```python
In [15]:  import yfinance as yf
          import pymongo
          from apscheduler.schedulers.background import BackgroundScheduler
          from datetime import datetime, timedelta
```

Main Code =>

```python
In [ ]:   import yfinance as yf
          import pymongo
          from apscheduler.schedulers.background import BackgroundScheduler
          from datetime import datetime, timedelta

          # Connection to MongoDB
          client = pymongo.MongoClient("mongodb://localhost:27017/")
          db = client["stock_data"]
          collection = db["icici_bank"]

          # ICICI Bank ticker
          ticker = "ICICIBANK.NS"

          # function to fetch and store stock data
          def fetch_and_store_data():
              now = datetime.now()
              start_time = datetime(now.year, now.month, now.day, 11, 15, 0)
              end_time = datetime(now.year, now.month, now.day, 14, 15, 0)

              if start_time <= now <= end_time:
                  data = yf.Ticker(ticker).history(period="15m", interval="15m")
                  data["Datetime"] = data.index
                  data = data.reset_index(drop=True)
                  data_dict = data.to_dict(orient="records")

                  collection.insert_many(data_dict)
                  print(f"Stored data for {now.strftime('%Y-%m-%d %H:%M:%S')}")

          #  APS Scheduler job to run every 15 minutes
          scheduler = BackgroundScheduler()
          scheduler.add_job(fetch_and_store_data, "interval", minutes=15)
          scheduler.start()

          try:
              # scheduler for a week (adjust the end date as needed)
              end_date = datetime.now() + timedelta(days=7)
              while datetime.now() <= end_date:
                  pass
          except (KeyboardInterrupt, SystemExit):
              # scheduler gracefully when manually interrupted
              scheduler.shutdown()
          # END OF CODE
```

```python
In [ ]:
```

```python
In [16]:  pip install pymongo
```

Defaulting to user installation because normal site-packages is not writeableNote: yo
u may need to restart the kernel to use updated packages.

Requirement already satisfied: pymongo in c:\users\pritik\appdata\roaming\python\pyth
on311\site-packages (4.6.0)
Requirement already satisfied: dnspython<3.0.0,>=1.16.0 in c:\users\pritik\appdata\ro
aming\python\python311\site-packages (from pymongo) (2.4.2)

```
In [17]:  pip install apscheduler
```

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: apscheduler in c:\users\pritik\appdata\roaming\python
\python311\site-packages (3.10.4)
Requirement already satisfied: six>=1.4.0 in c:\programdata\anaconda\lib\site-package
s (from apscheduler) (1.16.0)
Requirement already satisfied: pytz in c:\programdata\anaconda\lib\site-packages (fro
m apscheduler) (2022.7)
Requirement already satisfied: tzlocal!=3.*,>=2.0 in c:\users\pritik\appdata\roaming
\python\python311\site-packages (from apscheduler) (5.2)
Requirement already satisfied: tzdata in c:\users\pritik\appdata\roaming\python\pytho
n311\site-packages (from tzlocal!=3.*,>=2.0->apscheduler) (2023.3)
Note: you may need to restart the kernel to use updated packages.

```
In [18]:  client = pymongo.MongoClient("mongodb://localhost:27017/")
          db = client["stock_data"]
          collection = db["icici_bank"]
```

```
In [9]:   # ICICI Bank ticker
          ticker = "ICICIBANK.NS"
```

```
In [10]:  #  function to fetch and store stock data
          def fetch_and_store_data():
              now = datetime.now()
              start_time = datetime(now.year, now.month, now.day, 11, 15, 0)
              end_time = datetime(now.year, now.month, now.day, 14, 15, 0)

              if start_time <= now <= end_time:
                  data = yf.Ticker(ticker).history(period="15m", interval="15m")
                  data["Datetime"] = data.index
                  data = data.reset_index(drop=True)
                  data_dict = data.to_dict(orient="records")

                  collection.insert_many(data_dict)
                  print(f"Stored data for {now.strftime('%Y-%m-%d %H:%M:%S')}")
```

```
In [11]:  # APS Scheduler job to run every 15 minutes
          scheduler = BackgroundScheduler()
          scheduler.add_job(fetch_and_store_data, "interval", minutes=15)
          scheduler.start()
```

```
In [ ]:   try:
              # scheduler for a week (adjust fthe end date as needed)
              end_date = datetime.now() + timedelta(days=7)
              while datetime.now() <= end_date:
                  pass
          except (KeyboardInterrupt, SystemExit):
              # Shutdown the scheduler gracefully when manually interrupted
              scheduler.shutdown()
```

In [ ]:

In [ ]:

In [ ]: