RDP ESG CFS File Workflow

Importing libararies

In [1]:
```
import os
import sys
from dotenv import dotenv_values
config = dotenv_values(".env")
```

Set RDP credentials and Initial Parameters

In [2]:
```
username = config['RDP_USERNAME'] #Or replace with your RDP Machin-ID
password = config['RDP_PASSWORD'] #Or replace with your RDP Password
clientId = config['RDP_APP_KEY'] #Or replace with your RDP APP Key

RDP_HOST= 'https://api.refinitiv.com'
acccess_token = None
refresh_token = None
expires_in = 0
```

RDP APIs Application Workflow

Step 1: Authentication with RDP APIs

Refinitiv Data Platform entitlement check is based on OAuth 2.0 specification. The first step of an application workflow is to get a token from RDP Auth Service, which will allow access to the protected resource, i.e. data REST API.

The API requires the following access credential information:

- Username: The username.
- Password: Password associated with the username.
- Client ID: This is also known as `AppKey`, and it is generated using an App key Generator. This unique identifier is defined for the user or application and is deemed confidential (not shared between users). The client_id parameter can be passed in the request body or as an "Authorization" request header that is encoded as base64.

The HTTP request for the RDP APIs Authentication service is as follows:

```
POST /auth/oauth2/v1/token HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
Host: api.refinitiv.com:443
Content-Length: XXX
```

```
username=RDP_USERNAME
&password=RDP_PASSWORD
&client_id=RDP_APP_KEY
&grant_type=password
&takeExclusiveSignOnControl=true
&scope=trapi
```

Once the authentication success, the function gets the RDP Auth service response message and keeps the following RDP token information in the variables.

- **access_token**: The token used to invoke REST data API calls as described above. The application must keep this credential for further RDP APIs requests.
- **refresh_token**: Refresh token to be used for obtaining an updated access token before expiration. The application must keep this credential for access token renewal.
- **expires_in**: Access token validity time in seconds.

Next, after the application received the Access Token (and authorization token) from RDP Auth Service, all subsequent REST API calls will use this token to get the data. Please find more detail regarding RDP APIs workflow in the following resources:

- RDP APIs: Introduction to the Request-Response API page.
- RDP APIs: Authorization - All about tokens page.

In [3]:
```
#step 1 - get RDP Access Token from RDP

import http.client
import requests
import json

# Send HTTP Request
auth_url = f'{RDP_HOST}/auth/oauth2/v1/token'
payload = f'grant_type=password&username={username}&client_id={clientId}&password={
try:
    response = requests.post(auth_url,
                             headers = {'Content-Type':'application/x-www-form-urle
                             data = payload,
                             auth = (clientId, '')
                )
except requests.exceptions.RequestException as exp:
    print(f'Caught exception: {exp}')


if response.status_code == 200:  # HTTP Status 'OK'
    print('Authentication success')
    access_token = response.json()['access_token']
    refresh_token = response.json()['refresh_token']
    expires_in = int(response.json()['expires_in'])

if response.status_code != 200:
    print(f'RDP authentication failure: {response.status_code} {response.reason}')
    print(f'Text: {response.text}')
```

```
Authentication success
```

Requesting Data from RDP APIs

That brings us to requesting the RDP APIs data. All subsequent REST API calls use the Access Token via the *Authorization* HTTP request message header as shown below to get the data.

- Header:
  - Authorization = `Bearer <RDP Access Token>`

Please notice *the space* between the `Bearer` and `RDP Access Token` values.

The application then creates a request message in a JSON message format or URL query parameter based on the interested service and sends it as an HTTP request message to the Service Endpoint. Developers can get RDP APIs the Service Endpoint, HTTP operations, and parameters from Refinitiv Data Platform's API Playground page - which is an interactive documentation site developers can access once they have a valid Refinitiv Data Platform account.

Requesting Bulk ESG Data

Step 2: Listing the packageId of the Bulk ESG Data

To request the ESG Bulk data, the first step is to send an HTTP `GET` request to the RDP `/file-store/v1/file-sets?bucket=ESG` endpoint to list all FileSets.

