PySpark Certification T...

Apache Spark Streaming - Processing Multiple Batches > Presentation

Course Content

- Deep Dive into Spark ...
- Understanding Apache...
- Apache Spark Streami...
  - Presentation
  - In Class Demo - Inpu...
- Apache Spark Streami...
- In-class Project
- Spark GraphX
- Certification Project - ...

Personal Library

Presentation 3 / 75

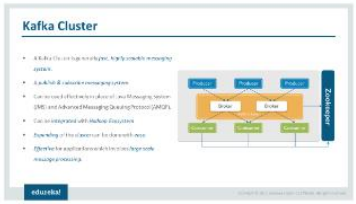
COURSE OUTLINE

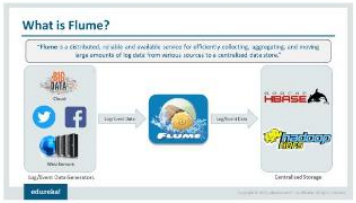
MODULE 10

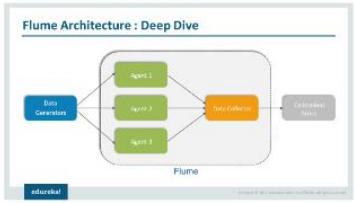
- Introduction to Big Data Hadoop and Spark
- Introduction to Python for Apache Spark
- Functions, OOPs, Modules in Python
- Deep Dive into Apache Spark Framework
- Playing with Spark RDDs
- Data Frames and Spark SQL
- Machine Learning using Spark MLlib
- Deep Dive into Spark MLlib
- Understanding Apache Kafka and Apache Flume
- **Apache Spark Streaming - Processing Multiple Batches**
- Apache Spark Streaming - Data Sources
- Implementing an End-to-End Project

Presentation 4 / 75

Recap





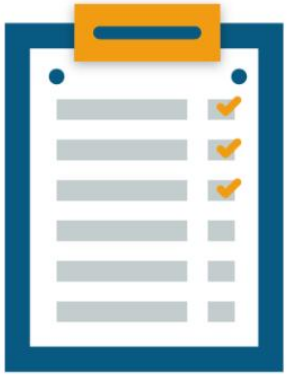


edureka!

Presentation 5 / 75

Topics

- Spark Streaming Features
- Spark Workflow
- Streaming Context
- Dstream
- Caching
- Accumulators, Broadcast Variables and Checkpoints
- Streaming Word Count
- Stateful Operators



edureka!

Presentation 6 / 75

Objectives

After completing this module, you should be able to:

- Analyze the Drawbacks in Existing Computing Methods
- Describe why Streaming is Necessary
- Explain what is Spark Streaming
- Describe Spark Streaming Features
- Describe Spark Streaming Workflow
- Understand how Uber uses Streaming Data
- Explain Streaming Context and DStreams
- Describe Transformations on DStreams
- Execute a Streaming WordCount Program

edureka!

Presentation 7 / 75

Drawbacks in Traditional Computing Methods

- Traditionally we used to perform analysis only on saved data
- That gives us results, but sometimes ***it's too late to act according to that result***
- Our best understanding comes ***when conclusions fit evidence*** and evidence and analysis leads to a good way of life
- Companies need to work on real time data, to improve their ***business and customer review***
- Banks can't detect ***online real-time fraud detection*** on saved data

edureka!

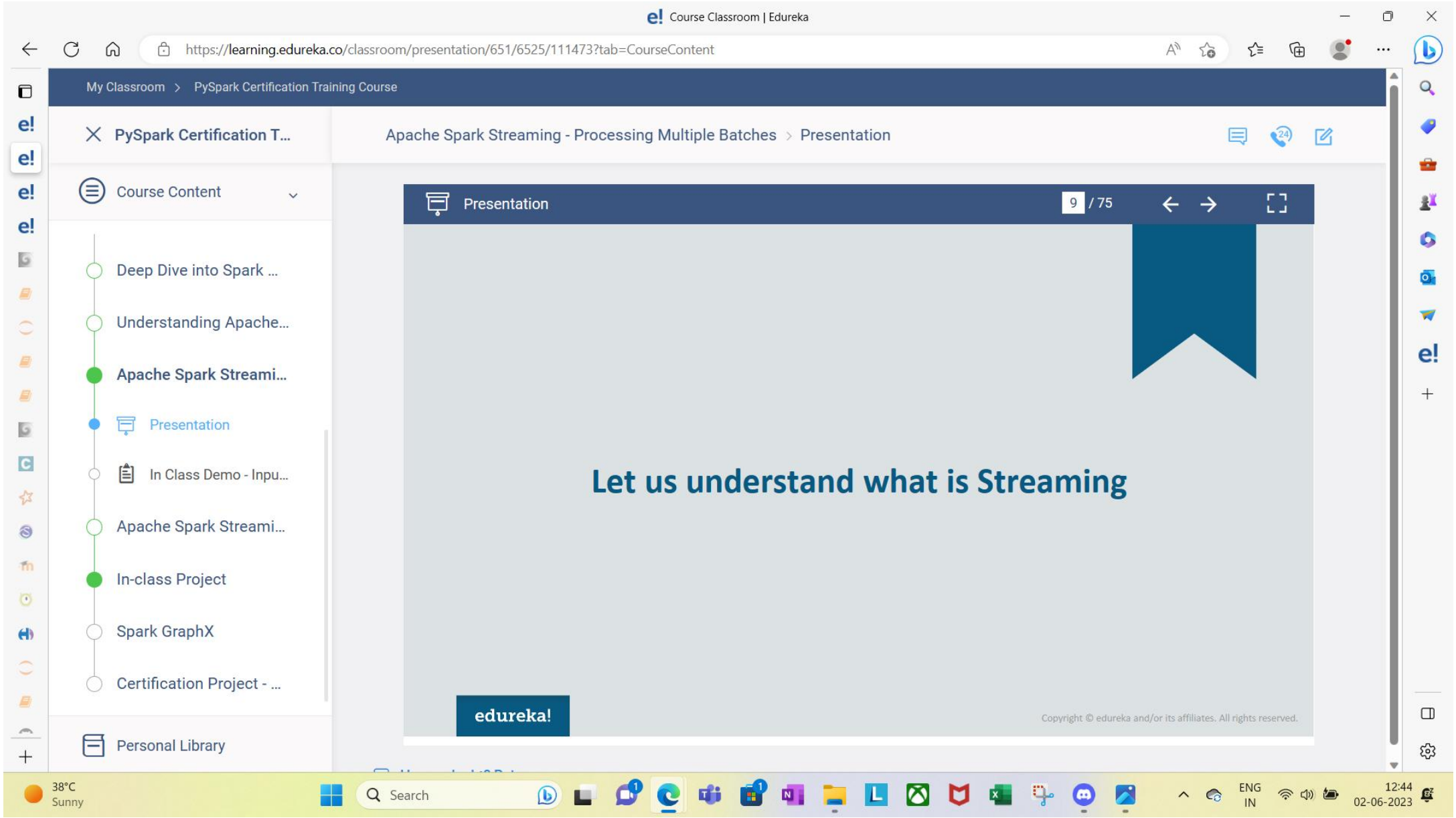
Presentation 8 / 75

Solution : Spark Streaming

Spark Streaming

Spark Streaming is used to stream real-time data from various sources like *Twitter*, *Stock Market* and *Geographical Systems* and perform *powerful analytics* to help businesses

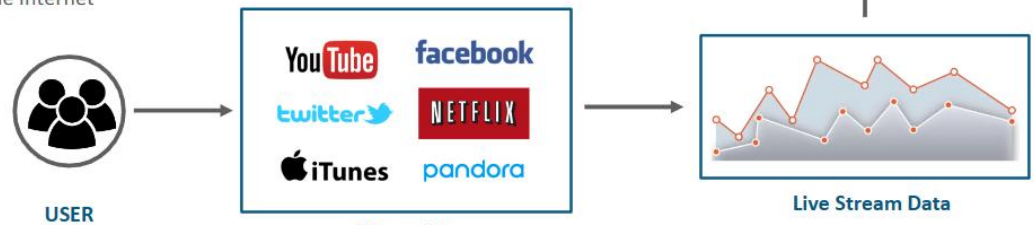
edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
9 / 75
Let us understand what is Streaming
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:44
02-06-2023

Presentation 10 / 75

What is Streaming?

- ***Data Streaming*** is a technique for transferring data so that it can be processed as a steady and continuous stream
- A ***data stream*** is an unbounded sequence of data arriving continuously
- Streaming technologies are becoming increasingly important with the growth of the Internet



edureka!

Presentation 11 / 75

Spark Streaming Overview

- ***Spark Streaming*** is used for processing ***real-time streaming data***
- It is a useful ***addition*** to the ***core Spark API***
- Spark Streaming enables ***high-throughput*** and ***fault-tolerant*** stream processing of live data streams
- The fundamental stream unit is ***DStream*** which is basically a series of RDDs to process the real-time data

Spark Streaming

edureka!

Presentation 12 / 75

Spark Streaming Overview : Representation

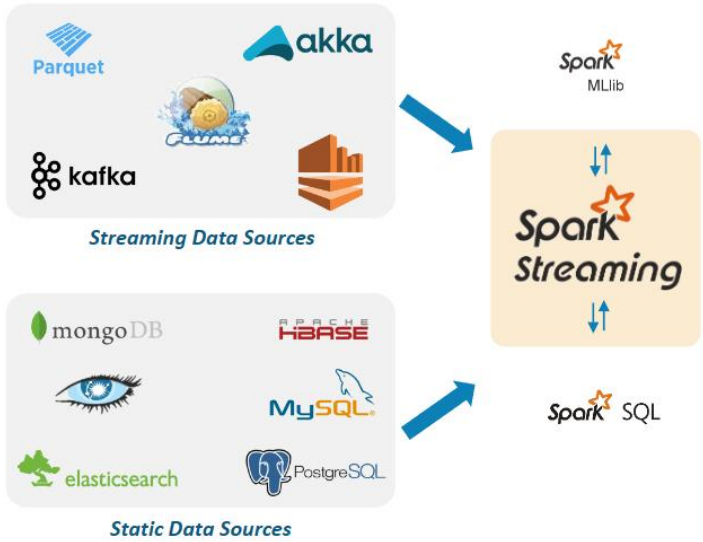
Streaming Data Sources

Static Data Sources

edureka!

Presentation 13 / 75

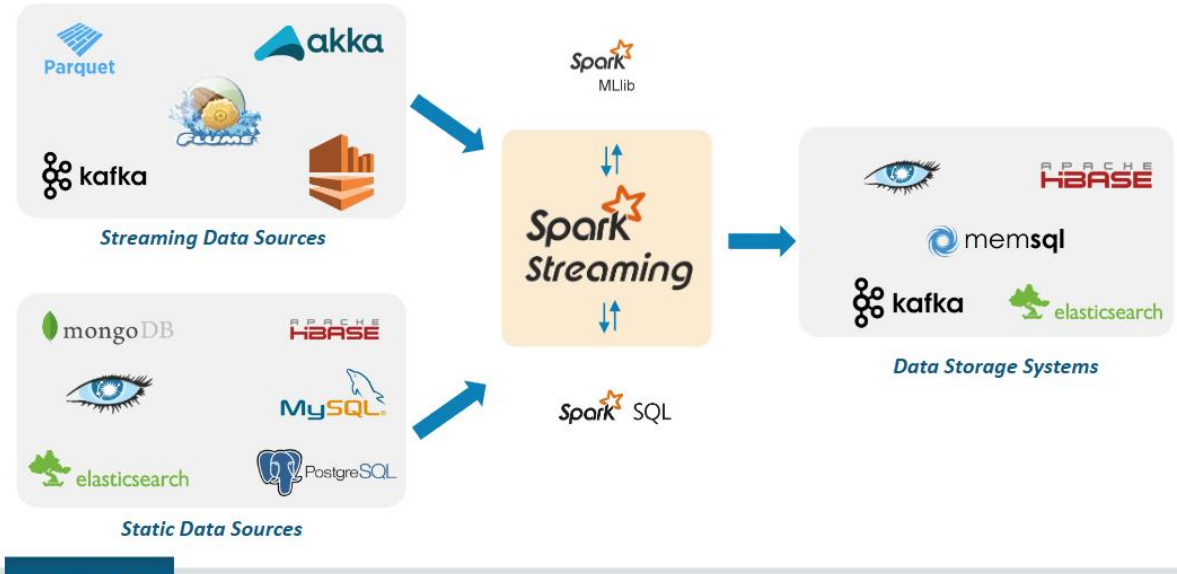
Spark Streaming Overview : Representation



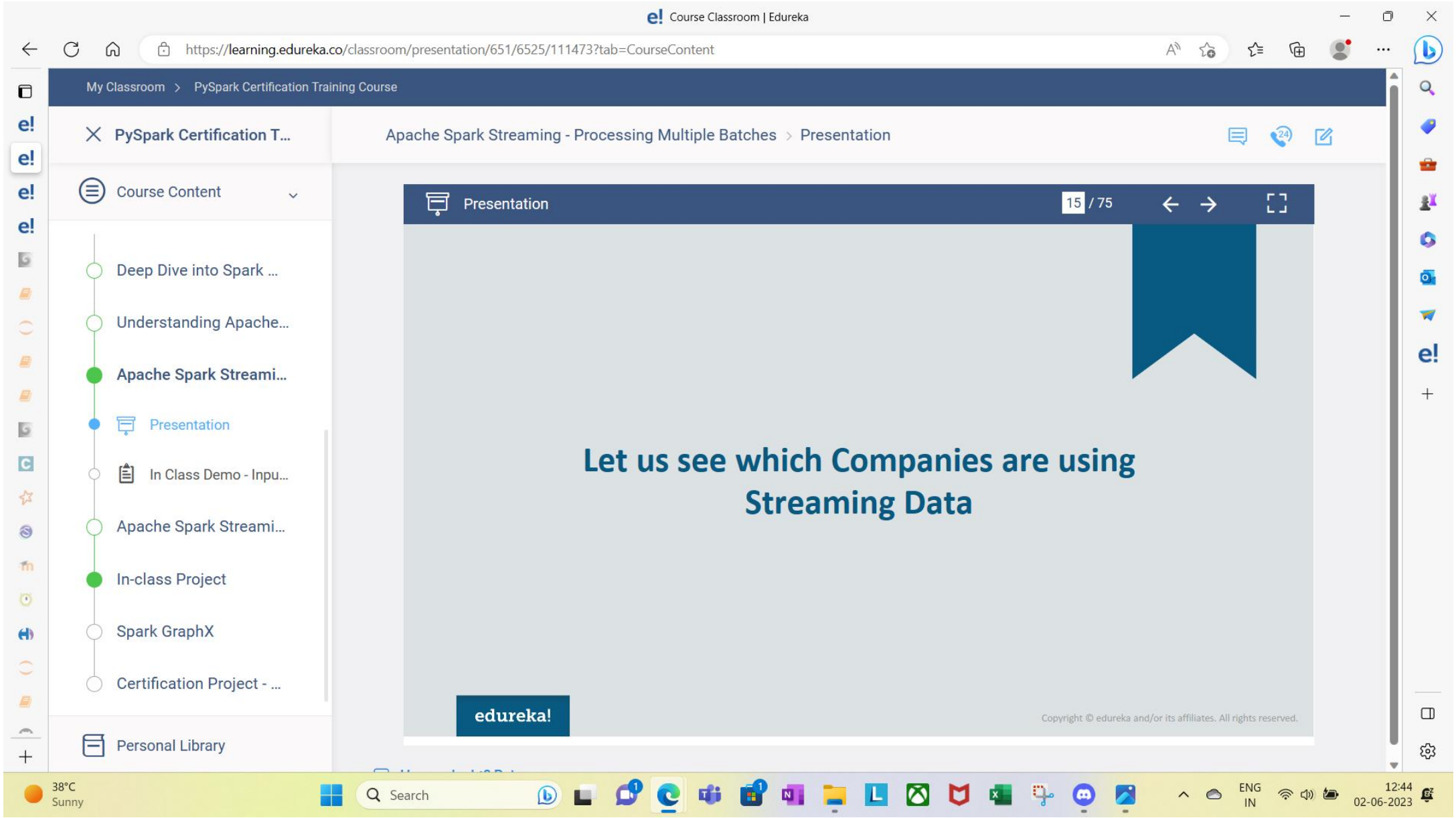
edureka!

Presentation 14 / 75

Spark Streaming Overview : Representation

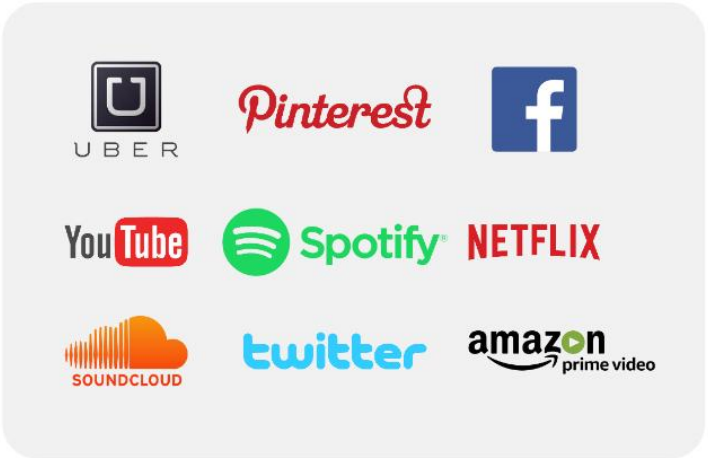


edureka!

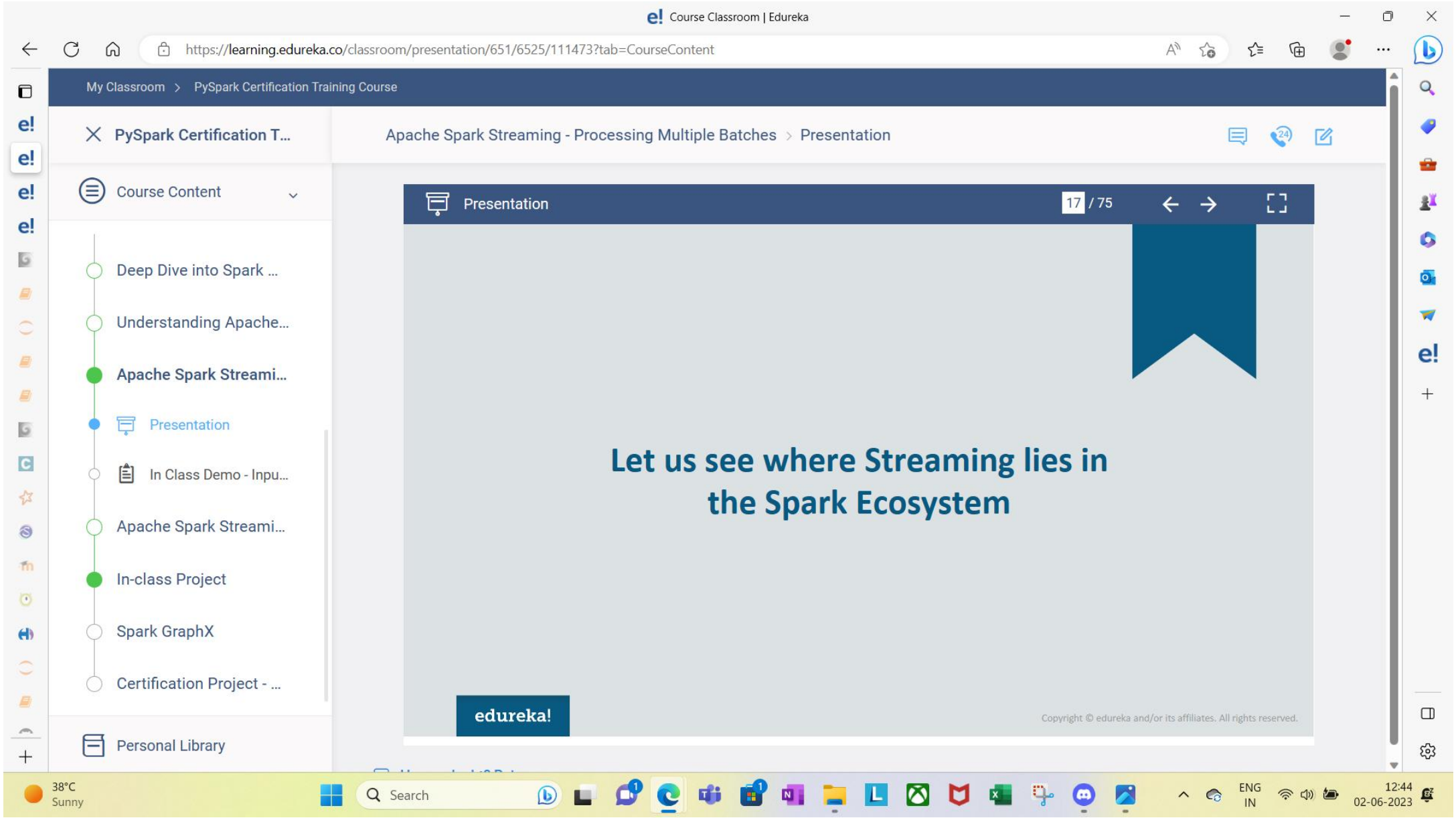

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
15 / 75
Let us see which Companies are using Streaming Data
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:44
02-06-2023

Presentation 16 / 75

Companies using Streaming Data

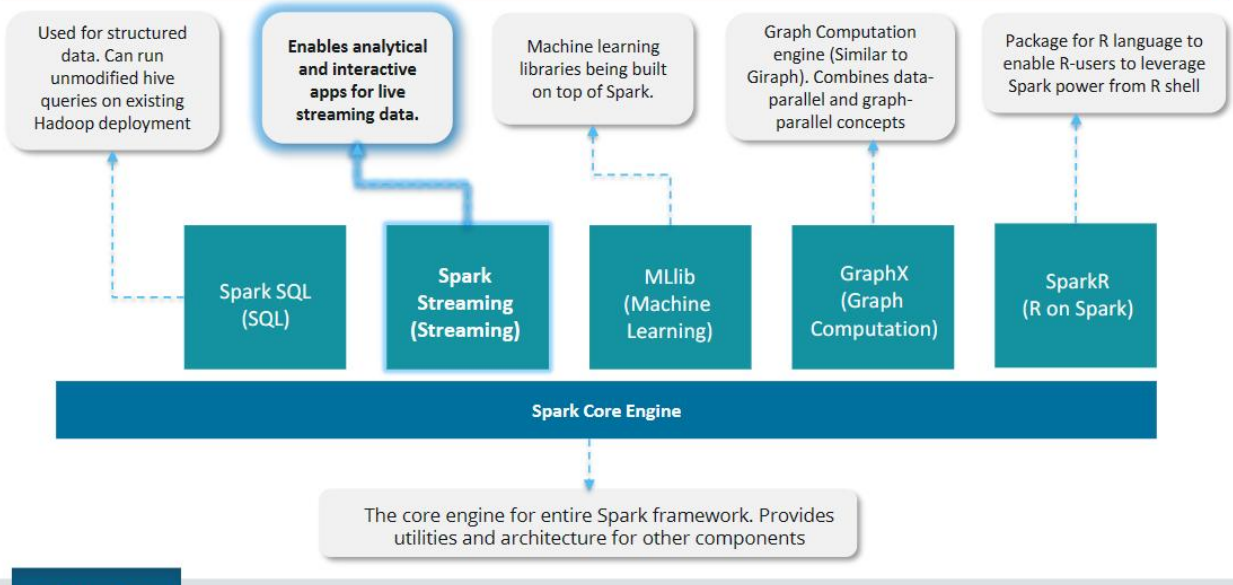


edureka!

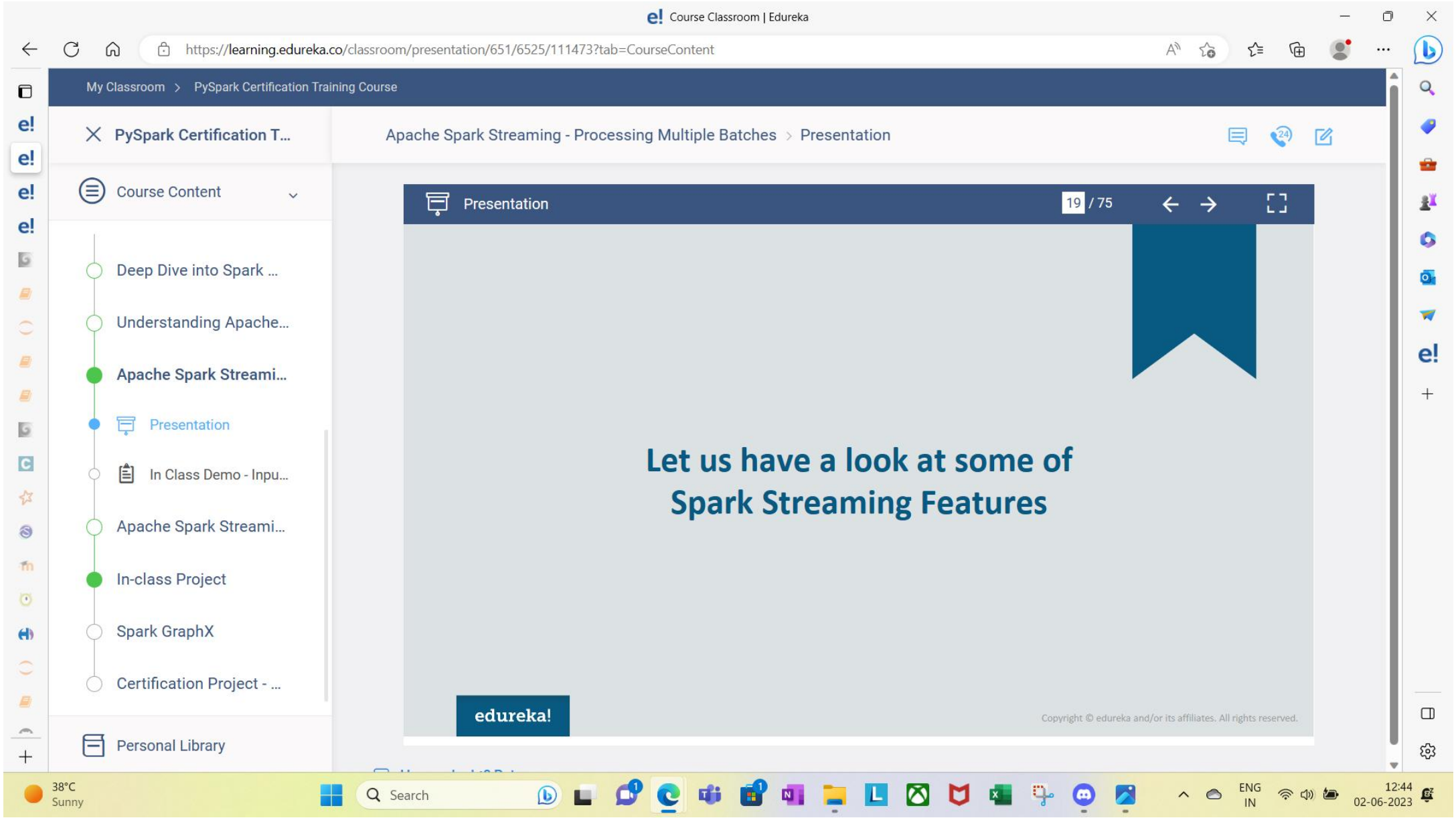

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
17 / 75
Let us see where Streaming lies in the Spark Ecosystem
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:44
02-06-2023

Presentation 18 / 75

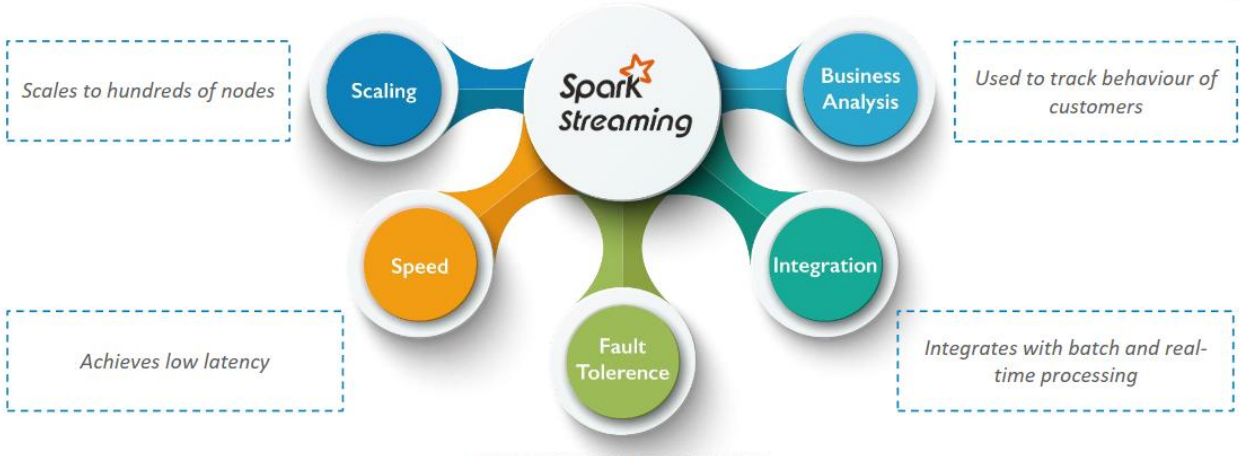
Spark Streaming in Spark Ecosystem



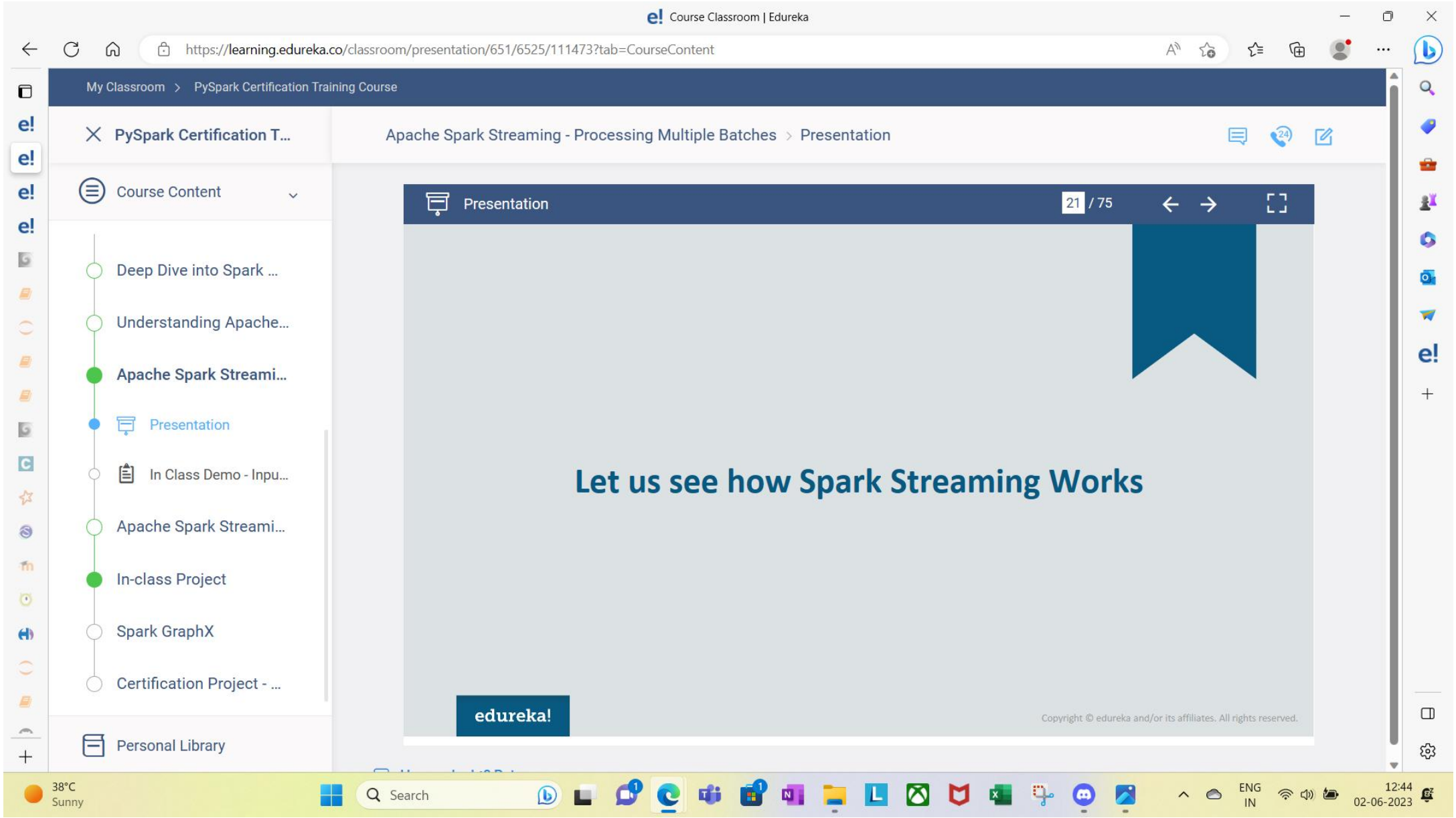
edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
19 / 75
Let us have a look at some of Spark Streaming Features
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:44
02-06-2023

Spark Streaming Features



edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
21 / 75
Let us see how Spark Streaming Works
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:44
02-06-2023

Presentation 22 / 75

Spark Streaming Workflow

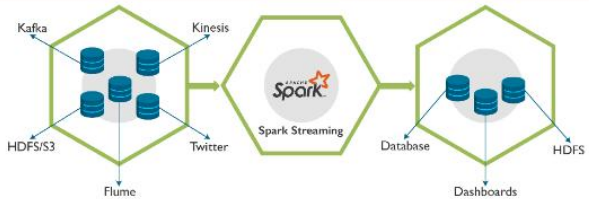


Figure: Data from a variety of sources to various storage systems

edureka!

Presentation 23 / 75

Spark Streaming Workflow

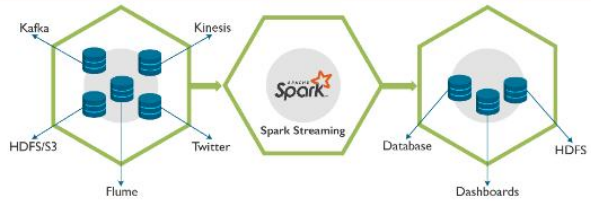


Figure: Data from a variety of sources to various storage systems

Figure: Incoming streams of data divided into batches

edureka!

Presentation 24 / 75

Spark Streaming Workflow

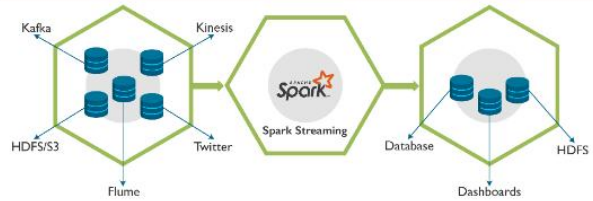


Figure: Data from a variety of sources to various storage systems

Figure: Incoming streams of data divided into batches

Figure: Input data stream divided into discrete chunks of data

edureka!

Presentation 25 / 75

Spark Streaming Workflow

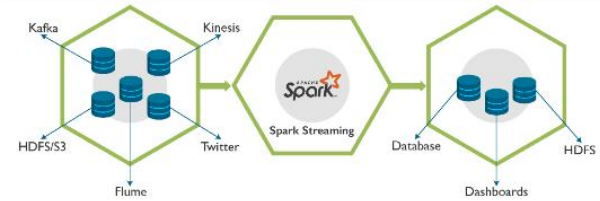


Figure: Data from a variety of sources to various storage systems

Figure: Incoming streams of data divided into batches

Figure: Input data stream divided into discrete chunks of data

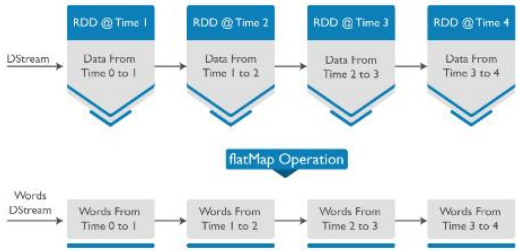
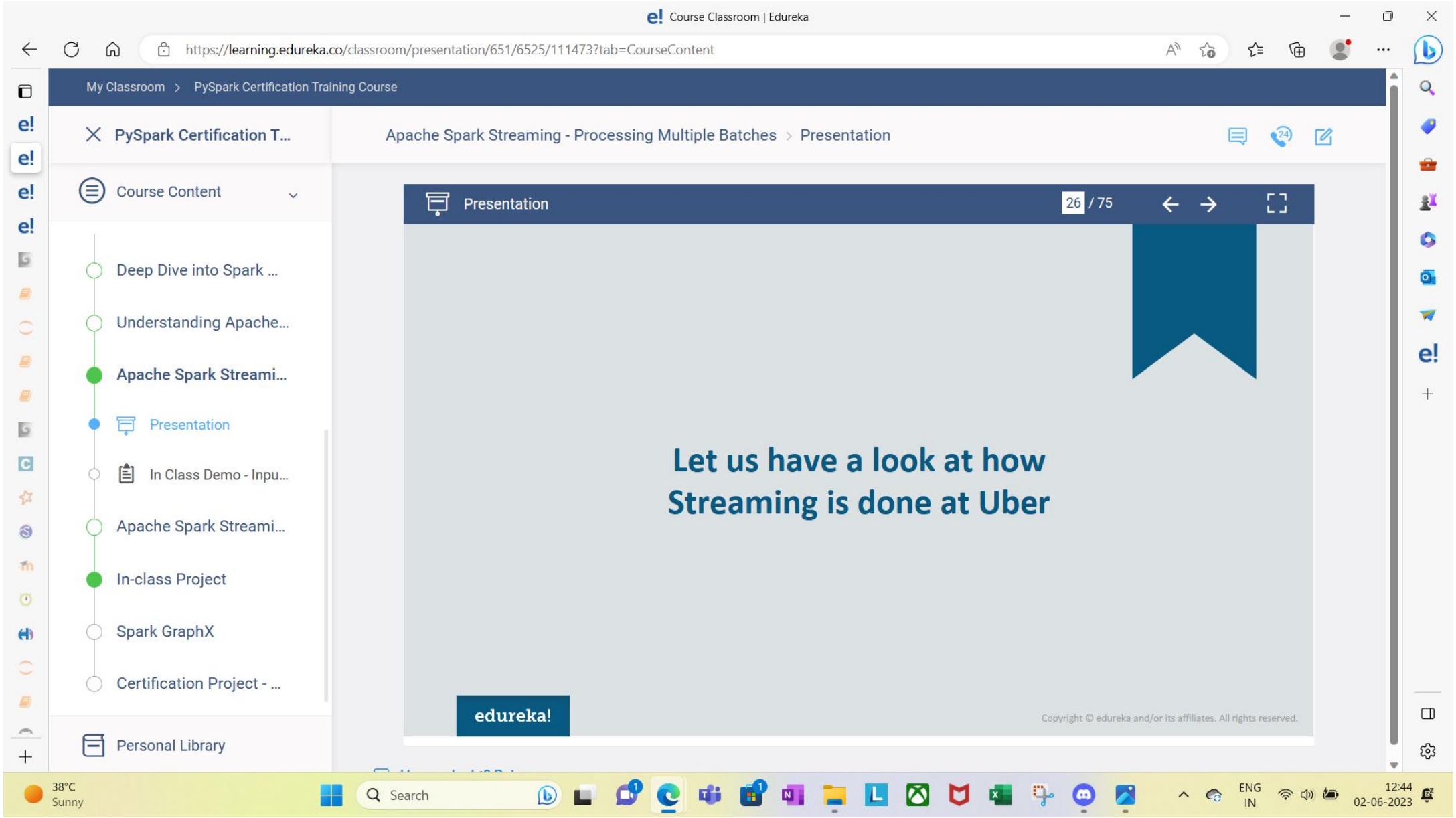


Figure: Extracting words from an Input Stream

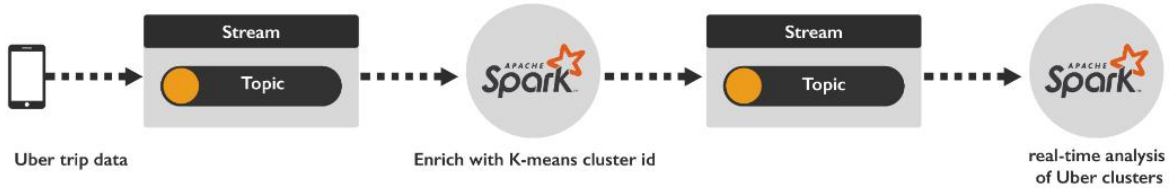
edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
26 / 75
Let us have a look at how Streaming is done at Uber
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:44
02-06-2023

Presentation 27 / 75

Uber Streaming : Process

- Uber trip data is published to a MapR Streams topic using the Kafka API
- A Spark streaming application subscribed to the first topic:
  - *Ingests a stream of uber trip events*
  - *Identifies the location cluster corresponding to the latitude and longitude of the uber trip*
  - *Adds the cluster location to the event and publishes the results in JSON format to another topic*
- A Spark streaming application subscribed to the second topic:
  - *Analyses the uber trip location clusters that are popular by date and time*



edureka!

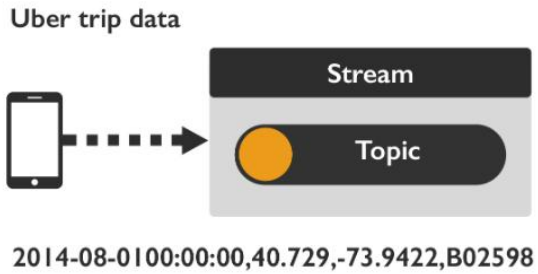
Presentation 28 / 75

Getting Data

- The example data set is Uber trip data
- The incoming data is in CSV format, an example is shown below , with the header:

date/time, latitude, longitude, base

2014-08-01 00:00:00,40.729,-73.9422,B02598

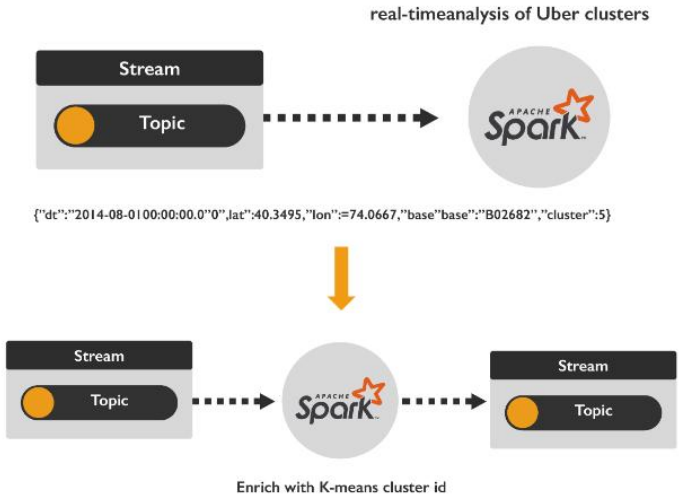


edureka!

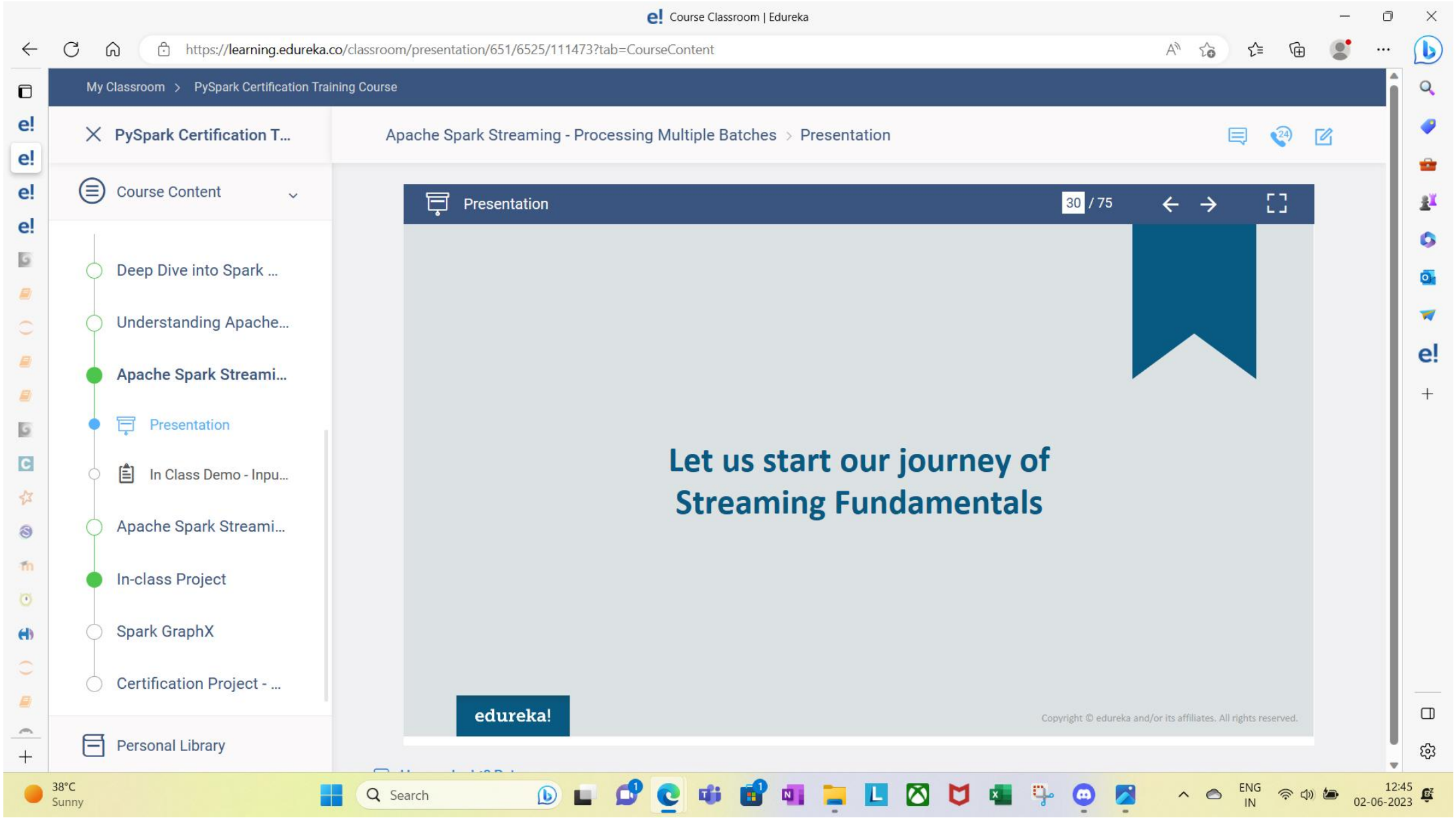
Presentation 29 / 75

Workflow

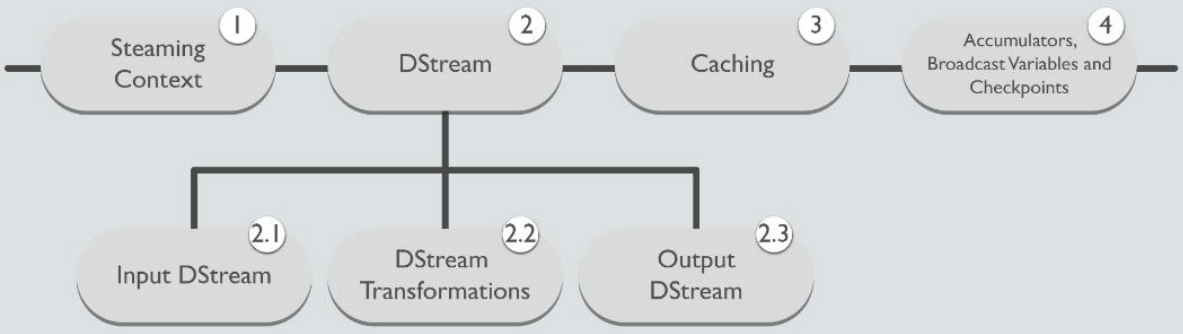
The enriched Data Records are in JSON format. An example line is shown below :



edureka!

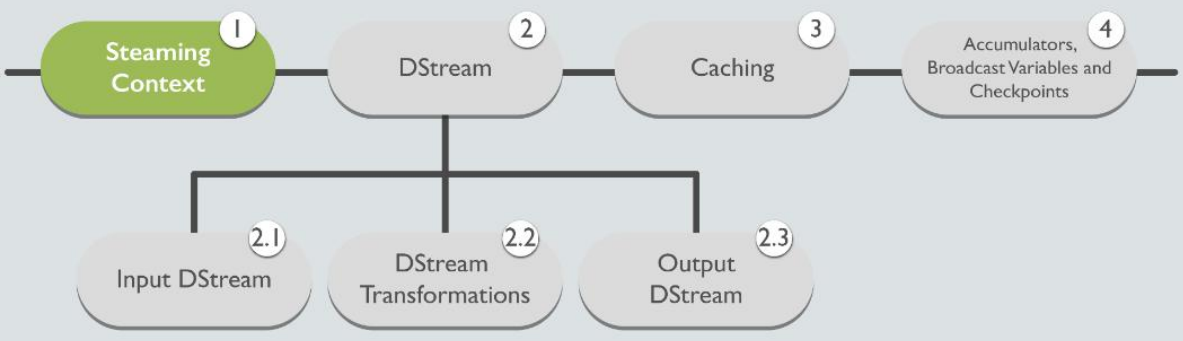

Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
30 / 75
Let us start our journey of Streaming Fundamentals
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C Sunny
Search
ENG IN
12:45 02-06-2023

Presentation 31 / 75



edureka!

Presentation 32 / 75



edureka!

Streaming Context

- It is the main ***entry point*** for Spark functionality
- Spark provides a number of default implementations of sources like ***Twitter, Akka Actor*** and ***ZeroMQ*** that are accessible from the context
- Whatever we do in Spark Streaming has to start from creating an instance of ***StreamingContext***

- Consumes a stream of data in Spark
- Registers an ***InputDStream*** to produce a ***Receiver*** object
- It provides methods used to create DStreams from various input sources

edureka!

Streaming Context : Initialization

- Streaming Context can be created in 3 ways:
  - By providing a *Spark master URL* and *an appName*, or
  - from a *org.apache.spark.SparkConf* configuration, or
  - from an *existing org.apache.spark.SparkContext*
- The associated SparkContext can be accessed using *context.sparkContext*. After creating and transforming DStreams, the streaming computation can be started and stopped using *context.start()* and *context.stop()*, respectively
- Let's see a simple initialization of StreamingContext :

```
#    Spark
from pyspark import SparkContext
#    Spark Streaming
from pyspark.streaming import StreamingContext
sc = SparkContext(appName="PythonSparkStreaming")
ssc = StreamingContext(sc, 60)
```

edureka!

Presentation 35 / 75

Streaming Context : Creation : Steps

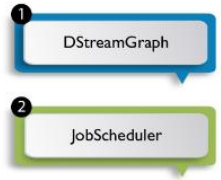
1 A ***DStreamGraph*** is created

edureka!

Presentation 36 / 75

Streaming Context : Creation : Steps

1. A ***DStreamGraph*** is created
2. A ***JobScheduler*** is created



edureka!

Presentation 37 / 75

Streaming Context : Creation : Steps

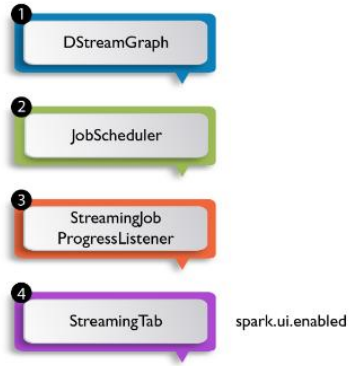
1. A ***DStreamGraph*** is created
2. A ***JobScheduler*** is created
3. A ***StreamingJobProgressListener*** is created

edureka!

Presentation 38 / 75

Streaming Context : Creation : Steps

1. A ***DStreamGraph*** is created
2. A ***JobScheduler*** is created
3. A ***StreamingJobProgressListener*** is created
4. Streaming tab in web UI is created (when ***spark.ui.enabled*** is enabled)

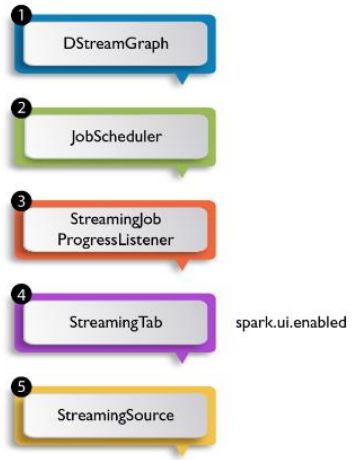


edureka!

Presentation 39 / 75

Streaming Context : Creation : Steps

1. A ***DStreamGraph*** is created
2. A ***JobScheduler*** is created
3. A ***StreamingJobProgressListener*** is created
4. Streaming tab in web UI is created (when ***spark.ui.enabled*** is enabled)
5. A ***StreamingSource*** is instantiated

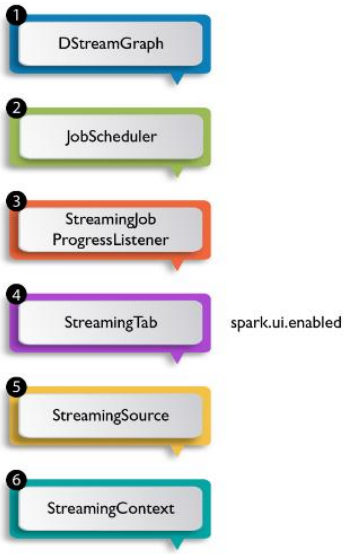


edureka!

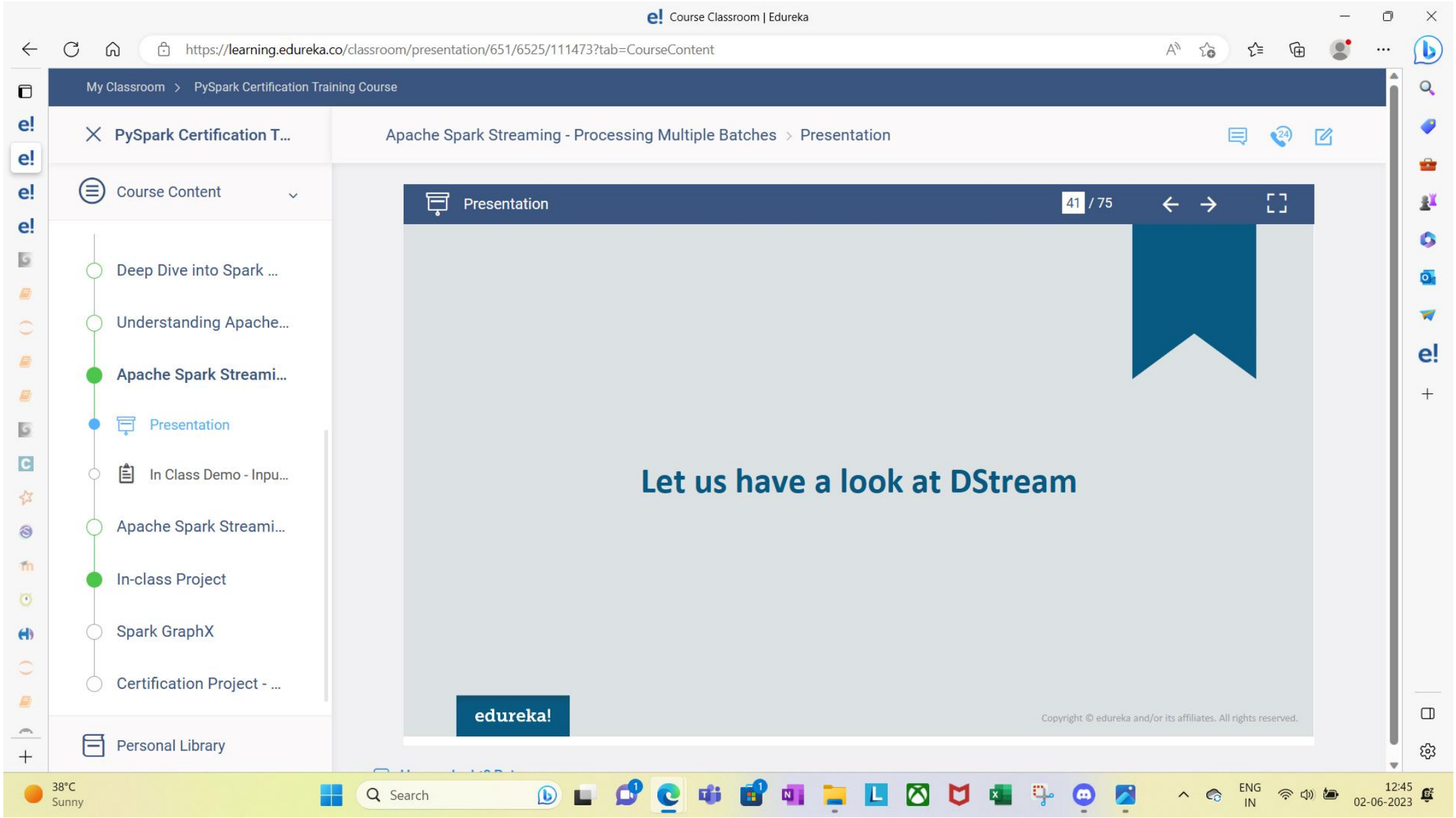
Presentation 40 / 75

Streaming Context : Creation : Steps

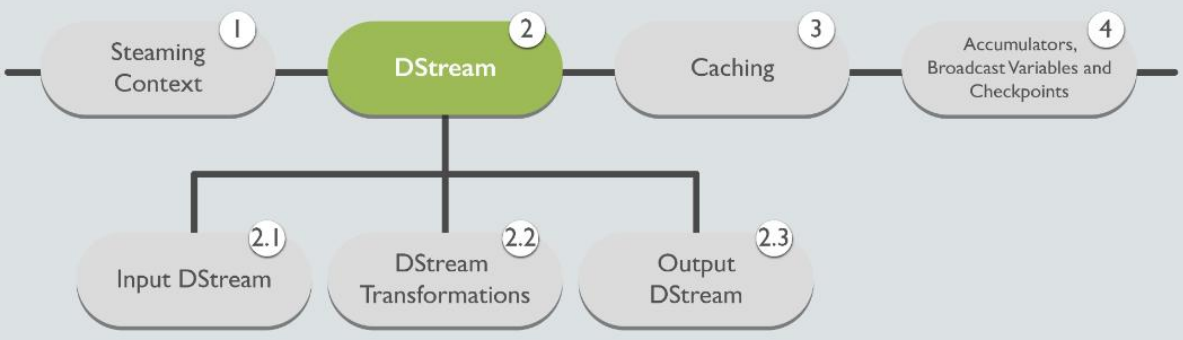
1. A ***DStreamGraph*** is created
2. A ***JobScheduler*** is created
3. A ***StreamingJobProgressListener*** is created
4. Streaming tab in web UI is created (when ***spark.ui.enabled*** is enabled)
5. A ***StreamingSource*** is instantiated
6. At this point, StreamingContext enters ***INITIALIZED*** state



edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
41 / 75
Let us have a look at DStream
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:45
02-06-2023

Presentation 42 / 75



edureka!

Presentation 43 / 75

DStream

- ***Discretized Stream*** (DStream) is the basic abstraction provided by Spark Streaming
- It is a ***continuous stream*** of data

Figure: Input data stream divided into discrete chunks of data

- It is received from ***source*** or from a ***processed data stream*** generated by transforming the input stream
- Internally, a DStream is represented by a continuous ***series of RDDs*** and each ***RDD*** contains data from a ***certain interval***

edureka!

Presentation 44 / 75

DStream Operation

- Any operation applied on a DStream translates to operations on the underlying RDDs
- For example, in the example of converting a stream of lines to words, the ***flatMap*** operation is applied on each RDD in the lines DStream to generate the RDDs of the words ***DStream***

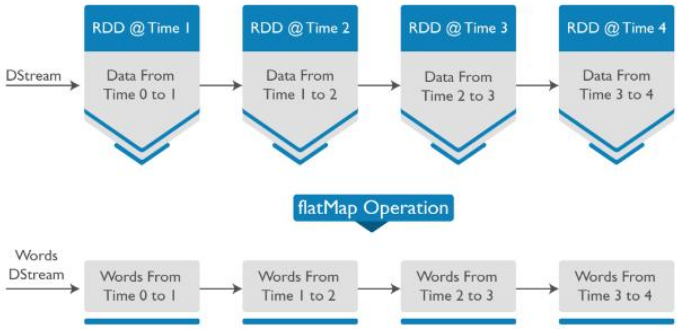
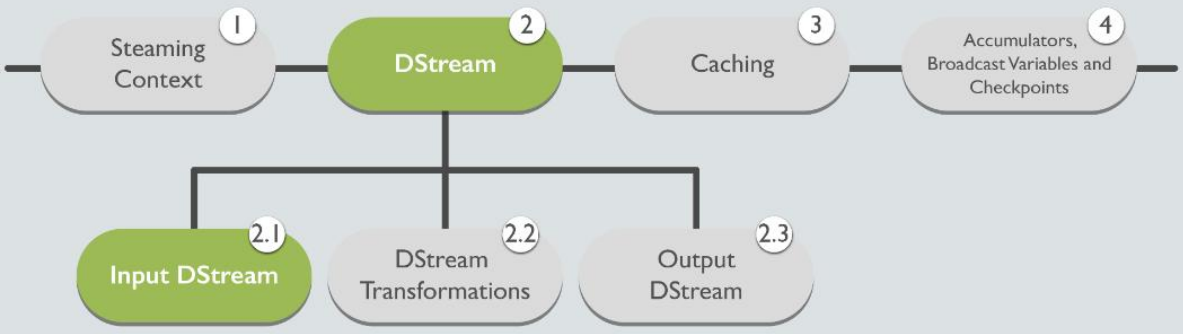


Figure: Extracting words from an Input Stream

edureka!

Presentation 45 / 75

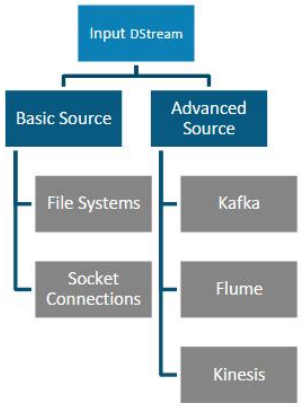


edureka!

Presentation 46 / 75

Input DStreams

- ***Input DStreams*** are DStreams representing the stream of ***input data*** received from streaming ***sources***



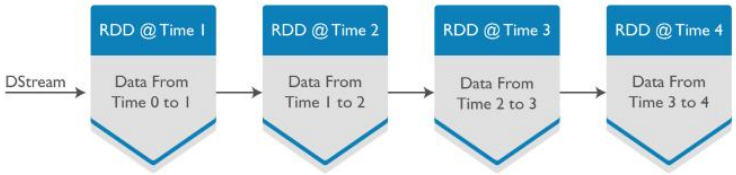


Figure: Input data stream divided into discrete chunks of data

edureka!

Presentation 47 / 75

Receiver

- Every input DStream is associated with a **Receiver** object which receives the data from a **source** and stores it in **Spark's memory** for processing

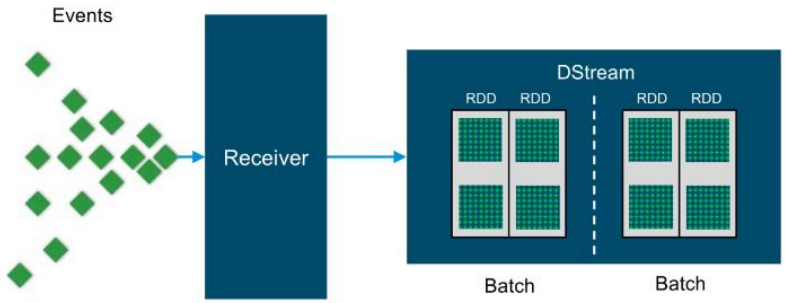
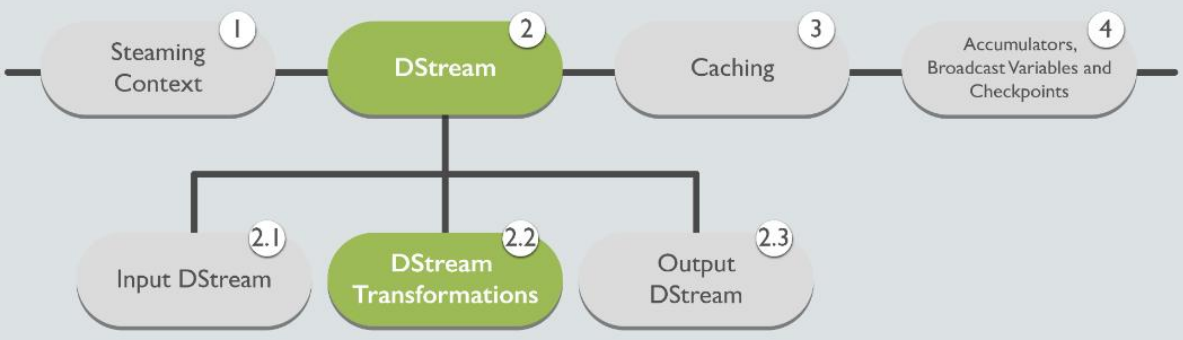


Figure: The Receiver sends data onto the DStream where each Batch contains RDDs

edureka!

Presentation 48 / 75



edureka!

Presentation 49 / 75

Transformations on DStreams

Most Popular Spark Streaming Transformations

1. map
2. flatMap
3. filter
4. reduce
5. groupBy

- **Transformations** allow the data from the **input DStream** to be **modified** similar to RDDs
- DStreams support many of the transformations available on normal Spark RDDs

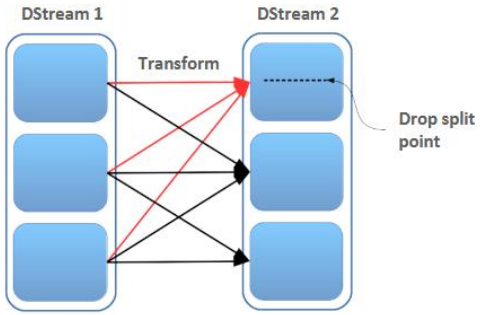


Figure: *DStream Transformations*

edureka!

Presentation 50 / 75

Transformations on DStreams

1. map
2. flatMap
3. filter
4. reduce
5. groupBy

- **map(*func*)**
- **map(*func*)** returns a new **DStream** by passing each element of the source DStream through a function ***func***

Figure: Input DStream being converted through map(func)

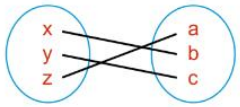


Figure: Map Function

edureka!

Presentation 51 / 75

Transformations on DStreams

1. map
2. flatMap
3. filter
4. reduce
5. groupBy

- **flatMap(*func*)**
- **flatMap(*func*)** is similar to **map(*func*)** but each input item can be mapped to **0 or more** output items and returns a new **DStream** by passing each source element through a function ***func***

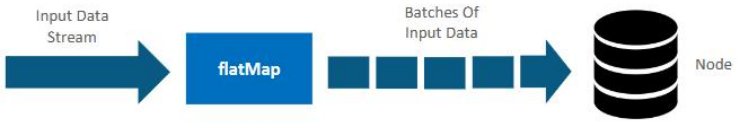


Figure: Input DStream being converted through flatMap(func)

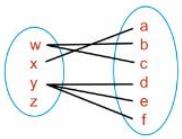


Figure: flatMap Function

edureka!

Presentation 52 / 75

Transformations on DStreams

1. map
2. flatMap
3. filter
4. reduce
5. groupBy

- **filter(*func*)**
- **filter(*func*)** returns a new **DStream** by **selecting** only the records of the source DStream on which ***func*** returns true

Figure: Input DStream being converted through filter(func)

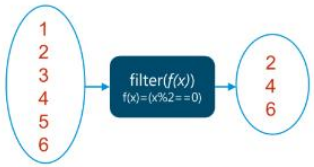


Figure: Filter Function For Even Numbers

edureka!

Presentation 53 / 75

Transformations on DStreams

1. map
2. flatMap
3. filter
4. reduce
5. groupBy

- **reduce(*func*)**
- **reduce(*func*)** returns a new DStream of **single-element RDDs** by **aggregating** the elements in each RDD of the source DStream using a function ***func***

Figure: Input DStream being converted through reduce(func)

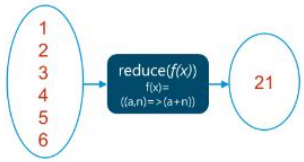


Figure: Reduce Function To Get Cumulative Sum

edureka!

Presentation 54 / 75

Transformations on DStreams

1. map
2. flatMap
3. filter
4. reduce
5. groupBy

- **groupBy(*func*)**
- **groupBy(*func*)** returns the new RDD which basically is made up with a **key** and **corresponding list** of items of that group

Figure: Input DStream being converted through groupBy(func)

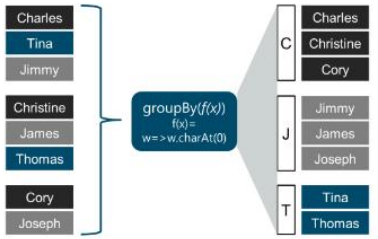
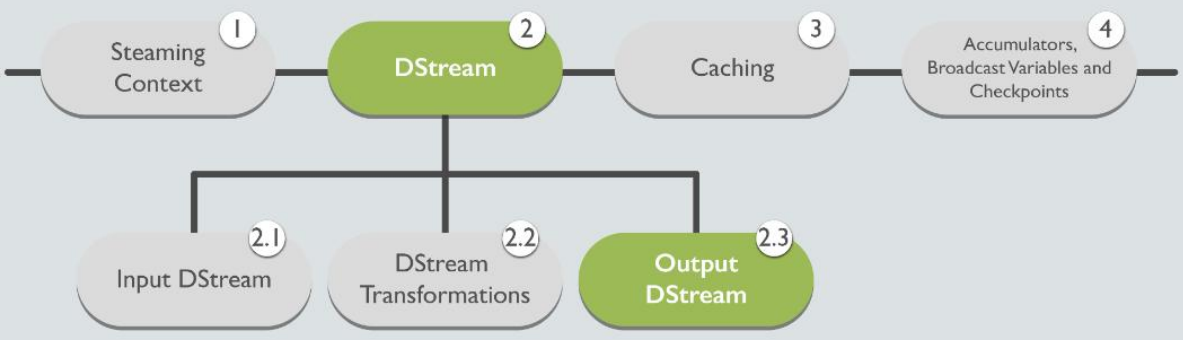


Figure: Grouping By First Letters

edureka!

Presentation 55 / 75



edureka!

Presentation 56 / 75

Output Operations on DStreams

- ***Output operations*** allow DStream's data to be pushed out to ***external systems*** like ***databases*** or ***file systems***
- Output operations ***trigger*** the ***actual execution*** of all the DStream transformations

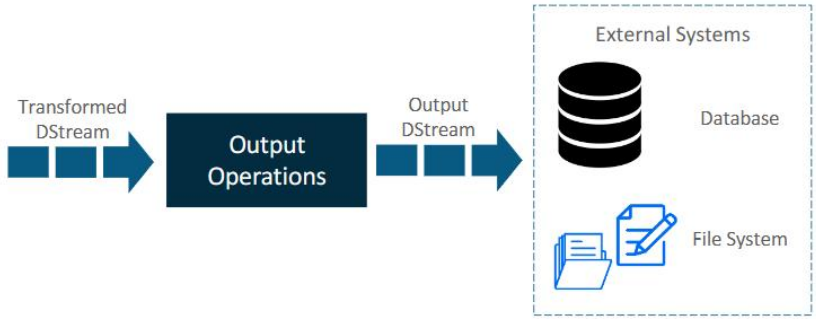


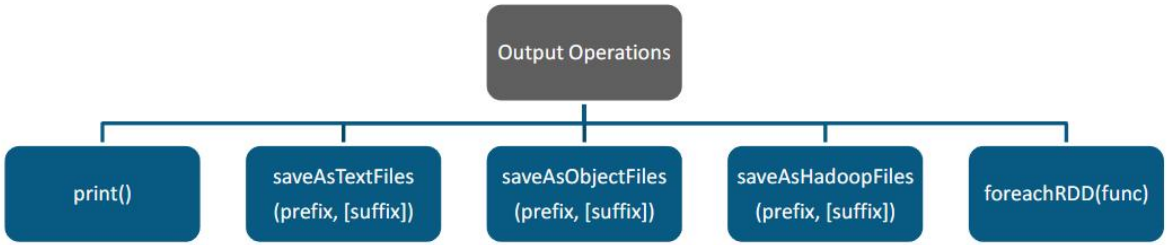
Figure: Output Operations on DStreams

edureka!

Presentation 57 / 75

Output Operations on DStreams

- Currently, the following output operations are defined:



edureka!

Presentation 58 / 75

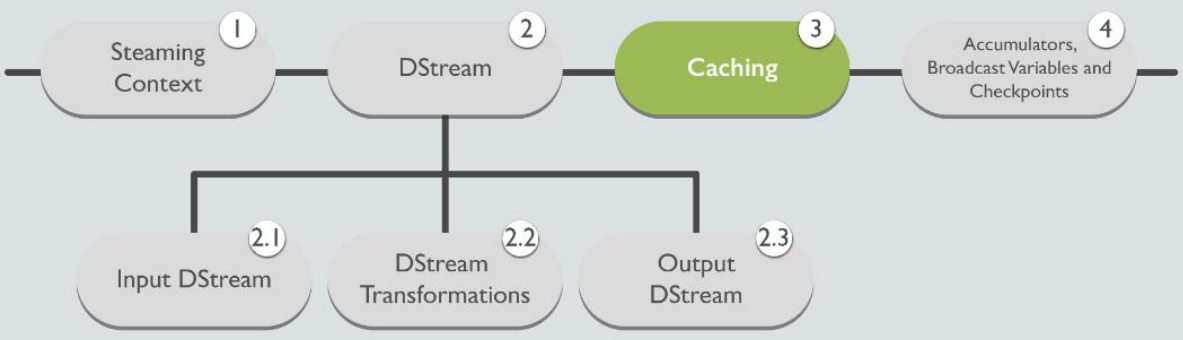
Output Operations Example - foreachRDD

- *foreachRDD*
- *dstream.foreachRDD* is a powerful primitive that allows data to be sent out to external systems
- The lazy evaluation achieves the most efficient transfer of data

```
def sendRecord(record):
        connection = createNewConnection()
        connection.send(record)
        connection.close()
dstream.foreachRDD(lambda rdd: rdd.foreach(sendRecord))
```

edureka!

Presentation 59 / 75



edureka!

Presentation 60 / 75

Caching/Persistence

- ***DStreams*** allow developers to ***cache/ persist*** the stream's data in memory
- This is useful if the data in the DStream will be computed multiple times
- This can be done using the ***persist()*** method on a DStream
- For input streams that receive data over the network (such as Kafka, Flume, Sockets, etc.), the default persistence level is set to replicate the data to two nodes for fault-tolerance

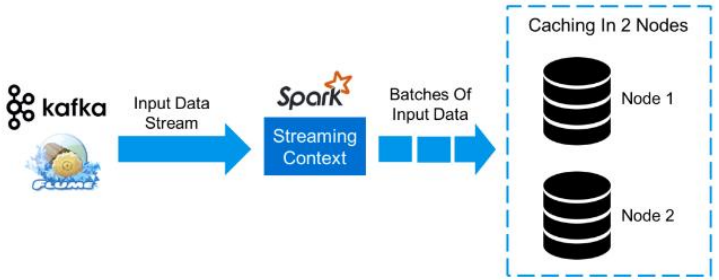
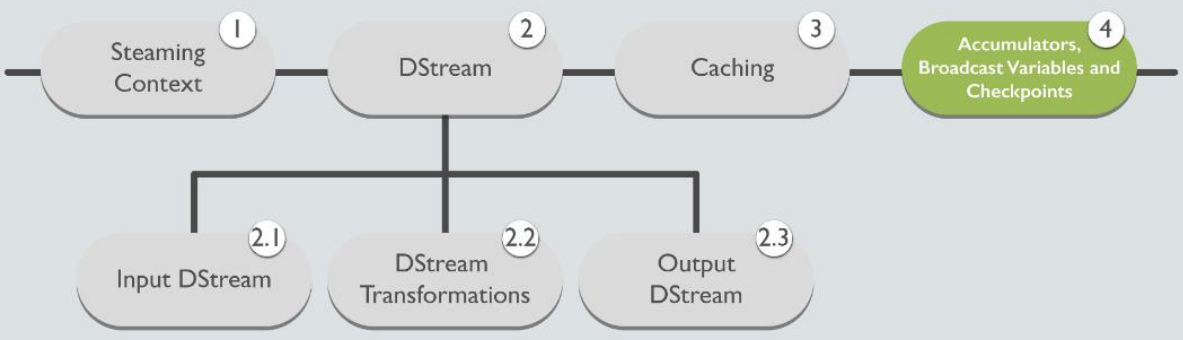


Figure: Caching Into 2 Nodes

edureka!

Presentation 61 / 75



edureka!

Presentation 62 / 75

Accumulators

- ***Accumulators*** are variables that are only added through an associative and commutative operation
- They are used to implement ***counters*** or ***sums***
- Tracking accumulators in the UI can be useful for ***understanding*** the ***progress*** of running stages
- Spark natively supports ***numeric*** accumulators. We can create ***named*** or ***unnamed*** accumulators

Accumulators

Accumulable	Value
counter	45

Tasks

Index ▲	ID	Attempt	Status	Locality Level	Executor ID / Host	Launch Time	Duration	GC Time	Accumulators	Errors
0	0	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms			
1	1	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 1	
2	2	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 2	
3	3	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 7	
4	4	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 5	
5	5	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 6	
6	6	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 7	
7	7	0	SUCCESS	PROCESS_LOCAL	driver / localhost	2016/04/21 10:10:41	17 ms		counter: 17	

edureka!

Presentation 63 / 75

Broadcast Variables

- ***Broadcast variables*** allow the programmer to keep ***a read-only variable cached*** on ***each machine*** rather than shipping a copy of it with tasks
- They can be used to give every node a ***copy*** of a ***large input dataset*** in an efficient manner
- ***Spark*** also attempts to distribute broadcast variables using efficient ***broadcast algorithms*** to reduce communication cost

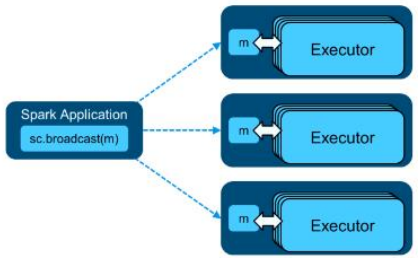


Figure: Broadcasting A Value To Executors

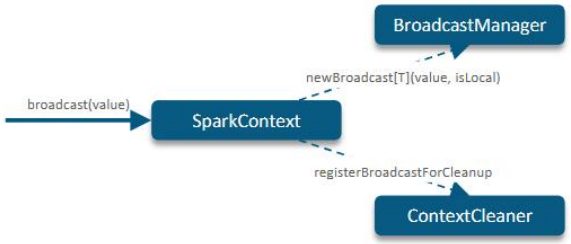


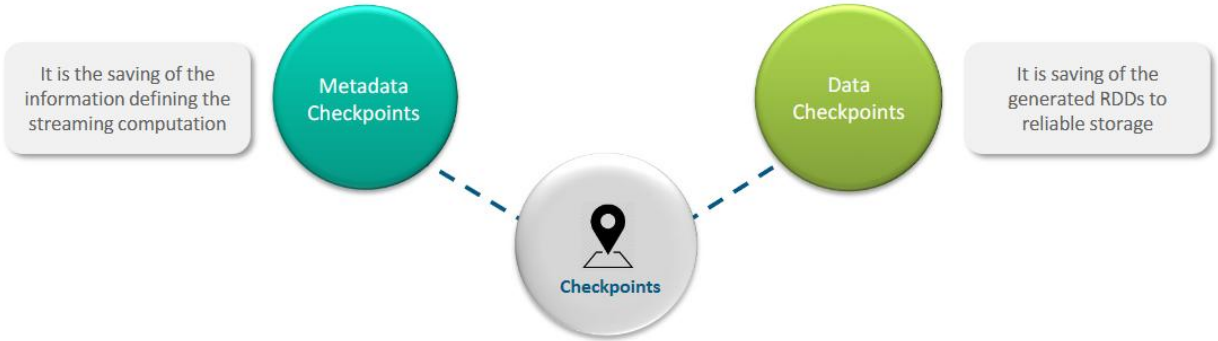
Figure: SparkContext and Broadcasting

edureka!

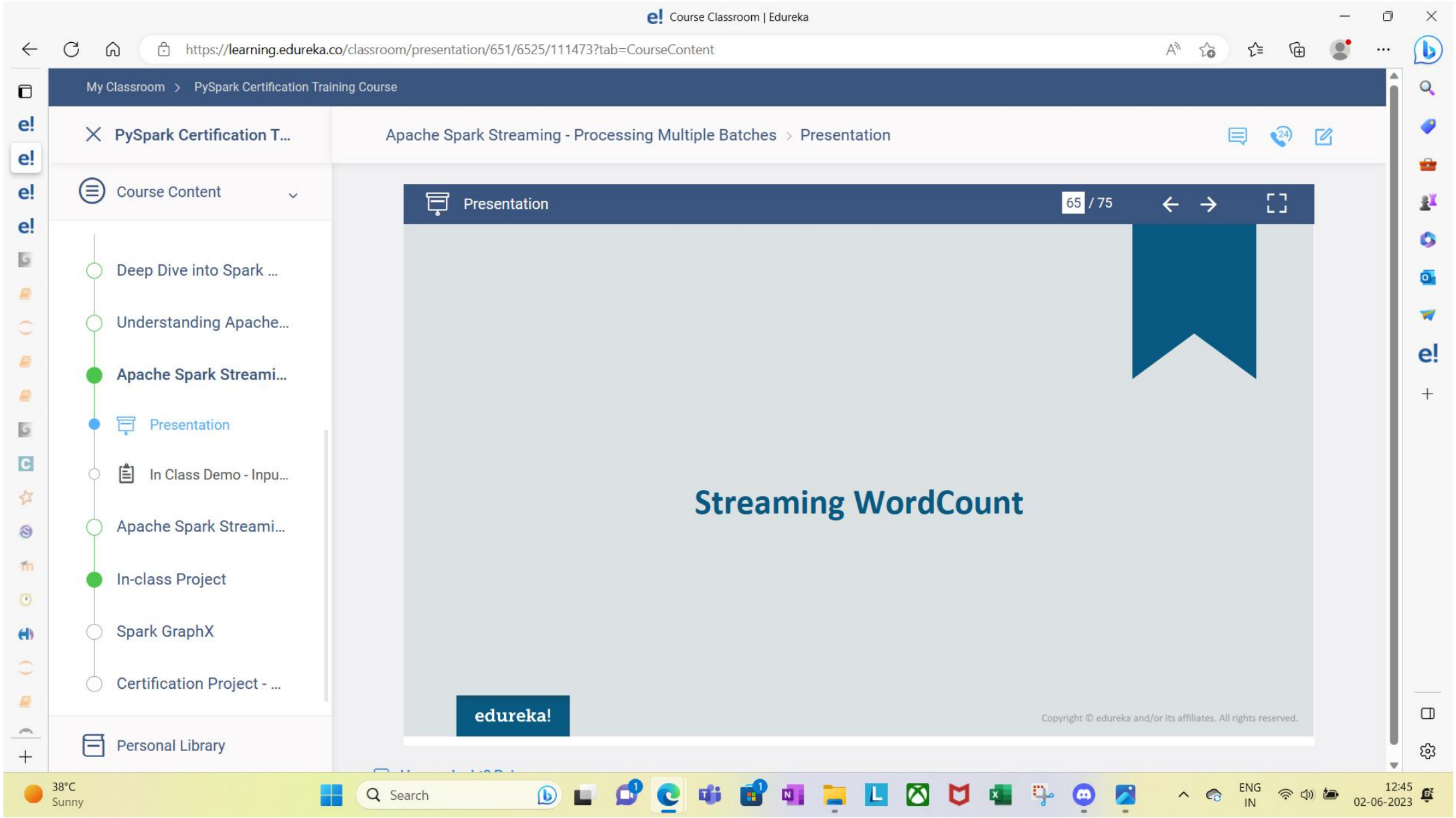
Presentation 64 / 75

Checkpoints

- ***Checkpoints*** are similar to checkpoints in ***gaming***. They make it ***run 24/7*** and make it ***resilient to failures*** unrelated to the application logic



edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
65 / 75
Streaming WordCount
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:45
02-06-2023

Presentation 66 / 75

Streaming WordCount

```
from pyspark import SparkContext
from pyspark.streaming import StreamingContext
# Create a local StreamingContext with two working thread and batch interval of 1 second
sc = SparkContext("local[2]", "NetworkWordCount")
ssc = StreamingContext(sc, 1)
# Create a DStream that will connect to hostname:port, like localhost:9999
lines = ssc.socketTextStream("localhost", 9999)
#Split each line into words
words = lines.flatMap(lambda line: line.split(" "))
```

Importing StreamingContext

Creating a local StreamingContext with two execution threads, and batch interval of 1 second

The lines DStream represents the stream of data that will be received from the data server

Each line will be split into multiple words and the stream of words is represented as the **words** DStream

edureka!

Presentation 67 / 75

Streaming WordCount

```
# Count each word in each batch
pairs = words.map(lambda word: (word, 1))

wordCounts = pairs.reduceByKey(lambda x, y: x + y) # Print the first ten elements of each RDD generated in this DStream to the console

wordCounts.pprint()

ssc.start() # Start the computation

ssc.awaitTermination() # Wait for the computation to terminate
```

To start the processing, ssc.start() command is used

The **words** DStream is to a DStream of (word, 1) pairs, which is then reduced to get the frequency of words in each batch of data

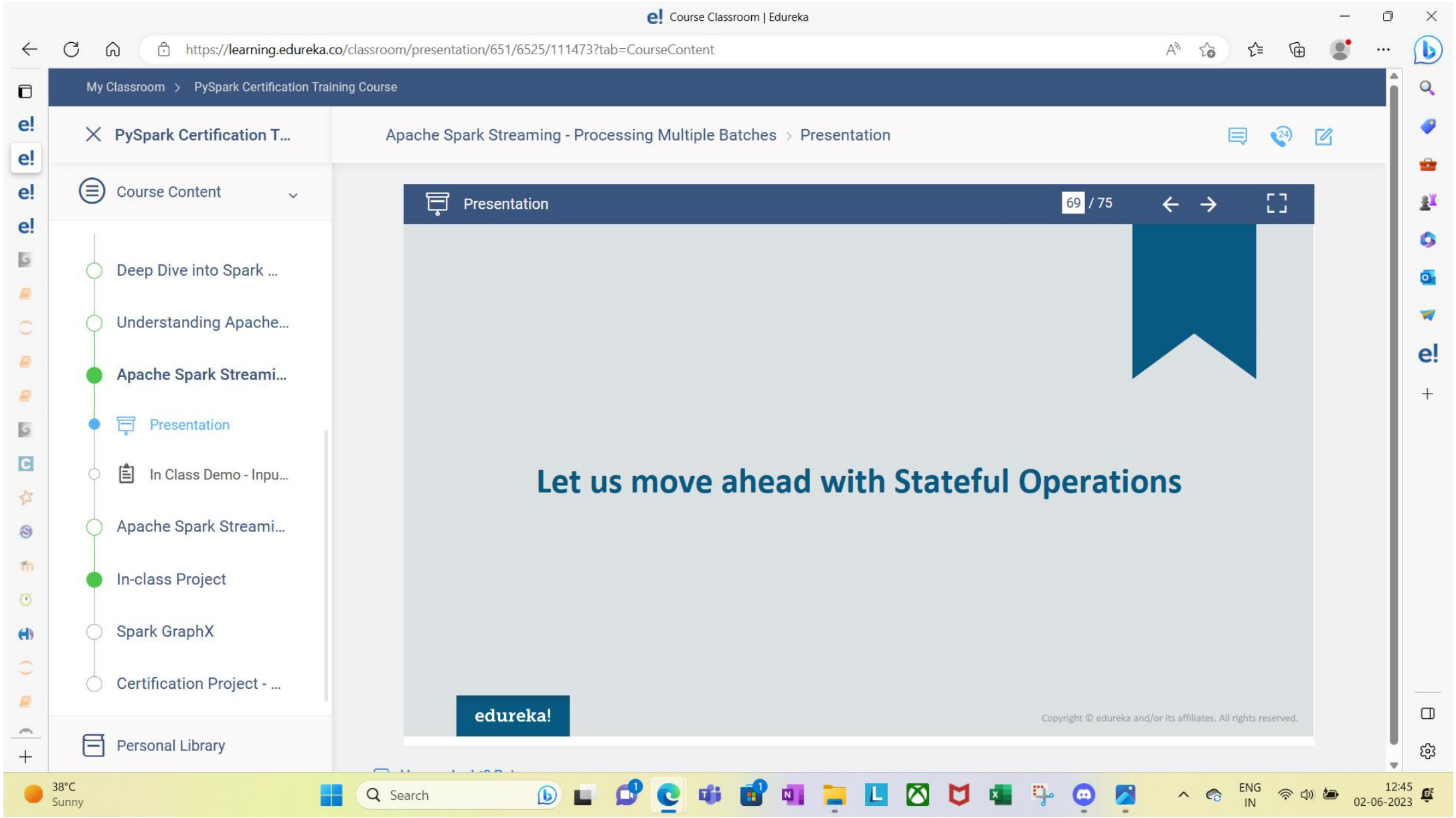
edureka!

Presentation 68 / 75

Windowed Operators

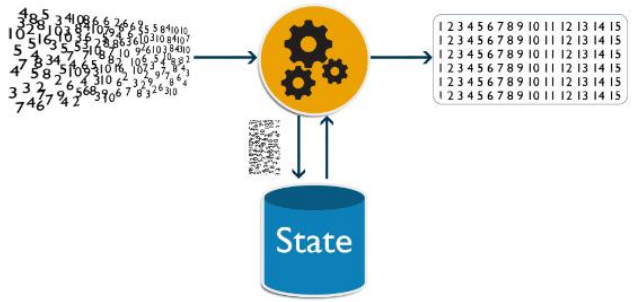
- In short, ***windowed operators*** allow you to apply transformations over a ***sliding window*** of data
- *There are mainly 3 types of operators :*

edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
69 / 75
Let us move ahead with Stateful Operations
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:45
02-06-2023

Stateful Operators

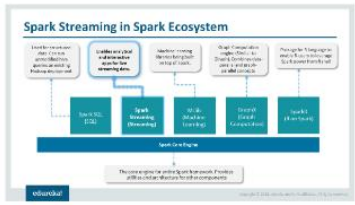
- ***Stateful operators*** (like `mapWithState` or `updateStateByKey`) are part of the set of additional operators available on DStreams of *key-value pairs*, i.e. instances of *DStream[(K, V)]*
- They allow you to build stateful stream processing pipelines and are also called cumulative calculations
- By design streaming operators are *stateless* and know nothing about the previous records and hence a state
- If we'd like to react to new records appropriately given the previous records we would have to resort to using ***persistent storages*** outside Spark Streaming
- Few Examples of Stateful Operators are:
  - ***mapWithState***
  - ***StateSpec***
  - ***updateStateByKey***

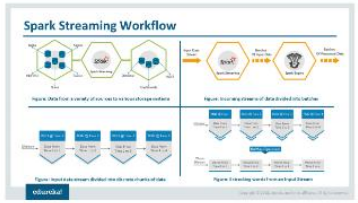


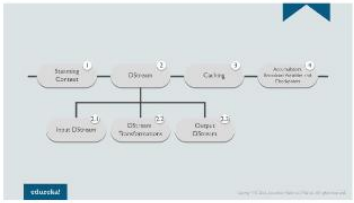
edureka!

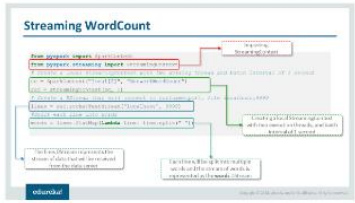
Presentation 71 / 75

Summary

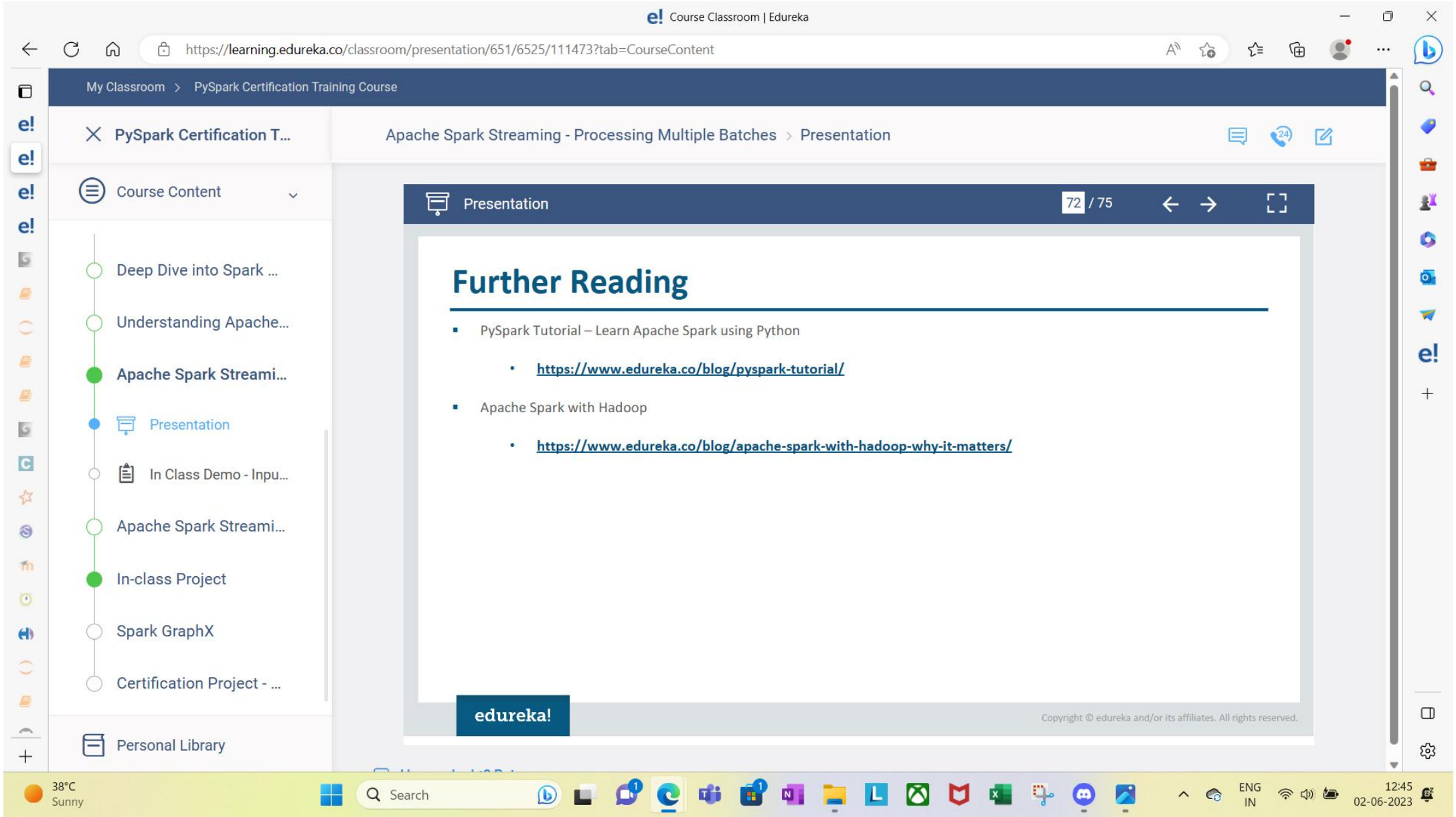








edureka!


Course Classroom | Edureka
https://learning.edureka.co/classroom/presentation/651/6525/111473?tab=CourseContent
My Classroom > PySpark Certification Training Course
PySpark Certification T...
Apache Spark Streaming - Processing Multiple Batches > Presentation
Course Content
Deep Dive into Spark ...
Understanding Apache...
Apache Spark Streami...
Presentation
In Class Demo - Inpu...
Apache Spark Streami...
In-class Project
Spark GraphX
Certification Project - ...
Personal Library
Presentation
72 / 75
Further Reading
PySpark Tutorial – Learn Apache Spark using Python
https://www.edureka.co/blog/pyspark-tutorial/
Apache Spark with Hadoop
https://www.edureka.co/blog/apache-spark-with-hadoop-why-it-matters/
edureka!
Copyright © edureka and/or its affiliates. All rights reserved.
38°C
Sunny
Search
ENG
IN
12:45
02-06-2023