Summary:

Please complete the following tasks using either Java or Python, the code will run daily so we need it to be easy for maintenance and issue debugging.

Note:

Please note that the SFTP server and AWS bucket mentioned in the task are not accessible. Just write the code based on the sample data provided.

Main funcionalities:

1. Step 1, using code to download files

- Download the data file(xml format) for the current day from an SFTP server.
- Server details:
  - SFTP Host: testFTP.dv.com
  - SFTP User: testuser
  - SFTP Password: 123456
  - SFTP Path: /data
- The SFTP server generates new files daily, but file names do not include the date. You must determine if a file is from the current day based on its last modification time in SFTP.
- Write code to download the file for today, save it locally and delete it from SFTP.

2. Step 2, transform files

- Read the downloaded XML file and convert it to a line-by-line JSON format.There are lots of fields, please refer to here.
- Each JSON object should include the following fields: UserID, UserName, UserAge, EventTime.
- Convert the EventTime field to ISO 8601 format: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'.
- Calculate the average UserAge (UserAgeAvg) across all records in the file.
- Based on the UserAge compared to UserAgeAvg, store records in two separate JSON files:
  - If UserAge is greater than UserAgeAvg, store the record in above_average_output.json.
  - Otherwise, store the record in below_average_output.json.

3. Step 3, upload Files to AWS s3 bucket

- Upload the generated above_average_output.json and below_average_output.json files to an AWS S3 bucket.
- S3 details:
  - S3 Bucket: testbucket
  - S3 Path: /output
  - AWS Access Key: testKeyString
  - Secret Access Key: testKeyString
- Write code to handle the file upload and ensure error handling during the process.

Additional Requirements:

- Provide brief documentation describing your solution, the tools and libraries used, and any assumptions made.
- Ensure the code is well-organized and commented for clarity.
- Handle exceptions and errors gracefully to ensure robust code.
- The converted JSON files should be stored in a line-by-line format, with each line being a separate JSON object.

Sample Data(xml file):

```
<?xml version="1.0" encoding="UTF-8"?>
<Users>
   <User>
      <UserID>1</UserID>
      <UserName>Alice</UserName>
      <UserAge>30</UserAge>
      <EventTime>2024-07-30T10:00:00</EventTime>
   </User>
   <User>
      <UserID>2</UserID>
      <UserName>Bob</UserName>
      <UserAge>25</UserAge>
      <EventTime>2024-07-30T11:00:00</EventTime>
   </User>
   <User>
      <UserID>3</UserID>
      <UserName>Charlie</UserName>
      <UserAge>35</UserAge>
      <EventTime>2024-07-30T12:00:00</EventTime>
```

```
</User>
<User>
  <UserID>4</UserID>
  <UserName>David</UserName>
  <UserAge>28</UserAge>
  <EventTime>2024-07-30T13:00:00</EventTime>
</User>
<User>
  <UserID>5</UserID>
  <UserName>Eve</UserName>
  <UserAge>40</UserAge>
  <EventTime>2024-07-30T14:00:00</EventTime>
</User>
<User>
  <UserID>6</UserID>
  <UserName>Frank</UserName>
  <UserAge>22</UserAge>
  <EventTime>2024-07-30T15:00:00</EventTime>
</User>
<User>
  <UserID>7</UserID>
  <UserName>Grace</UserName>
  <UserAge>33</UserAge>
  <EventTime>2024-07-30T16:00:00</EventTime>
</User>
<User>
  <UserID>8</UserID>
  <UserName>Heidi</UserName>
  <UserAge>27</UserAge>
  <EventTime>2024-07-30T17:00:00</EventTime>
</User>
<User>
  <UserID>9</UserID>
  <UserName>Ivy</UserName>
  <UserAge>31</UserAge>
  <EventTime>2024-07-30T18:00:00</EventTime>
</User>
<User>
  <UserID>10</UserID>
  <UserName>Judy</UserName>
  <UserAge>29</UserAge>
  <EventTime>2024-07-30T19:00:00</EventTime>
</User>
```

```
</Users>
```

Sample Results:

For the sample XML data above, the resulting JSON files might look like this:

- above_average_output.json:

```
{"UserID": "1", "UserName": "Alice", "UserAge": "32", "EventTime": "2024-07-30T10:00:00.000Z"}
{"UserID": "3", "UserName": "Charlie", "UserAge": "35", "EventTime": "2024-07-30T12:00:00.000Z"}
{"UserID": "5", "UserName": "Eve", "UserAge": "40", "EventTime": "2024-07-30T14:00:00.000Z"}
{"UserID": "7", "UserName": "Grace", "UserAge": "33", "EventTime": "2024-07-30T16:00:00.000Z"}
{"UserID": "9", "UserName": "Ivy", "UserAge": "31", "EventTime": "2024-07-30T18:00:00.000Z"}
```

- below_average_output.json

```
{"UserID": "2", "UserName": "Bob", "UserAge": "25", "EventTime": "2024-07-30T11:00:00.000Z"}
{"UserID": "4", "UserName": "David", "UserAge": "28", "EventTime": "2024-07-30T13:00:00.000Z"}
{"UserID": "6", "UserName": "Frank", "UserAge": "22", "EventTime": "2024-07-30T15:00:00.000Z"}
{"UserID": "8", "UserName": "Heidi", "UserAge": "27", "EventTime": "2024-07-30T17:00:00.000Z"}
{"UserID": "10", "UserName": "Judy", "UserAge": "29", "EventTime": "2024-07-30T19:00:00.000Z"}
```