The HTTP Request structure is as follows:

```
GET /file-store/v1/file-sets?bucket=bulk-ESG HTTP/1.1
Host: api.refinitiv.com
Authorization: Bearer <Acces Token>
```

In [4]:
```
#step 2 - list Package IDs from bucket name

CFS_url = f'{RDP_HOST}/file-store/v1/file-sets?bucket=bulk-ESG'

try:
    response = requests.get(CFS_url, headers={'Authorization': f'Bearer {access_tok
except requests.exceptions.RequestException as exp:
    print(f'Caught exception: {exp}')


if response.status_code == 200:  # HTTP Status 'OK'
    print('Receive list Package IDs from RDP APIs')
else:
    print(f'RDP APIs: CFS request failure: {response.status_code} {response.reason}
    print(f'Text: {response.text}')
```

```
Receive list Package IDs from RDP APIs
```

Example of the first entry of package IDs, the pacakgeId is the `packageId` field.

In [5]:
```
print(json.dumps(response.json()['value'][0], sort_keys=True, indent=2, separators=
```


```
{
  "attributes":[
    {
      "name":"ContentType",
      "value":"ESG Raw Full A"
    }
  ],
  "availableFrom":"2023-10-22T20:50:47Z",
  "availableTo":"2023-11-05T20:50:47Z",
  "bucketName":"bulk-ESG",
  "contentFrom":"1970-01-01T00:00:00Z",
  "contentTo":"2023-10-22T20:17:56Z",
  "created":"2023-10-22T20:50:47Z",
  "files":[
    "4219-9bfa-7dd72e15-9830-4eb82c52f46d",
    "434d-a1a1-8e0a4698-adb4-0daf6ea3096f",
    "4570-3bf3-85145c7e-b6ce-cb45c54a3c03",
    "45b5-5812-0fb594c1-b8f8-58901628b4e9",
    "486b-6890-4759a311-9a9c-f5fb9c0bcf8c",
    "49b3-9160-50514d30-a03b-7ef1f1a35a58",
    "49dc-6fbf-dc234e97-80be-78ec07c554a5",
    "4d85-432d-e7b9e87c-a88f-c0719b72b782",
    "4ddd-98fe-3f5ecb00-a827-fa806f4520d6",
    "4f86-ed4d-50e0ef2d-a323-d6415b6594ac"
  ],
  "id":"400a-456b-28d5be8b-8f8a-ee5fcef5ae7a",
  "modified":"2023-10-22T21:13:37Z",
  "name":"Bulk-ESG-Global-Raw-Full-SchemeA-v1-Jsonl-Init-2023-10-22T20:24:19.238Z",
  "numFiles":10,
  "packageId":"4133-fe0b-fe1981eb-b254-ffb9f828b286",
  "status":"READY"
}
```


The next step is choosing the package Id.

In [6]:
```
packageId = response.json()['value'][0]['packageId']
packageId
```

Out[6]:
```
'4133-fe0b-fe1981eb-b254-ffb9f828b286'
```

Step 3: Listing the Filesets of the Bulk ESG Data with the packageId

The next step is calling the CFS API with the buket name and package Id to list all FileSets using **the package Id**.

API endpint is `/file-store/v1/file-sets?bucket=bulk-ESG&packageId={packageId}`

The HTTP Request structure is as follows:

```
GET /file-store/v1/file-sets?bucket=bulk-ESG&packageId={packageId}
HTTP/1.1
Host: api.refinitiv.com
Authorization: Bearer <Access Token>
```

