxes in one box all salesforce objects are there and one box is empty.
* So, select the object which changed data event you want.
* In Our case We are selecting Users.

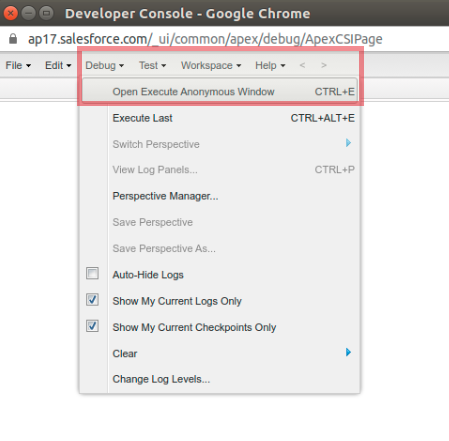


## Enable Salesforce for PushTopic:

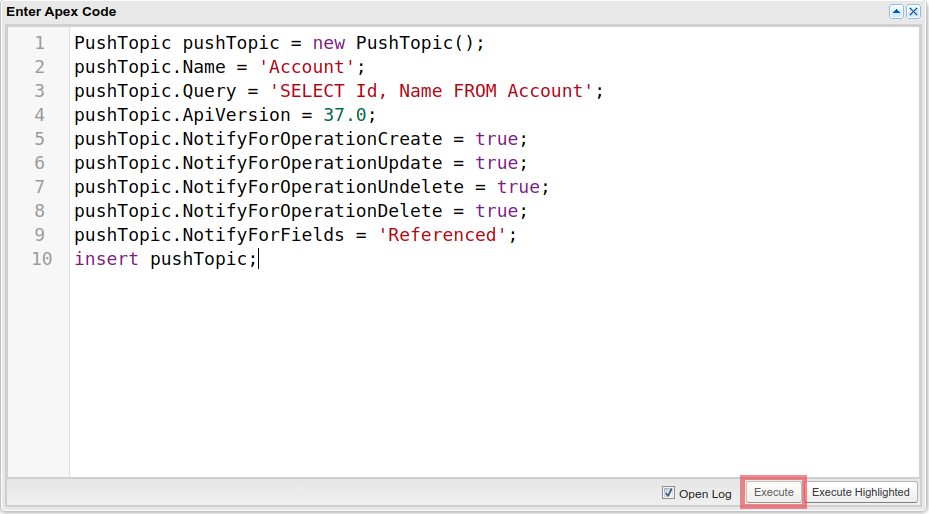
* The pushTopic record contains a SOQL query. Event notifications are generated for updates that match the query. Alternatively, you can also use Workbench to create a PushTopic. In this sample we using Salesforce Developer Console to create a Push Topic.
* **Login** to the **Salesforce Account**. Navigate to the top right corner of the **Home page** and click the **Setup** icon. Then select **Developer Console**.



* After populating the Developer console, click **Debug** -> Open **Execute Anonymous Window**.



* Add the following entry in the **Enter Apex Code** window and click **Execute**.



PushTopic pushTopic = **new** PushTopic();

pushTopic.Name = 'Account';

pushTopic.Query = 'SELECT Id, Name FROM Account';

pushTopic.ApiVersion = 37.0;

pushTopic.NotifyForOperationCreate = true;

pushTopic.NotifyForOperationUpdate = true;

pushTopic.NotifyForOperationUndelete = true;

pushTopic.NotifyForOperationDelete = true;

pushTopic.NotifyForFields = 'Referenced';

insert pushTopic;

* We are essentially creating a SOQL query with a few extra parameters that watch for changes in a specified object. If the Push Topic is executed successfully then Salesforce is ready to post notification to WSO2 Salesforce Inbound Endpoint, if any changes are made in the Account object in Salesforce. This is because the below Push Topic has been created for Salesforce's Account object.

# **Application Development:**

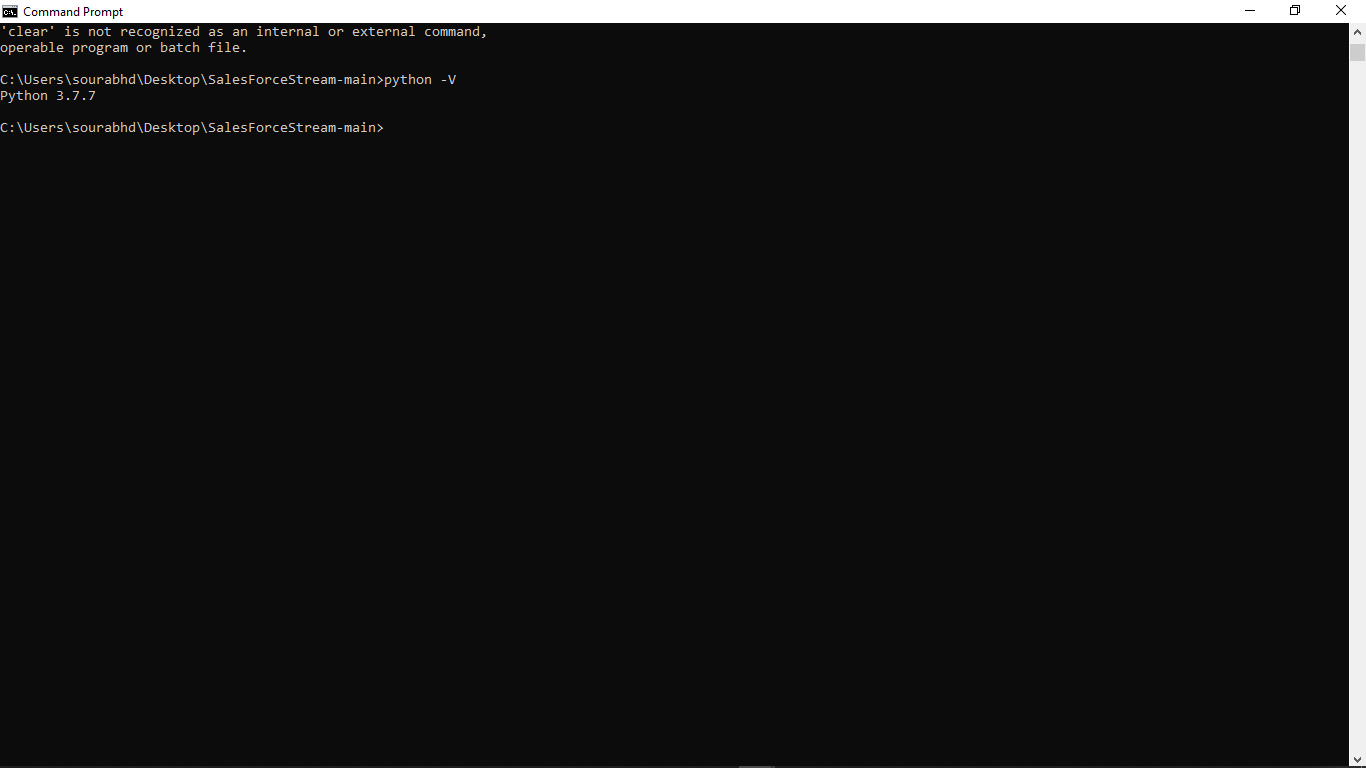
## Software Requirements:

|  |  |  |
| --- | --- | --- |
| **Requirements** | **Version** | **Libraries** |
| Python | 3.7 | Pyodbc,  Requests,  Pytz,  kafka-python,  mysql-connector,  aiosfstream |
| Git hub | **-** |  |
| java | jdk 8 |  |
| Kafka | 2.13-2.8 |  |
| PostmanAPI | **-** |  |
| Salesforce account | Developer account |  |
|  |  |  |
|  |  |  |

## Installation Details:

Step 1. Install Python and Required Libraries:

* Consider this process in Windows
* Firstly, install python 3.7 and install all required libraries which mentioned in the starting of this document.

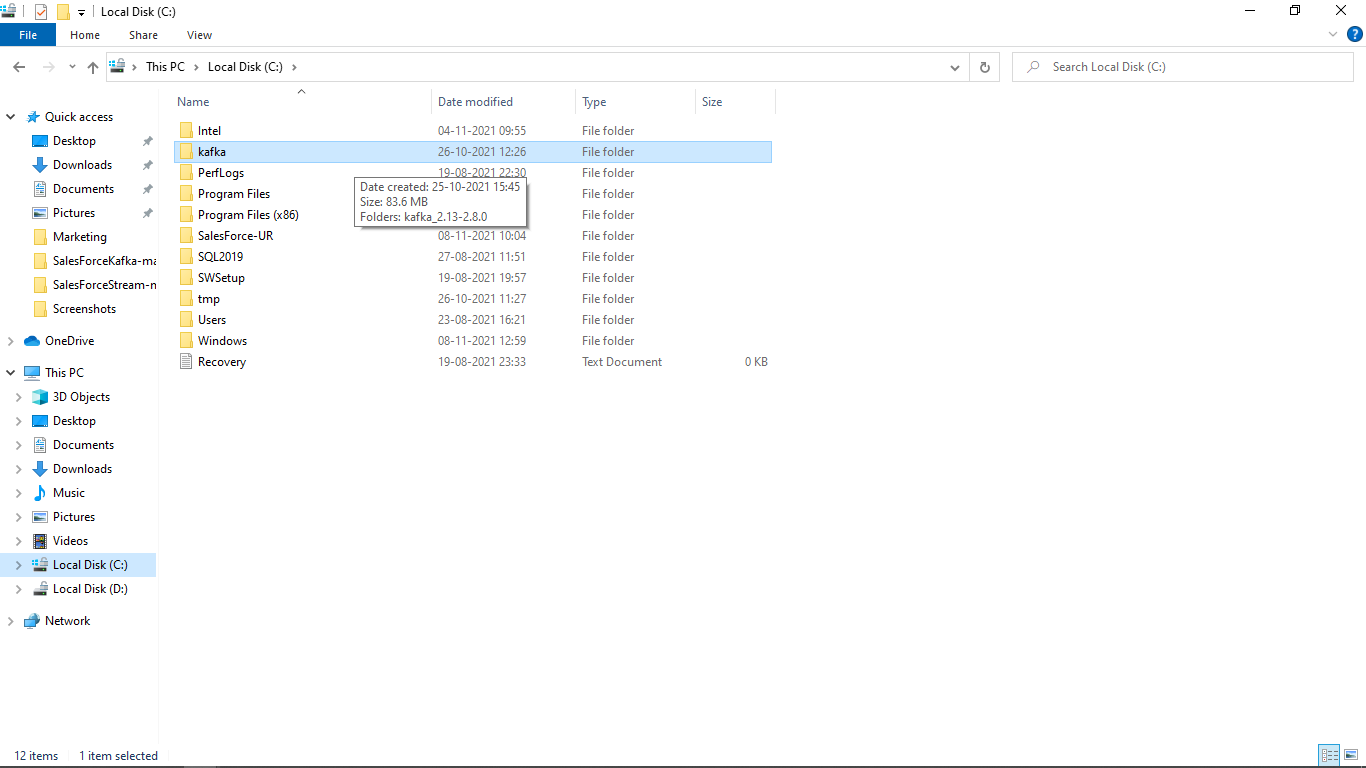


Step 2. Install git hub and git bash in your system.

* Install git 2.33.1 in your system
* See its all command

Step 3. Install Kafka

* Then download kafka 2.13-2.8.0 tar file in your system.
* Move that tar file into C drive and Unzip into it.
* Since it’s based on JVM languages like Scala and Java, you must make sure that you are using Java 7 or greater.

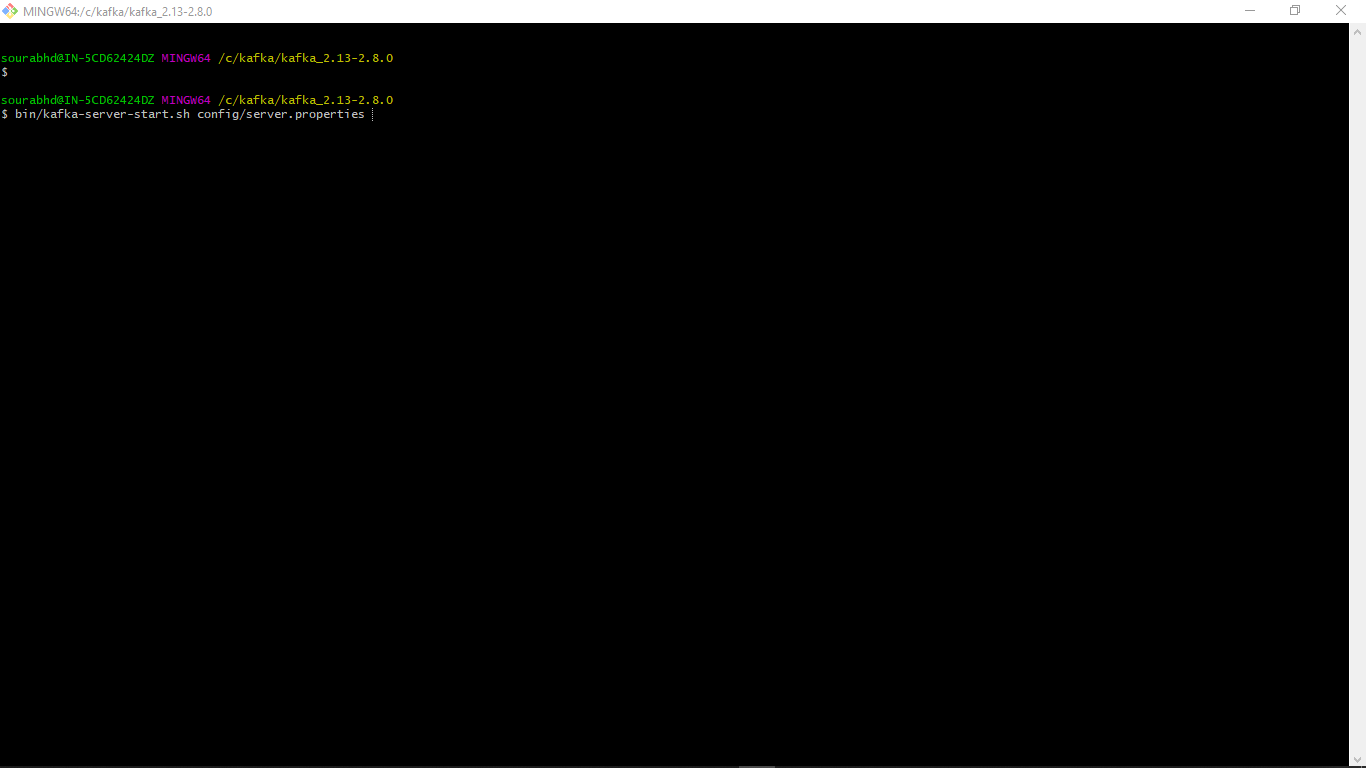


After installing kafka Follow these steps to start kafka server:

## Start Gitbash

* Go in Kafka Folder and Right click on any white space. Then click on git bash.
* Minimise First git bash and again follow the first step to open one more git bash.
* Note: check previous git bash and cmd prompt aren’t open at the same time.





The easiest way to install Kafka is to download binaries and run it. Since it’s based on JVM languages like Scala and Java, you must make sure that you are using Java 7 or greater.

Kafka is available in two different flavours: One by [**Apache foundation**](https://kafka.apache.org/downloads) and other by [**Confluent**](https://www.confluent.io/about/) as a [**package**](https://www.confluent.io/download/). For this tutorial, I will go with the one provided by Apache foundation. By the way, Confluent was founded by the [**original developers**](https://www.confluent.io/about/) of Kafka.

## Start Zookeeper

Kafka relies on Zookeeper, in order to make it run we will have to run Zookeeper first. Write this Command in First git bash.

bin/zookeeper-server-start.sh config/zookeeper.properties

it will display lots of text on the screen, if see the following it means it’s up properly.

2018-06-10 06:36:15,023] INFO maxSessionTimeout set to -1 (org.apache.zookeeper.server.ZooKeeperServer)  
[2018-06-10 06:36:15,044] INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apache.zookeeper.server.NIOServerCnxnFactory)

## Start Kafka Server

Next, we have to start Kafka broker server in the second git bash:

bin/kafka-server-start.sh config/server.properties

And if you see the following text on the console, it means it’s up.

2018-06-10 06:38:44,477] INFO Kafka commitId: fdcf75ea326b8e07 (org.apache.kafka.common.utils.AppInfoParser)  
[2018-06-10 06:38:44,478] INFO [KafkaServer id=0] started (kafka.server.KafkaServer)

## Create Topics

Messages are published in topics. Use this command to create a new topic.

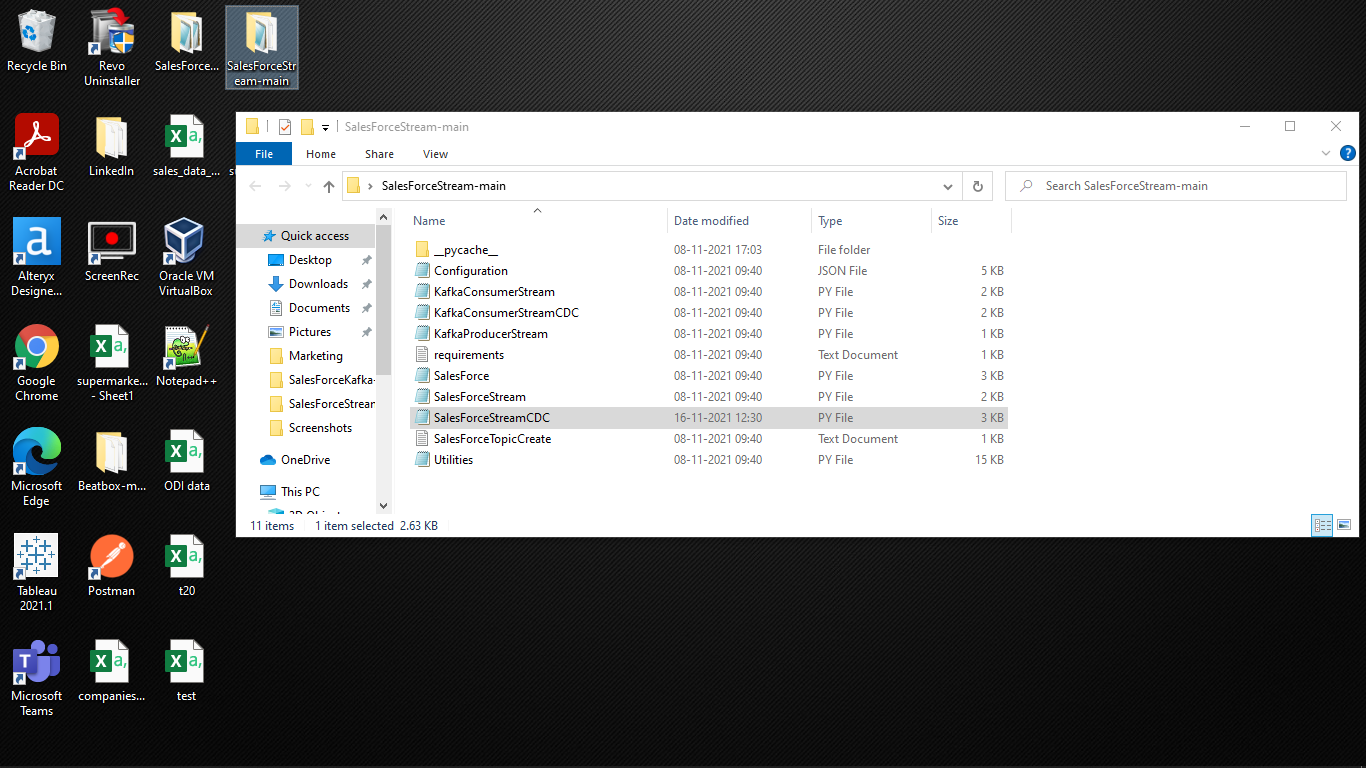
➜ kafka\_2.11-1.1.0 bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test  
Created topic "test".

You can also list all available topics by running the following command.

➜ kafka\_2.11-1.1.0 bin/kafka-topics.sh --list --zookeeper localhost:2181  
test

Step 4. Clone or Download the Python Code.

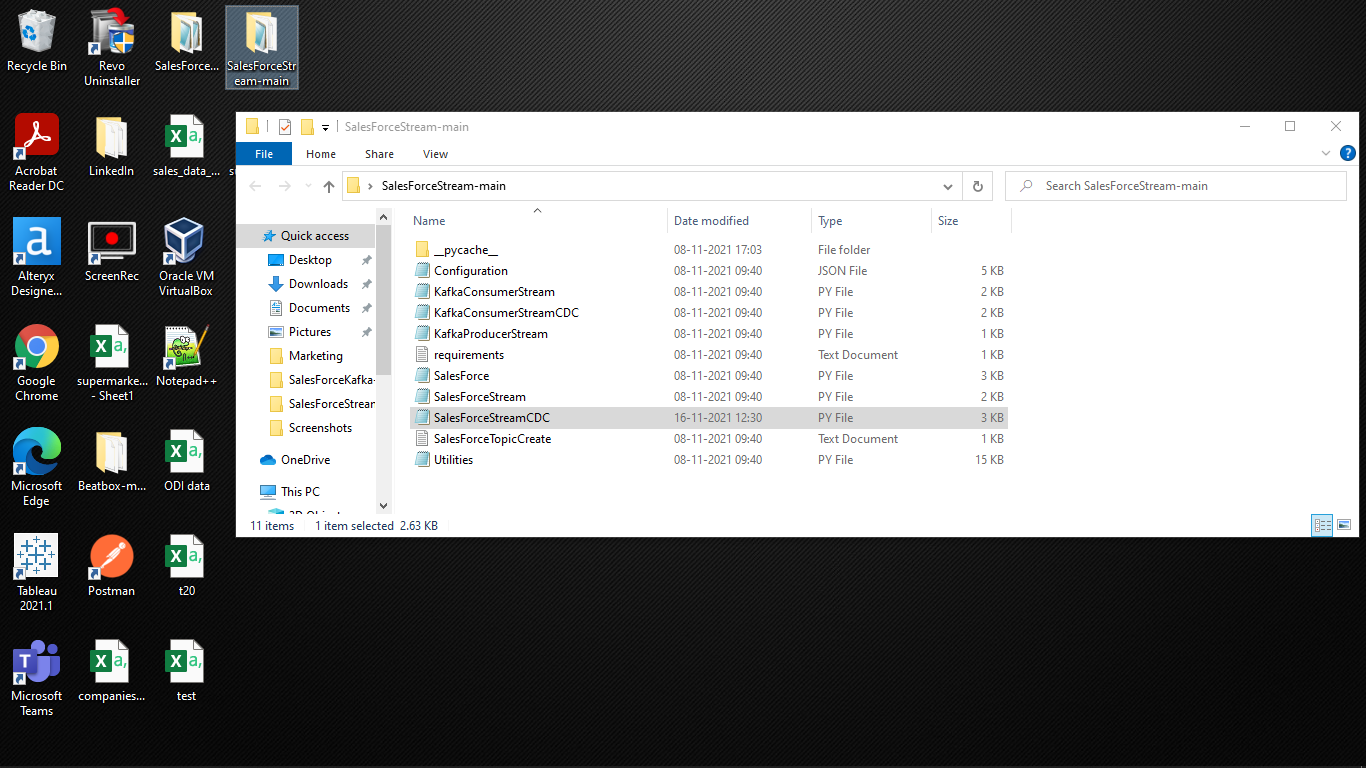
* Clone or download the python codes from git hub repo
* Unzip that file on desktop



## Application Configuration Details:

Clone or Download the Python Code.

* Clone or download the python codes from git hub repo
* Unzip that file on desktop



* Set Up Your Configuration in Configuration File:
* Go to the file where all python codes are there.
* Look for a configuration file where all configurations are pre-written in json format.
* You just need to change the configuration as per your requirement.
* Now, Let’s Look into the config file.

**Configuration file (json code):**

|  |
| --- |
| {  "DATABASE": {  "MSSQL": [  {  "SERVER": "mssqlldb",  "NAME": "**database\_name**",  "SCHEMA": "DBO",  "TABLE\_PREFIX": "**table\_prefix”**  "DRIVER": "{SQL Server}",  "AUTH\_TYPE": "WINLOGIN",  "USERNAME": "**your\_username**",  "PASSWORD": "**your\_pwd**"  }  ],  "MYSQL": [  {  "SERVER": "127.0.0.1”, - **Localhost sever**  "NAME": "XYZ”, - **Database Name**  "SCHEMA": "“, - **Schema Name**  "TABLE\_PREFIX": "XYZ”, - **Table Prefix**  "DRIVER": "", - **Driver name**  "AUTH\_TYPE": "WINLOGIN", - **Authentication Type**  "USERNAME": "XXXX", - **your Username**  "PASSWORD": "XXXX" - **your password**  }  ],  "SNOWFLAKE": [  {  "WAREHOUSE": "PC\_ALTERYX\_WH",  "DSN": "SnowFlake32",  "CONNECTION\_URI": "pua75685.snowflakecomputing.com",  "NAME": "**database\_name**",  "SCHEMA": "**schema\_name**",  "TABLE\_NAME": "**table\_name**",  "STAGE\_NAME": "**Your\_stage\_name**",  "FILE\_FORMAT\_NAME": "**Filene name**",  "FILE\_FORMAT\_TYPE": "CSV",  "FILE\_FIELD\_DELIMITER": ",",  "SNOWSQL\_CMD": "C:\\\"Program Files\"\\\"Snowflake SnowSQL\"\\snowsql",  "TABLE\_PREFIX": "Snowflake\_Table\_",  "DRIVER": "SnowflakeDSIIDriver",  "USERNAME": "**Your\_Username**",  "PASSWORD": "**your\_pwd**"  },  {  "WAREHOUSE": "COMPUTE\_WH",  "DSN": "SnowFlake32",  "CONNECTION\_URI": "mpa43641.snowflakecomputing.com",  "NAME": "SNOWDBKK",  "SCHEMA": "SNOWSCHEMAKK",  "TABLE\_NAME": "SNOWTABLEKK",  "STAGE\_NAME": "PY\_MYSQL\_STAGE",  "FILE\_FORMAT\_NAME": "PYCSV\_FORMAT",  "FILE\_FORMAT\_TYPE": "CSV",  "FILE\_FIELD\_DELIMITER": ",",  "SNOWSQL\_CMD": "C:\\\"Program Files\"\\\"Snowflake SnowSQL\"\\snowsql",  "TABLE\_PREFIX": "SnowTableKK\_",  "DRIVER": "SnowflakeDSIIDriver",  "USERNAME": "**your\_username**",  "PASSWORD": "**your\_pwd**"  }  ]  },  "CLOUD\_PLATFORM": "AWS",  "AWS": {  "AWS\_REGION": "us-west-2",  "SNS\_NAME\_OR\_ARN": "",  "SQS\_NAME\_OR\_ARN": "",  "S3\_BUCKET\_NAME": "",  "S3\_OBJECT\_NAME": "",  "AWS\_SNS\_ARN": ""  },  "WAREHOUSE\_PLATFORM": "SNOWFLAKE",  "STREAMING\_SERVICE": "KAFKA",  "KAFKA": [  {  "TOPIC": " **Topic Name** ",  "SOURCE\_SYSTEM": "SALESFORCE", - **Source**  "TARGET\_SYSTEM": "HDFS", - **Target**  "BOOTSTRAP\_SERVERS": "['localhost:9092']"  }  ],  "SFTP": [  {  "SERVER": "192.168.1.116",  "PUT\_PATH": "D:\\SnowFlake-UR\\SNOWFLAKE\\Processed\\",  "GET\_PATH": "D:\\SnowFlake-UR\\SNOWFLAKE\\",  "USERNAME": "ftpuser",  "PASSWORD": "password",  "FILE\_PATTERN": ".\*\\.csv"  }  ],  "FTP": [  {  "SERVER": "192.168.1.16",  "PUT\_PATH": "/Processed/",  "GET\_PATH": "/",  "USERNAME": "ftpuser",  "PASSWORD": "password",  "FILE\_PATTERN": "^[aA0-zZ1]\*\\.csv"  }  ],  "PROCESS\_LOG": {  "LOG\_FILE\_RETENTION\_PERIOD": 10,  "LOG\_FILE\_MAX\_FILE\_SIZE": 50000000,  "PRINT\_LOG\_MESSGAE\_ON\_CONSOLE": "YES",  "ENABLE\_FILE\_LOGGING": "YES",  "LOG\_LEVEL": "WARNING",  "LOG\_FILE\_NAME": "LogForSalesForce\_"  },  "SYSTEM\_PARAMETERS": {  "LOCAL\_FILE\_PATH": "C:\\SalesForce-UR\\",  "CLEANUP\_HISTORICAL\_FILES": "NO",  "TIMEZONE": "Etc/UTC",  "CREDENTIAL": "SECRET\_MANAGER\_NO",  "ENTITY\_NAME": "SNOWFLAKE"  },  "SEARCH\_STRING\_FOR\_INDEX": {  "SNOWFLAKE1": "mpa43641",  "MSSQL1": "mssqlldb",  "MYSQL1": "mysql",  "KAFKA\_TOPIC\_SNOWDB": "salesforceTCRM",  "FTP\_LOCAL": "192.168.1.16"  },  "WEBAPI": {  "EnableDataSerachWithLIKE": "yes",  "WebPort": "800",  "EnableDataSearchAcrossAttributes": "yes",  "WebContentDisplayType": "TABLE",  "EnableUserToControlSearch": "Yes",  "EnableUserFullAccessToSearch": "yes"  },  "SALESFORCE": {  "AUTHENTICATION": {  "AuthURL": "https://login.salesforce.com/services/oauth2/token",  "UserName": " **Salesforce Username** ",  "Password": " **Salesforce Password** ",  "GrantType": "password", - **Keep it as same**  "ClientID": " **Your Client Id**",  "ClientSecret": " **Your Client Secrete** "  },  "GetUserRole": { "EndPointURL": "https://<**Your\_Instance\_Url**>.salesforce.com/services/data/v52.0/sobjects/UserRole/", "QueryOrHeaderParameters": "" }, "SOBJECTS\_DETAILED\_DATA": { "EndPointURL": "[https:// <**Your\_Instance\_Url**>.salesforce.com/services/data/v52.0/sobjects/"](https://usereaddy-fsc-dev-ed.my.salesforce.com/services/data/v52.0/sobjects/%22), "SOBJECTS": [ "Profile", "UserRole", "Contact1", "User" ] } },  "POSTAPI": {  "PostAccountDetail": {  "EndPointURL": "",  "HeaderParameters": "",  "BodyParameters": ""  }  },  "STREAMAPI": {  "SALESFORCE\_PUSHTOPICS": [  "pushtopic\_name1”, - **Your PushTopic Name**  "Pushtopic\_name2" - **Your PushTopic Name**  ]  }  }  } |

**SalesForce auth configuration details:**

* Go in configuration file change all the configuration for salesforce Auth.
* Auth url is same as mentioned but write your salesforce instance and write correct Version.
* Paste your username, password, ClientID and client secrete. Get client id and client secrete from Your salesforce connected app which we have done in First Step.
* Below the salesforce configuration there is a get request pulling the salesforce data.
* By this you can get all the data of mentioned salesforce Objects.
* You just need to type objects name in the "SOBJECTS" key to pull the data.

**database configuration details:**

* So, after the salesforce auth now we configure and set up Mysql
* Just Install MySql in your system and then configure it,
* Name the database engine under the database key in json format.
* Then mention server, Name, schema, table\_prefix, driver, auth\_type, Username and password done.

**kafka configuration details:**

* In kafka configuration firstly create the topic while writing the command in git bash. You can follow Third step of this document for it.
* After creating the topic now configure in the json configuration file for kafka setup.
* Under the KAFKA key mention topic name, source system, targeted system and server.
* You also can use SFTP and FTP file protocol services if you have requirement of it. Just set up the file according to your requirement.
* In our case Our source is “Salesforce” and Destination is “Mysql” or any other database and cloud platform.
* You can adjust source and target as per your requirement. This is the complete code for the integration and data migration between any cloud and desktop platform.

## Application Development Details:

|  |  |  |
| --- | --- | --- |
| **File name** | **File type** | **Working** |
| configuration | json | It holds all the configuration of this process. |
| SalesForceStreamCDC | py | This code would subscribe the CDC event channel by using ‘aiosfstream’ library. At the same it will produce the changed data event information in kafka producer and it will store the data in MySql. |
| KafkaConsumerStreamCDC | py | This code is for kafka consumer. It would consume all the changed data. |
| SalesForceStream | py | This code would subscribe the pushTopic by using ‘aiosfstream’ liberary. At the same it will produce the changed data event information in kafka producer and it will store the data in MySql. |
| KafkaConsumerStream | py | This code is for kafka consumer. It would consume all the changed data. |
| SalesForce | py | This file integrates salesforce and python to get all the data from salesforce org. |
| Utilities | py | This is a utility tool file for Database connectivity, Log setup and application configuration setup. |
|  |  |  |

# **Application Deployment and Testing:**

**Now, we are ready with our configuration and the set up.**

**Let’s Implement that how it looks:**

**Note:** when you will download the python code file you will get configuration file as well that you can change in.

Here, our objective is to pull the Changed data information from salesforce using Change data capture and store all the data into MySql.

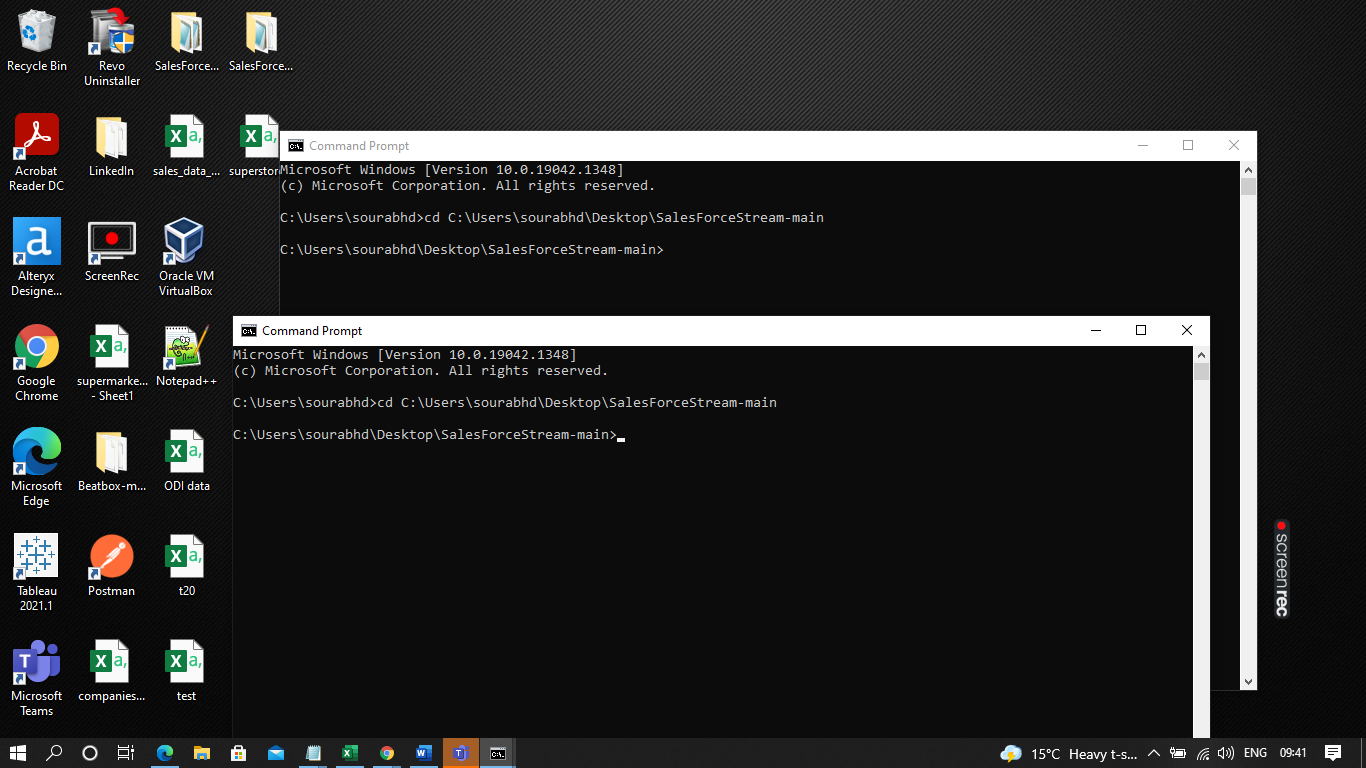
This is the salesforce streaming data so we are using python liberary called “aiosfstream”.

To run this python code you need to install aiosfstream using this command “pip install aiosfstream”.

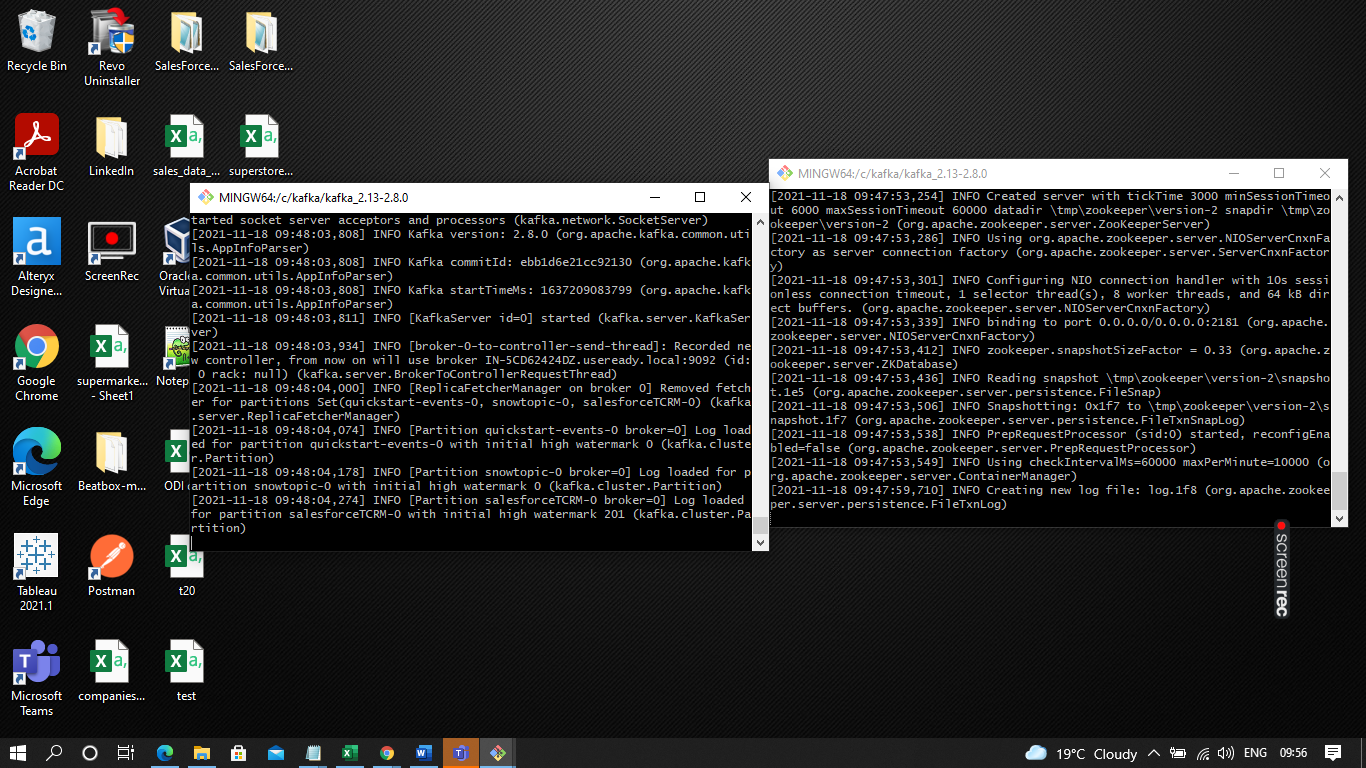
Import all other supported libraries which are mentioned in the code itself. Just “pip install” it.

Step 7. Implementation:

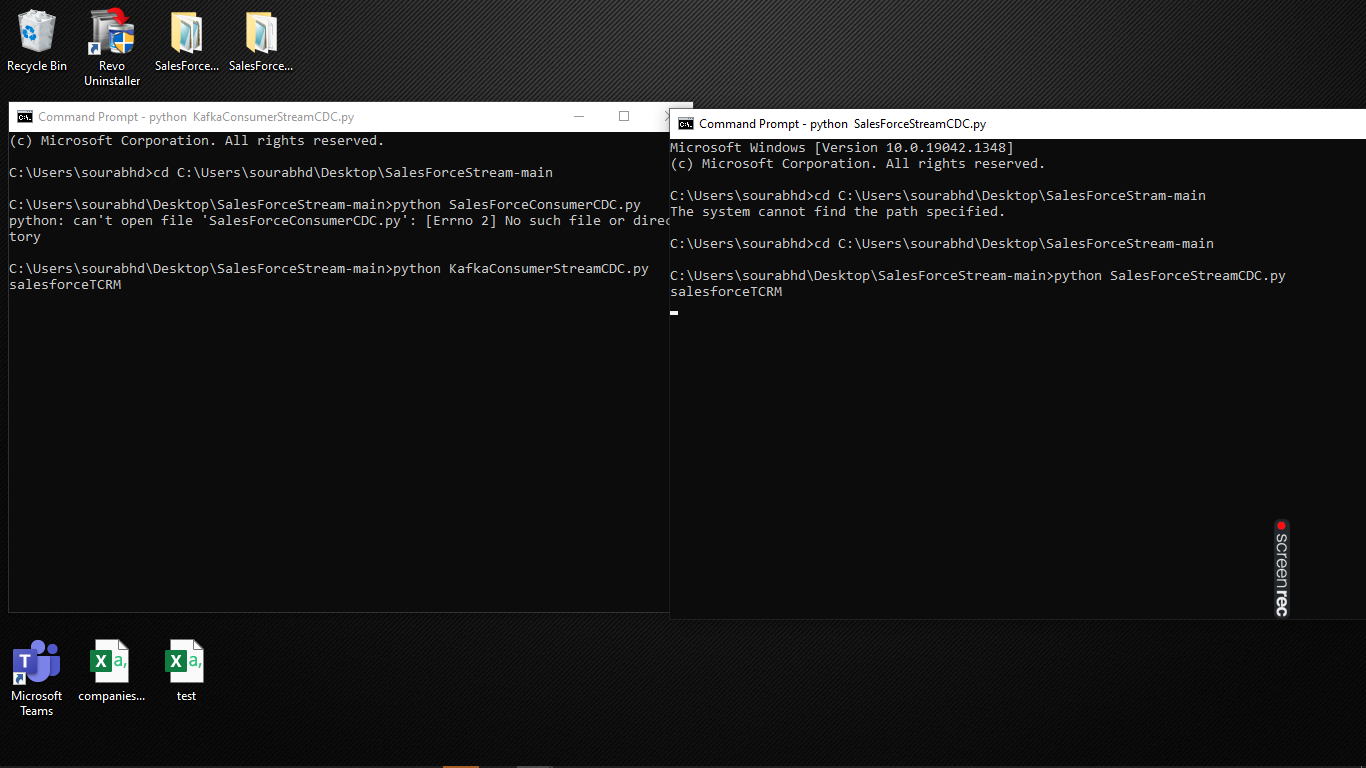
* Start the kafka zookeeper and server. (Follow Third step’s step 3)
* Then Open two command prompts and go into the python codes file using this command: **“cd C:\Users\Username\Desktop\\*python\_codes\_filename\*”** then enter it.
* Use this command in both cmd prompts.



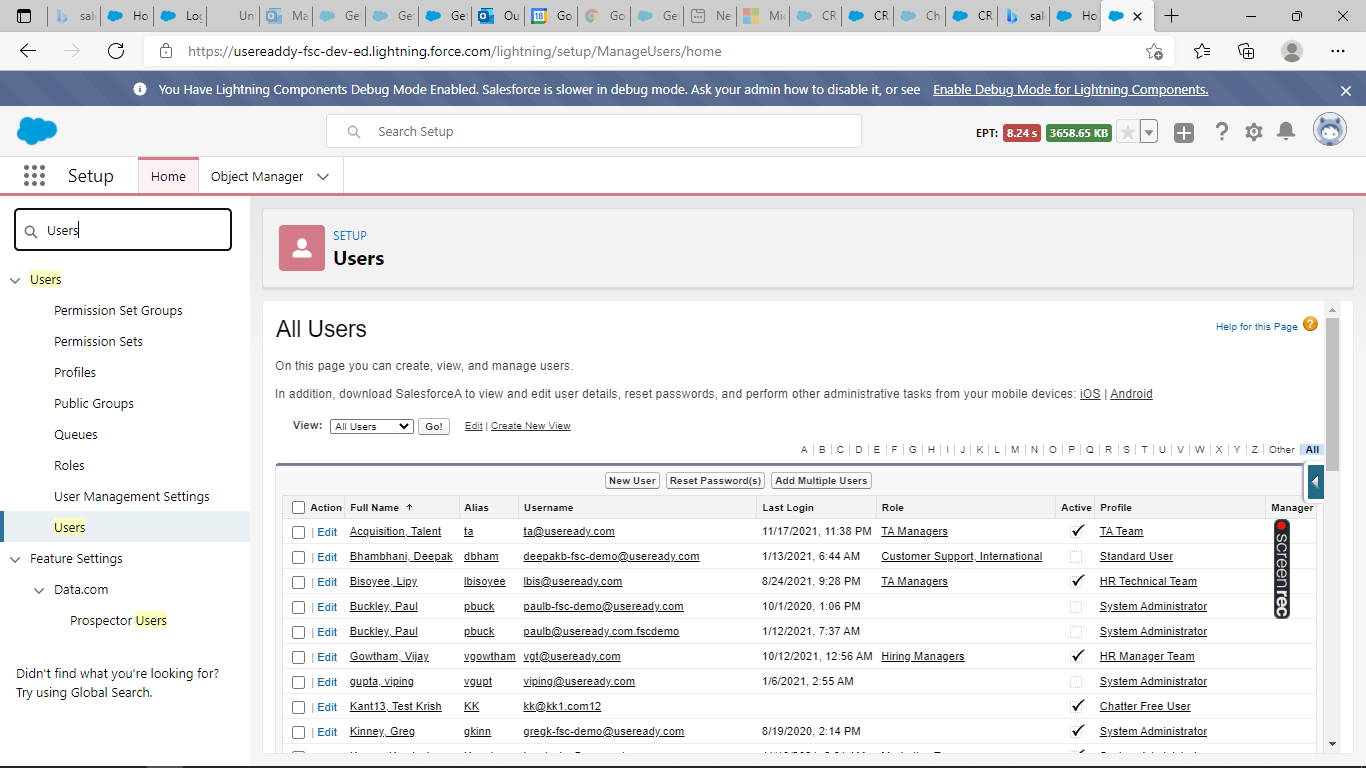
* Now kafka server is running. So, let’s run python codes in cmd.



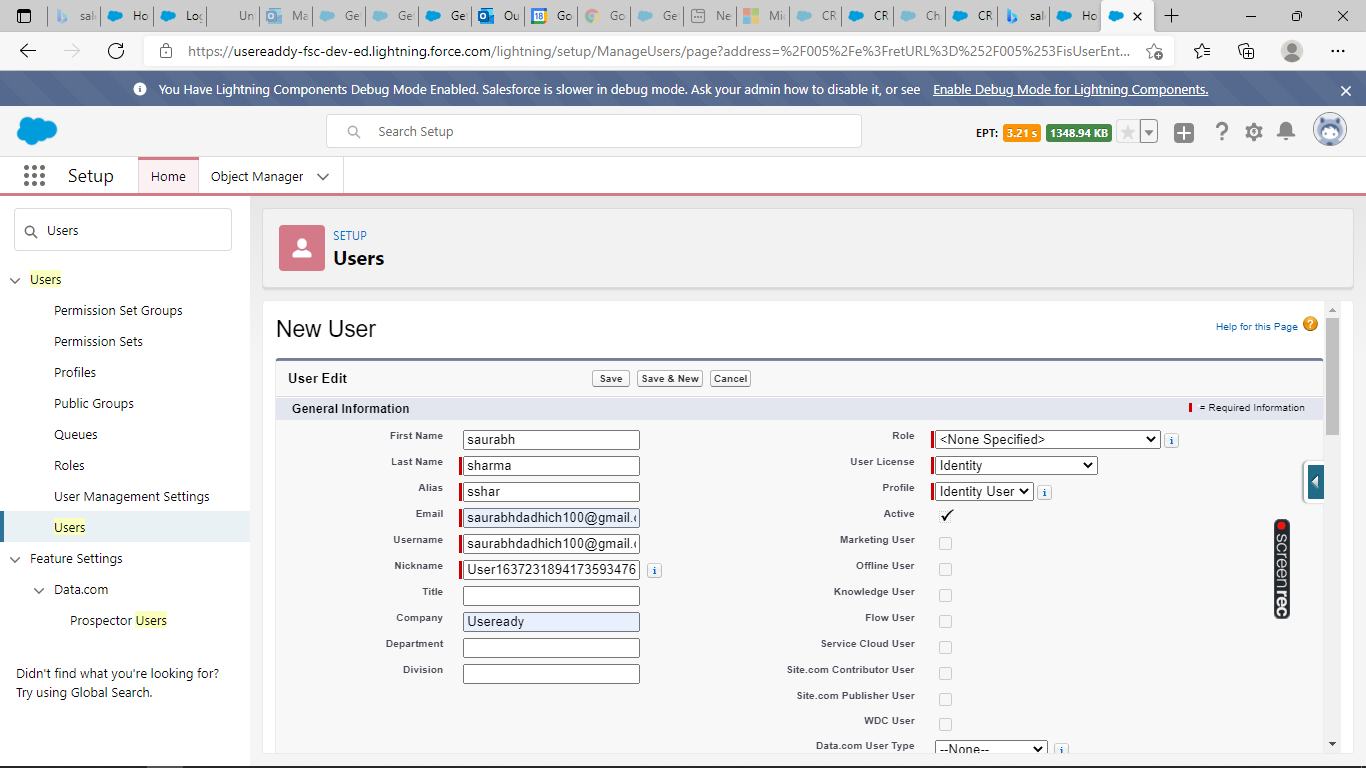
* In one cmd prompt, type “python SalesForceStreamCDC.py” to run the python file.
* In second cmd prompt, type “python SalesForceConsumerCDC.py”.
* In first code, we are pulling the changed data information from Salesforce and producing it from kafka. At the same time, we are dumping this data into MySql. how we will get the information we will look ahead.
* In second code we are consuming the information in the kafka.

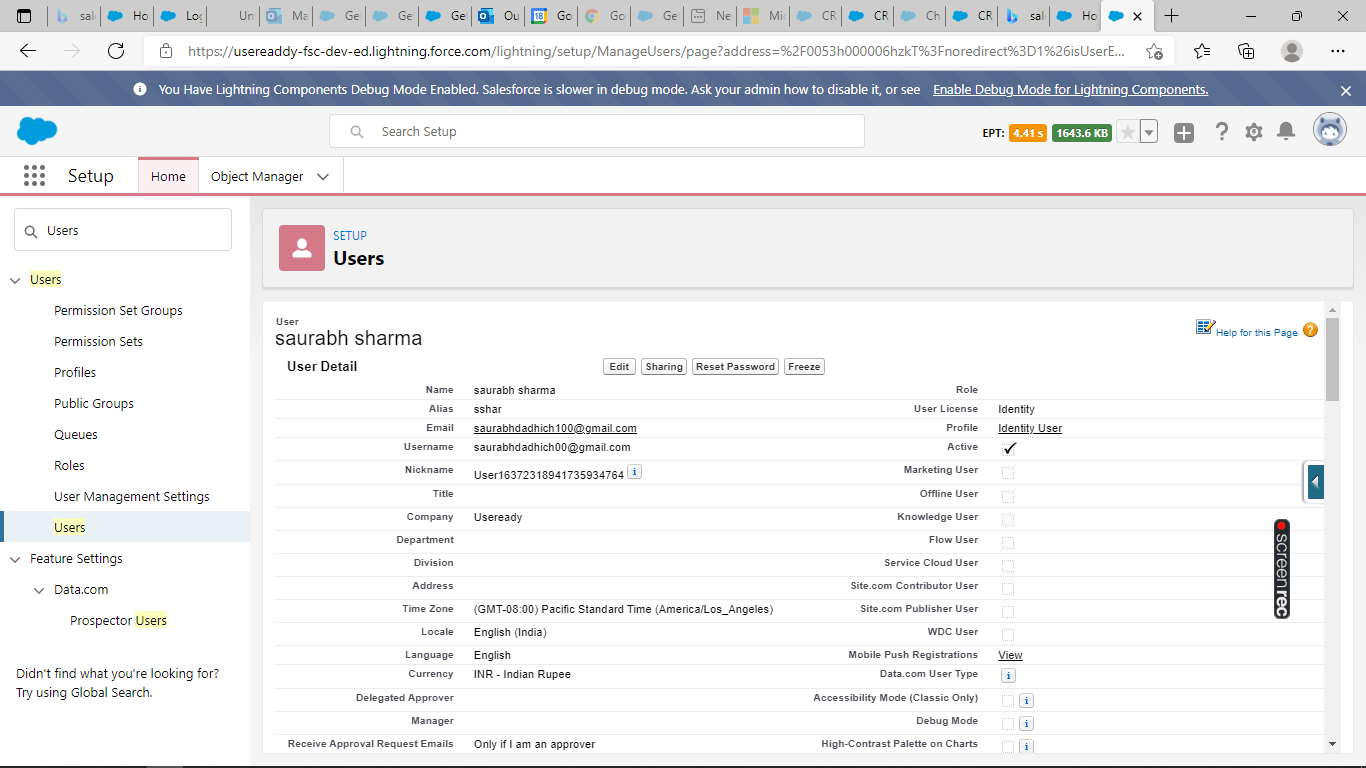


* Now Go to salesforce Org and Change some data or add new User in the Users section. (Refer Third Step’s step 5)
* Let’s add New User.

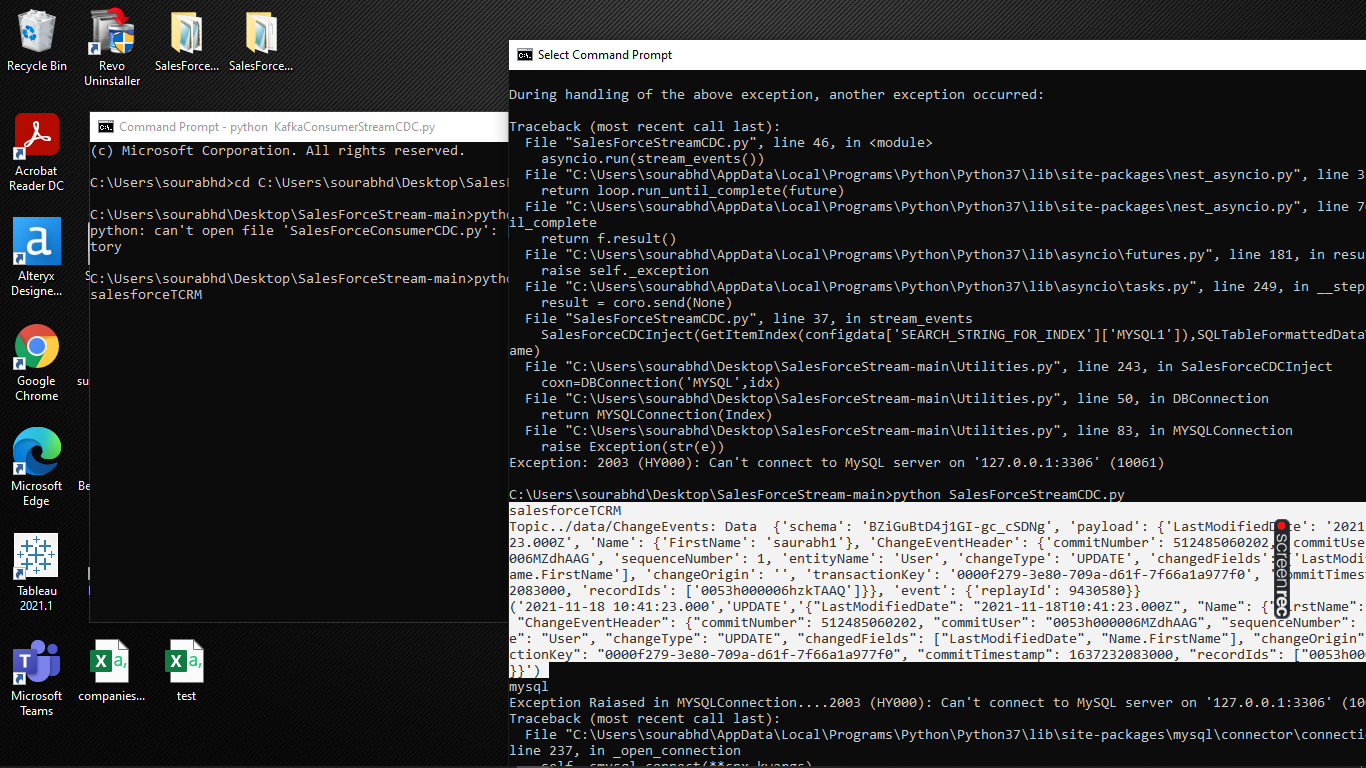


* Type Users in Quick find box.
* Click on Users and then Click on New User
* Fill all the details and save it.





* After adding a new user check your cmd prompt for the changed data information.



* so, we have got the data here.
* You can get all the information related to create, delete and update data of the salesforce Object.

**Note:**  even you can get all objects data by get request and soql query.

* We have one more python file named “Salesforce.py” for getting all objects data.
* Just go to the cmd prompt and type “python Salesforce.py”.
* You will get all the data from the object you have queried in configuration file.