In [7]:
```
#step 3 - get file id from bucket name

CFS_url = f'{RDP_HOST}/file-store/v1/file-sets?bucket=bulk-ESG&packageId={packageId

try:
    response = requests.get(CFS_url, headers={'Authorization': f'Bearer {access_tok
except requests.exceptions.RequestException as exp:
    print(f'Caught exception: {exp}')


if response.status_code == 200:  # HTTP Status 'OK'
    print('Receive FileSets list from RDP APIs')
else:
    print(f'RDP APIs: CFS request failure: {response.status_code} {response.reason}
    print(f'Text: {response.text}')
```

```
Receive FileSets list from RDP APIs
```

In [8]:
```
print(json.dumps(response.json()['value'][0], sort_keys=True, indent=2, separators=
```

```
{
  "attributes":[
    {
      "name":"ContentType",
      "value":"ESG Raw Full A"
    }
  ],
  "availableFrom":"2023-10-22T20:50:47Z",
  "availableTo":"2023-11-05T20:50:47Z",
  "bucketName":"bulk-ESG",
  "contentFrom":"1970-01-01T00:00:00Z",
  "contentTo":"2023-10-22T20:17:56Z",
  "created":"2023-10-22T20:50:47Z",
  "files":[
    "4219-9bfa-7dd72e15-9830-4eb82c52f46d",
    "434d-a1a1-8e0a4698-adb4-0daf6ea3096f",
    "4570-3bf3-85145c7e-b6ce-cb45c54a3c03",
    "45b5-5812-0fb594c1-b8f8-58901628b4e9",
    "486b-6890-4759a311-9a9c-f5fb9c0bcf8c",
    "49b3-9160-50514d30-a03b-7ef1f1a35a58",
    "49dc-6fbf-dc234e97-80be-78ec07c554a5",
    "4d85-432d-e7b9e87c-a88f-c0719b72b782",
    "4ddd-98fe-3f5ecb00-a827-fa806f4520d6",
    "4f86-ed4d-50e0ef2d-a323-d6415b6594ac"
  ],
  "id":"400a-456b-28d5be8b-8f8a-ee5fcef5ae7a",
  "modified":"2023-10-22T21:13:37Z",
  "name":"Bulk-ESG-Global-Raw-Full-SchemeA-v1-Jsonl-Init-2023-10-22T20:24:19.238Z",
  "numFiles":10,
  "packageId":"4133-fe0b-fe1981eb-b254-ffb9f828b286",
  "status":"READY"
}
```


The File ID is in the `files` array

In [9]:
```
# try just one file
file_id = response.json()['value'][0]['files'][1]
file_id
```

Out[9]:
```
'434d-a1a1-8e0a4698-adb4-0daf6ea3096f'
```

Step 4: Get the Bulk file URL on AWS S3

The last step is downloading the FIle using File ID with the RDP `/file-store/v1/files/{file ID}/stream` endpoint.

The HTTP Request structure is as follows:

```
GET /file-store/v1/files/{fileId}/stream?doNotRedirect=true HTTP/1.1
Host: api.refinitiv.com
Authorization: Bearer <Access Token>
```

In [10]:
```
#step 3 - get file stream (content) from file id
```

```
FileID_url = f'{RDP_HOST}/file-store/v1/files/{file_id}/stream?doNotRedirect=true'

try:
    response = requests.get(FileID_url, headers={'Authorization': f'Bearer {access_
except requests.exceptions.RequestException as exp:
    print(f'Caught exception: {exp}')


if response.status_code == 200:  # HTTP Status 'OK'
    print('Receive File URL from RDP APIs')
else:
    print(f'RDP APIs: CFS request failure: {response.status_code} {response.reason}
    print(f'Text: {response.text}')
```

```
Receive File URL from RDP APIs
```

The File URL is in the `url` attribute.

In [11]:
```
file_url = response.json()['url']
file_url
```

Out[11]:
```
'https://a206464-bulk-esg.s3.amazonaws.com/Bulk-ESG-Global-Raw-Full-SchemeA-v1/202
3/10/22/Bulk-ESG-Global-Raw-Full-SchemeA-v1-Init-2023-10-22T20%3A24%3A19.238Z-part
09.jsonl.gz?x-request-Id=1d6e6df1-6901-4323-8b05-ee315aedb605&x-package-id=4133-fe
0b-fe1981eb-b254-ffb9f828b286&x-client-app-id=b4842f3904fb4a1fa18234796368799086c6
3541&x-file-name=Bulk-ESG-Global-Raw-Full-SchemeA-v1-Init-2023-10-22T20%3A24%3A19.
238Z-part09.jsonl.gz&x-fileset-id=400a-456b-28d5be8b-8f8a-ee5fcef5ae7a&x-bucket-na
me=bulk-ESG&x-uuid=GESG1-178570&x-file-Id=434d-a1a1-8e0a4698-adb4-0daf6ea3096f&x-f
ileset-name=Bulk-ESG-Global-Raw-Full-SchemeA-v1-Jsonl-Init-2023-10-22T20%3A24%3A1
9.238Z&x-event-external-name=cfs-claimCheck-download&X-Amz-Security-Token=IQoJb3Jp
Z2luX2VjEEwaCXVzLWVhc3QtMSJGMEQCIGxneQ7M7AZaRfLMbCL%2FLUgD%2BrPtV%2F6pePOuZS1Hy%2B
CWAiBCUpdolXLgY28YTQMPRJOcAOqjauYbO%2FbQISpu2gaysyqaAgh0EAQaDDY0MjE1NzE4MTMyNiIMkh
e5vfRhzBdAwjquKvcBUGveWLvB%2FvXrX9nAlIicTIldEHnrUBU8GNuOXnIMOzFJMSeSIS7h1NHblx8Z10
YNWy4DlNm8liVay3%2Ba9k6mdCO6LlPfSUCaQx3%2BaMoXLWb3968eLIoROchPIQxenNP2Aartpn%2FhOx
khFOxR0p%2FWSLaABrm1or3MqtvJ3IYidsTyXNk7LUmgv2Kn5VmLtpPVKZAjBza%2BrxEWIzeQW5c56FNa
8%2BIMTyR6NnLO299UVMx1Xip38YiwLxCtU23spEXbPUaXOIWW7VgV0fyLYkzZWAqQNFoCV5zIdDE4zcSI
aaUyMPKnQlDVZhwPRHrqprD26wqeMUihzTCxlumpBjqeAclXvqolpJLN6qrrKEFSHSCnSGLK4%2Bo4Xaak
dmCpze9KGIiOlxs0yIafzh6rjQtTpopS5tmGyiCHt5%2B7JzHTP7x%2BYNJMtsGhjmM5OOQ2lRsoJwfOz8
1bujTpPxA%2BPbh149tmEojSwbByPgaAQt4Tq2aYKK3ro4j9QAYvs5PTat5vOxkrqAu87%2FPPAYSEeVW0
FSZvGRk%2Bgns0hGaai58G&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20231026T111913
Z&X-Amz-SignedHeaders=host&X-Amz-Expires=21600&X-Amz-Credential=ASIAZLA4M7GHKPAQQW
BZ%2F20231026%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Signature=97055e40299f09dda284
1daf49d876daeb18aaa33bab6a351a1ff281abe873a0'
```

Step 5: Downloading the file

Based on the S3 `file_url` above, the actual file name is *Bulk-ESG-Global-Raw-Full-SchemeA-v1-Init-2023-10-22T20_24_19.238Z-part09.jsonl.gz*. So you need to replace the escape character `%3A` with `_` (underscore) character.

In [13]:
```
#Downlaod file
zipfilename = file_url.split("?")[0].split("/")[-1].replace("%3A","_")
print(f'Downloading File {zipfilename} ...')
```

```
try:
    response = requests.get(file_url)
except requests.exceptions.RequestException as exp:
    print(f'Caught exception: {exp}')

if response.status_code == 200:  # HTTP Status 'OK'
    print('Receive File Successfully')
    open(zipfilename, 'wb').write(response.content)
    print(f'{zipfilename} Saved')
else:
    print(f'RDP APIs: Request file failure: {response.status_code} {response.reason
    print(f'Text: {response.text}')
```

```
Downloading File Bulk-ESG-Global-Raw-Full-SchemeA-v1-Init-2023-10-22T20_24_19.238Z-p
art09.jsonl.gz ...
Receive File Successfully
Bulk-ESG-Global-Raw-Full-SchemeA-v1-Init-2023-10-22T20_24_19.238Z-part09.jsonl.gz Sa
ved
```

And then unzip the file.

The normal ESG Bulk file type provides data for developers in **.gz** file.

In [16]:
```
#unzip file
import gzip
import shutil
try:
    unzipfilename = zipfilename.split('.gz')[0]
    print(f'Unzip to {unzipfilename} ...')
    with gzip.open(zipfilename, 'rb') as f_in:
        with open(unzipfilename, 'wb') as f_out:
            shutil.copyfileobj(f_in, f_out)
    print('Done')
except Exception as e:
    print('The error is: ',e)
```

```
Unzip to Bulk-ESG-Global-Raw-Full-SchemeA-v1-Init-2023-10-22T20_24_19.238Z-part09.js
onl ...
Done
```

In [15]:
```
# View some data (for nomral ESG Bulk only)
with open(unzipfilename) as f:
    data = f.read(2000)
    print(data)
```

{"ObjectId":"4295906353;7","StatementDetails":{"OrganizationId":"4295906353","FinancialPeriodEndDate":"2009-12-31T00:00:00.000Z","FinancialPeriodFiscalYear":"2009","FinancialPeriodIsIncomplete":"false","EsgOrAtdIndicator":"ESG"},"ESGOrganization":{"Names":{"Name":{"OrganizationName":[{"NormalizedName":"KBR Inc"}]}}},"CommunityDataPoints":{"AntiCompetitionControversiesCount":{"Value":null,"ValueDate":null},"BusinessEthicsControversies":{"Value":1,"ValueDate":"2012-12-03T13:49:50.947Z"},"CommunityLendingAndInvestments":{"ValueCurrencyId_v2":null,"Value":null,"ValueDate":null,"ValueScore":null},"CorporateResponsibilityAwards":{"Value":false,"ValueDate":null,"ValueScore":"0"},"CrisisManagementSystems":{"Value":false,"ValueDate":null},"CriticalCountriesControversies":{"Value":null,"ValueDate":null},"CriticalCountry1":{"Value":null,"ValueDate":null,"ValueScore":null},"CriticalCountry2":{"Value":null,"ValueDate":null},"CriticalCountry3":{"Value":null,"ValueDate":null},"CriticalCountry4":{"Value":null,"ValueDate":null},"CriticalCountry5":{"Value":null,"ValueDate":null},"DiseasesOfTheDevelopingWorld":{"Value":false,"ValueDate":null,"ValueScore":null},"DonationsTotal":{"ValueCurrencyId_v2":"USD","Value":"5150000","ValueDate":"2010-04-29T05:55:01.420Z"},"EmployeeEngagementVoluntaryWork":{"Value":true,"ValueDate":"2010-04-27T06:26:18.297Z"},"ExtractiveIndustriesTransparencyInitiative":{"Value":false,"ValueDate":null,"ValueScore":null},"ImprovementToolsBusinessEthics":{"Value":true,"ValueDate":"2010-04-27T06:26:18.293Z","ValueScore":"0.6417322834645669"},"IntellectualPropertyControversies":{"Value":null,"ValueDate":null},"LobbyingContributionAmount":{"ValueCurrencyId_v2":null,"Value":null,"ValueDate":null},"OecdGuidelinesForMultinationEnterprises":{"Value":false,"ValueDate":null,"ValueScore":null},"PolicyBriberyAndCorruption":{"Value":true,"ValueDate":"2010-04-27T06:26:18.293Z","ValueScore":"0.6417322834645669"},"PolicyBusinessEthics":{"Value":true,"ValueDate":"2010-04-27T06:26:

In [ ]: