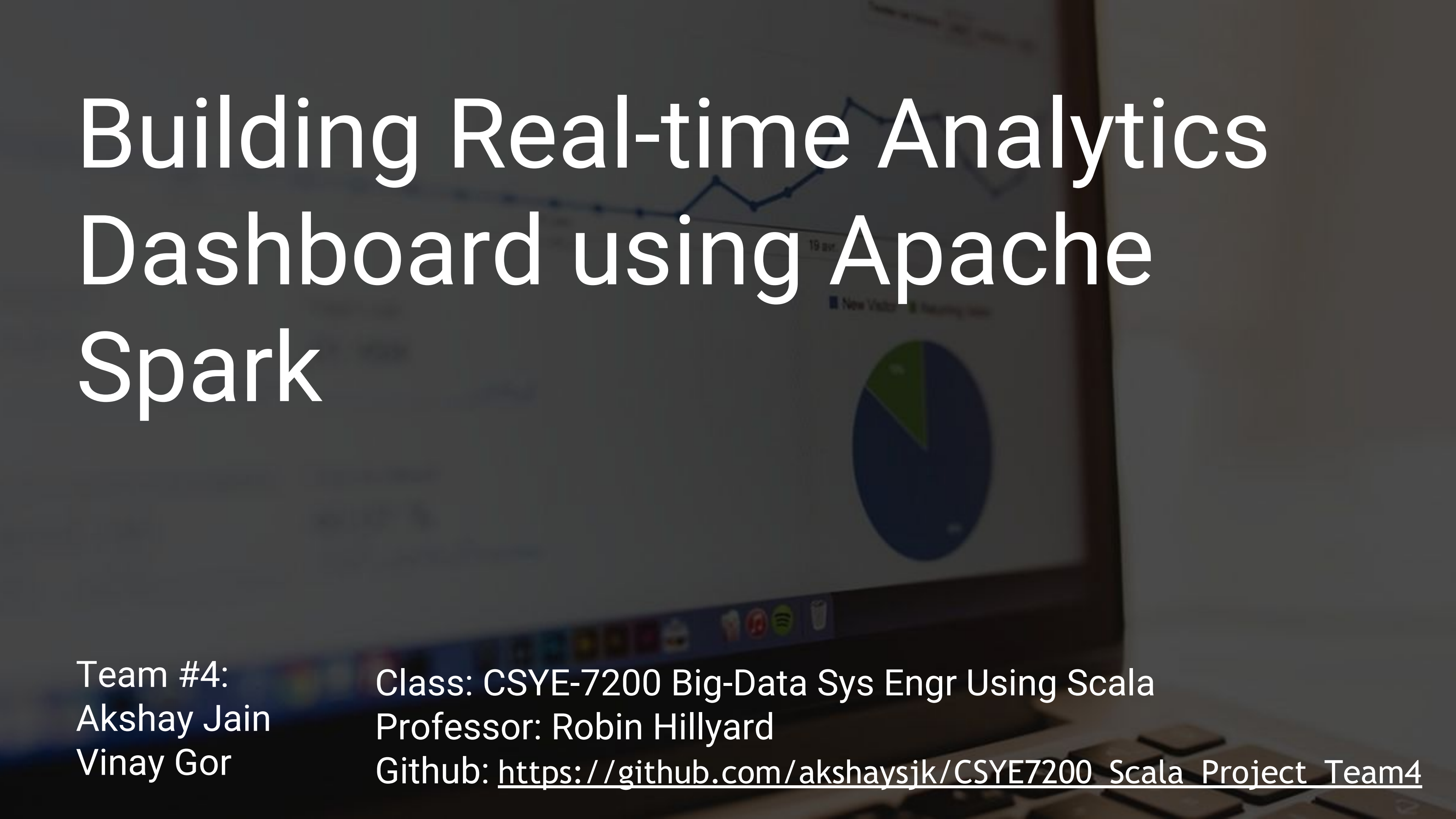
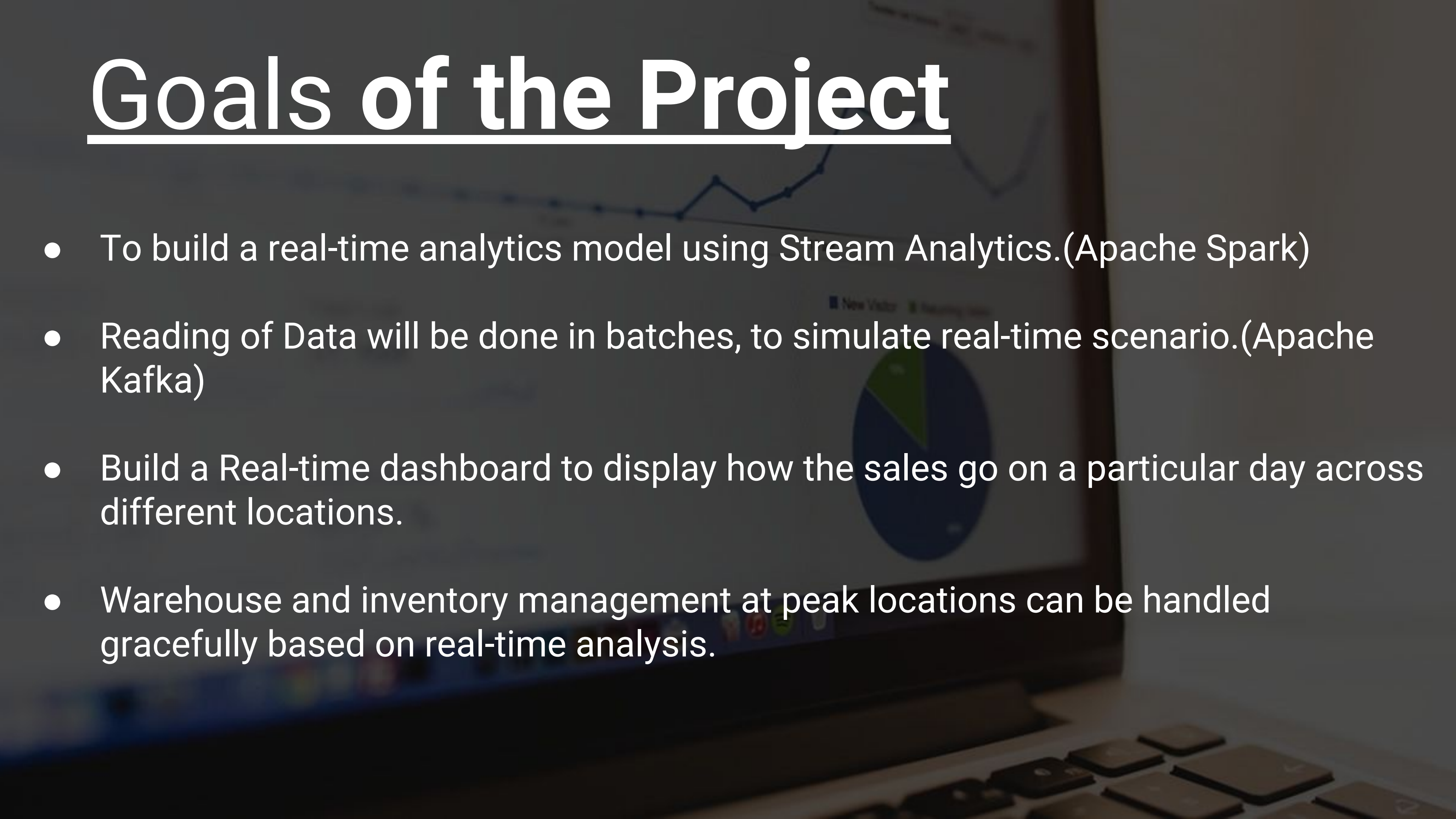


Building Real-time Analytics Dashboard using Apache Spark

Team #4:
Akshay Jain
Vinay Gor

Class: CSYE-7200 Big-Data Sys Engr Using Scala
Professor: Robin Hillyard
Github: https://github.com/akshaysjk/CSYE7200_Scala_Project_Team4



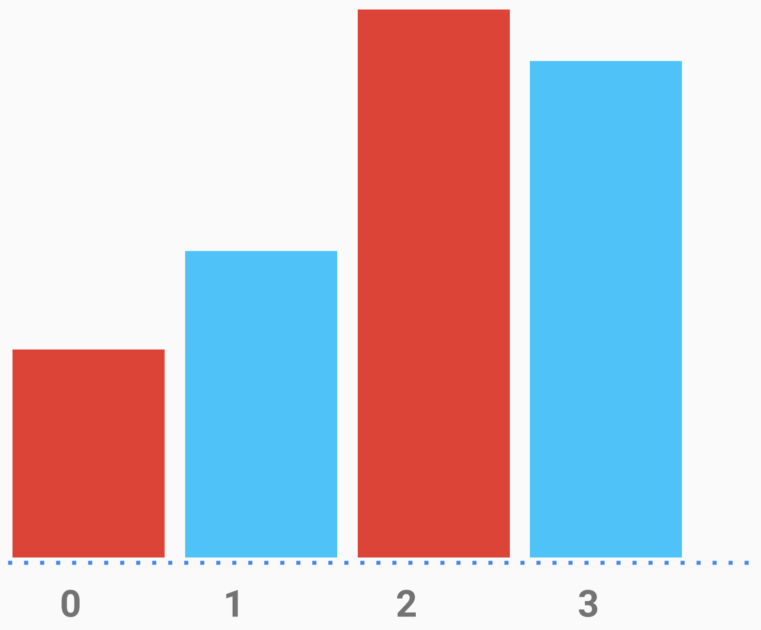
Goals **of the Project**

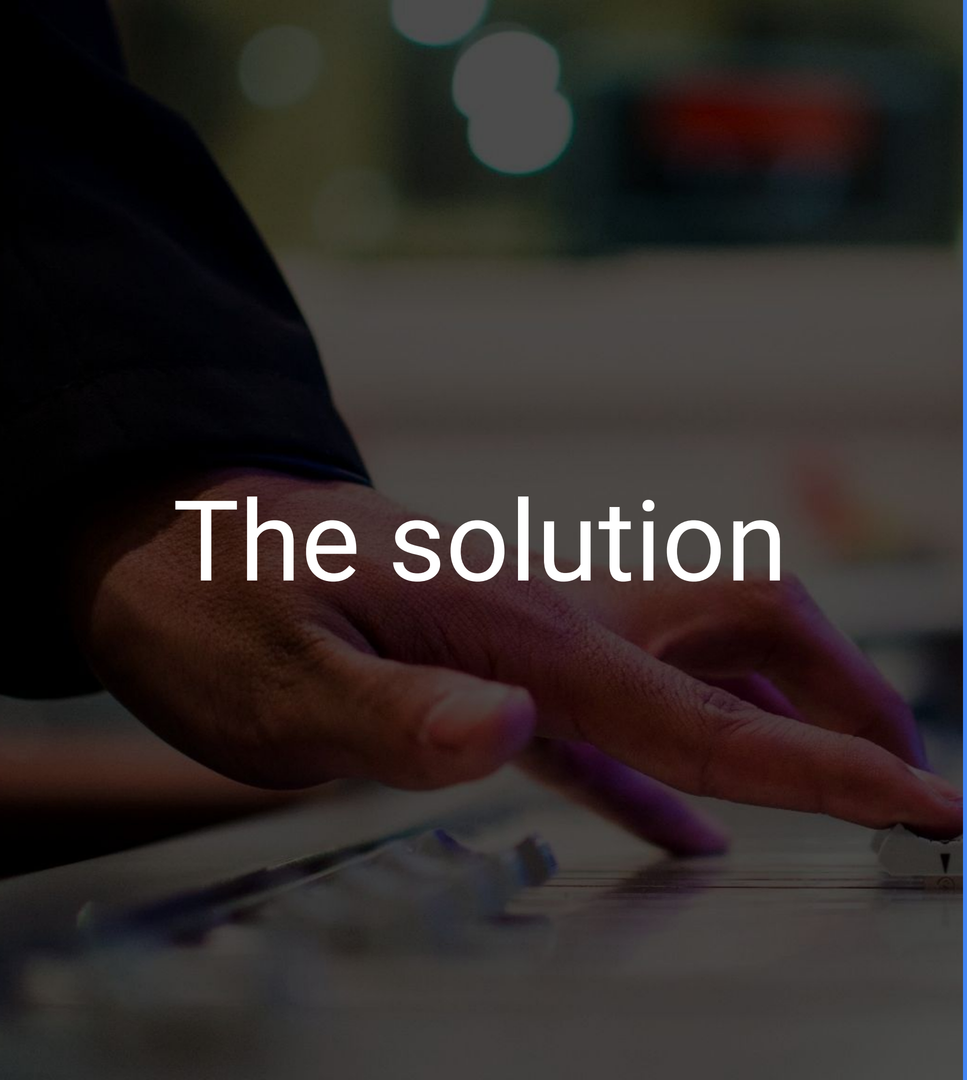
- To build a real-time analytics model using Stream Analytics.(Apache Spark)
- Reading of Data will be done in batches, to simulate real-time scenario.(Apache Kafka)
- Build a Real-time dashboard to display how the sales go on a particular day across different locations.
- Warehouse and inventory management at peak locations can be handled gracefully based on real-time analysis.

The problem

Batch Processing

The time delay between the collection of data and getting the result after the batch process.





The solution

Real-time Processing

Get the analytics in real-time on Dashboard

Data Source

Data is taken from an ongoing competition on Analytics Vidhya website : Practice Problem: Black Friday

URL:
https://datahack.analyticsvidhya.com/contest/black-friday/#data_dictionary

Data :

train.csv : - consists of products and user details 0.5 million rows and 12 columns

Milestones / Sprints

Sprint 1

- Data pipelining through Kafka
- Integrating Kafka and Spark Streaming
- Unit Testing

Sprint 2

- Data cleaning and processing
- Initial setup for UI
- Unit Testing

Sprint 3

- Integration and implementation of UI
- User Dashboard, Machine Learning model (optional)
- Unit and Integration Testing

Sprint 4

- Handle fallouts from previous sprint(s), if any
- System Testing

Blockers faced

Version compatibility with Scala/Play/Spark

Play Framework

Web Sockets

Methodology

Step 1

Read Data from CSV through Apache Kafka

Step 2

Using Spark Streaming to read data

Step 3

Clean the Data, run analysis and pass data to Dashboard through Websockets

Technology Stack

Technologies:

- Apache Kafka
- Spark Streaming
- Play Framework
- Web Socket Communication to pass data from Controller to Dashboard
- Highcharts.js for displaying charts
- Akka Actors for communication with Web Socket

Languages Used:

- Scala (66%)
- JavaScript (26%)
- HTML (6%)

Actor/Use cases

Actors:

Ecommerce System

Ecommerce Company employees

Ecommerce End users

Use cases:

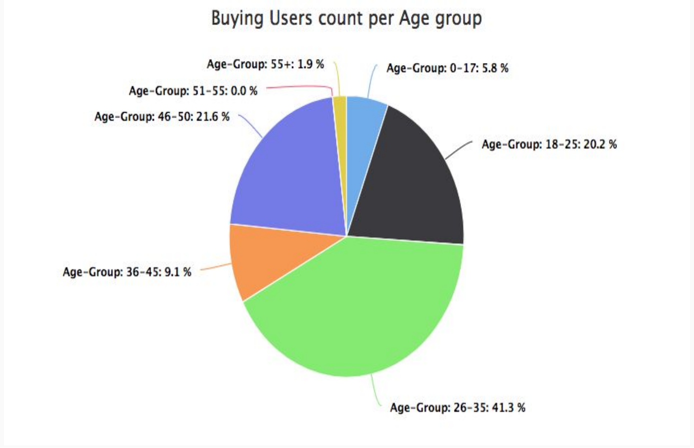
1. Employee can see overview of the sales of products(eg: highest selling Product) on the dashboard homepage

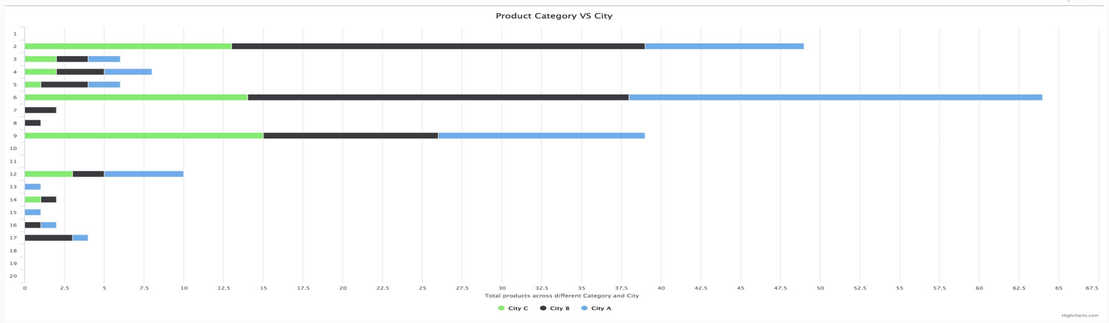
2. Employee inputs query (such as product number) and gets the query specific real-time values.

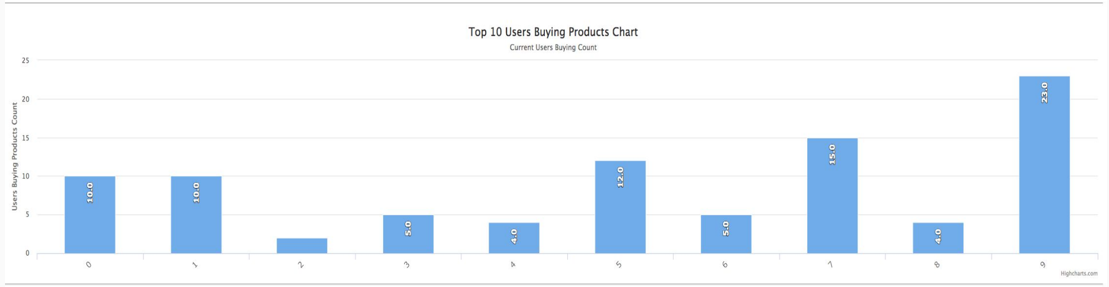
3. On the End-users' dashboard, users will be able to see the recommendations of products based the historical purchases they have made. (optional)

Use Case 1

Employee can see overview of the sales of products(eg: highest selling Product) on the dashboard homepage

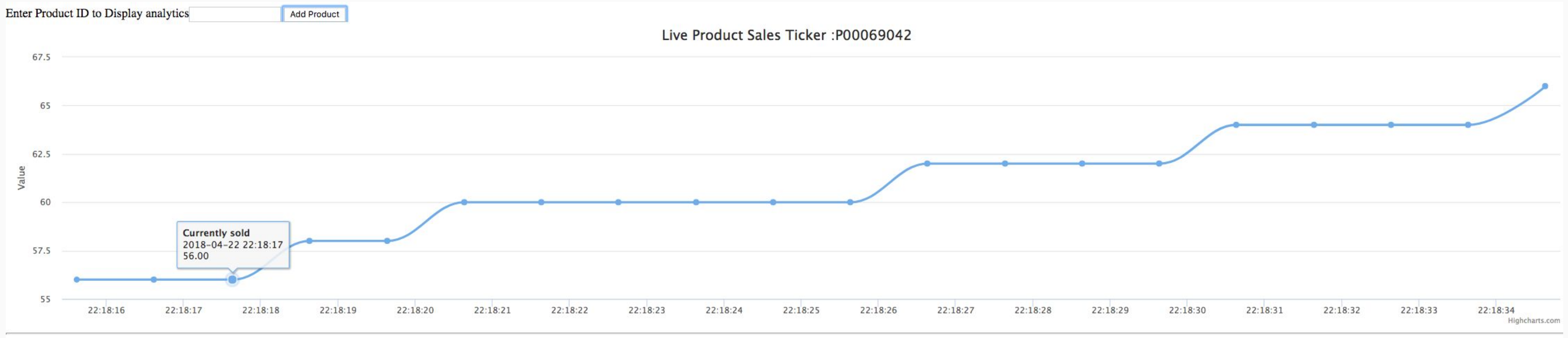




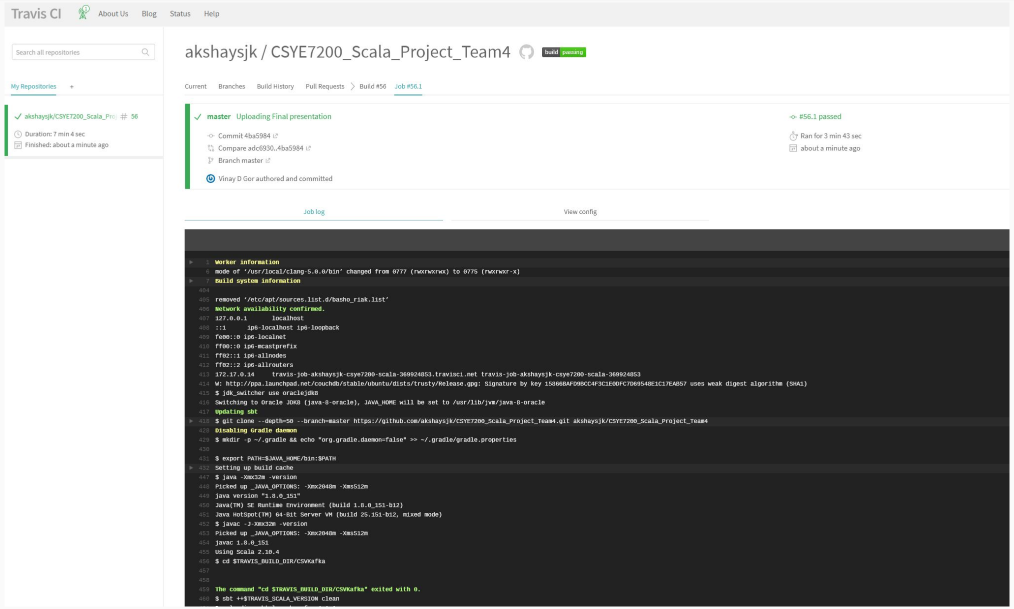


Use case 2

Employee inputs query (such as product number) and gets the query specific real-time values.



Integration with TravisCI and Test Cases



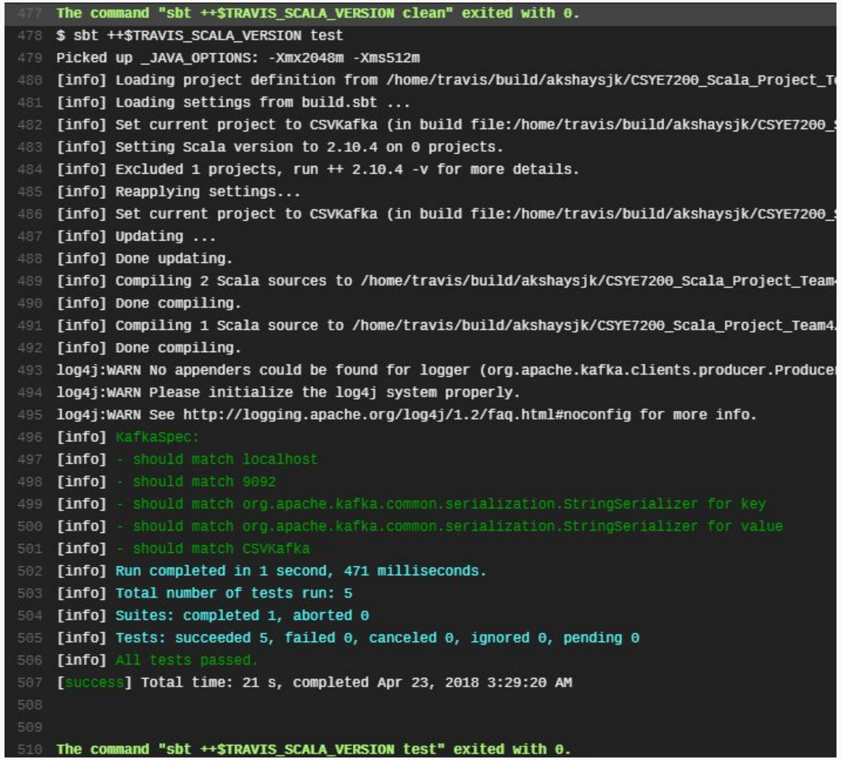
akshaysjk / CSYE7200_Scala_Project_Team4

akshaysjk/CSYE7200_Scala_Pro # 56
Duration: 7 min 4 sec
Finished: about a minute ago

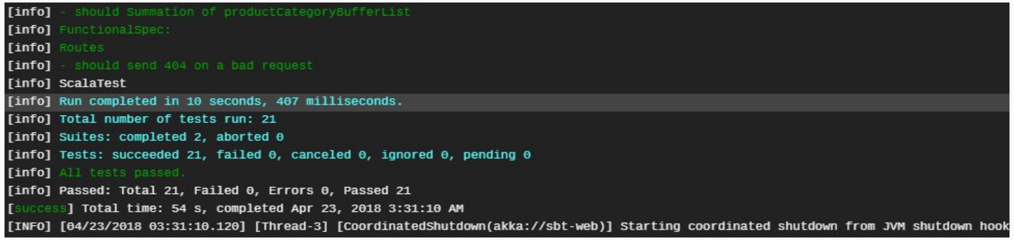
master Uploading Final presentation
Commit 4ba5984
Compare adc6930..4ba5984
Branch master
Vinay D Gor authored and committed

#56.1 passed
Ran for 3 min 43 sec
about a minute ago

```
Worker information
mode of '/usr/local/clang-5.0.0/bin' changed from 0777 (rwxrwxrwx) to 0775 (rwxrwxr-x)
Build system information

removed '/etc/apt/sources.list.d/basho_riak.list'
Network availability confirmed.
127.0.0.1       localhost
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.17.0.14     travis-job-akshaysjk-csye7200-scala-369924853.travisci.net travis-job-akshaysjk-csye7200-scala-369924853
W: http://ppa.launchpad.net/couchdb/stable/ubuntu/dists/trusty/Release.gpg: Signature by key 15866BAFD9BCC4F3C1E0DFC7D69548E1C17EAB57 uses weak digest algorithm (SHA1)
$ jdk_switcher use oraclejdk8
Switching to Oracle JDK8 (java-8-oracle), JAVA_HOME will be set to /usr/lib/jvm/java-8-oracle
Updating sbt
$ git clone --depth=50 --branch=master https://github.com/akshaysjk/CSYE7200_Scala_Project_Team4.git akshaysjk/CSYE7200_Scala_Project_Team4
Disabling Gradle daemon
$ mkdir -p ~/.gradle && echo "org.gradle.daemon=false" >> ~/.gradle/gradle.properties

$ export PATH=$JAVA_HOME/bin:$PATH
Setting up build cache
$ java -Xmx32m -version
Picked up _JAVA_OPTIONS: -Xmx2048m -Xms512m
java version "1.8.0_151"
Java(TM) SE Runtime Environment (build 1.8.0_151-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.151-b12, mixed mode)
$ javac -J-Xmx32m -version
Picked up _JAVA_OPTIONS: -Xmx2048m -Xms512m
javac 1.8.0_151
Using Scala 2.10.4
$ cd $TRAVIS_BUILD_DIR/CSVKafka


The command "cd $TRAVIS_BUILD_DIR/CSVKafka" exited with 0.
$ sbt ++$TRAVIS_SCALA_VERSION clean
```



```
The command "sbt ++$TRAVIS_SCALA_VERSION clean" exited with 0.
$ sbt ++$TRAVIS_SCALA_VERSION test
Picked up _JAVA_OPTIONS: -Xmx2048m -Xms512m
[info] Loading project definition from /home/travis/build/akshaysjk/CSYE7200_Scala_Project_T
[info] Loading settings from build.sbt ...
[info] Set current project to CSVKafka (in build file:/home/travis/build/akshaysjk/CSYE7200_
[info] Setting Scala version to 2.10.4 on 0 projects.
[info] Excluded 1 projects, run ++ 2.10.4 -v for more details.
[info] Reapplying settings...
[info] Set current project to CSVKafka (in build file:/home/travis/build/akshaysjk/CSYE7200_
[info] Updating ...
[info] Done updating.
[info] Compiling 2 Scala sources to /home/travis/build/akshaysjk/CSYE7200_Scala_Project_Team
[info] Done compiling.
[info] Compiling 1 Scala source to /home/travis/build/akshaysjk/CSYE7200_Scala_Project_Team4
[info] Done compiling.
log4j:WARN No appenders could be found for logger (org.apache.kafka.clients.producer.Produce
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
[info] KafkaSpec:
[info] - should match localhost
[info] - should match 9092
[info] - should match org.apache.kafka.common.serialization.StringSerializer for key
[info] - should match org.apache.kafka.common.serialization.StringSerializer for value
[info] - should match CSVKafka
[info] Run completed in 1 second, 471 milliseconds.
[info] Total number of tests run: 5
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 5, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[success] Total time: 21 s, completed Apr 23, 2018 3:29:20 AM


The command "sbt ++$TRAVIS_SCALA_VERSION test" exited with 0.
```



```
[info] - should Summation of productCategoryBufferList
[info] FunctionalSpec:
[info] Routes
[info] - should send 404 on a bad request
[info] ScalaTest
[info] Run completed in 10 seconds, 407 milliseconds.
[info] Total number of tests run: 21
[info] Suites: completed 2, aborted 0
[info] Tests: succeeded 21, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[info] Passed: Total 21, Failed 0, Errors 0, Passed 21
[success] Total time: 54 s, completed Apr 23, 2018 3:31:10 AM
[INFO] [04/23/2018 03:31:10.120] [Thread-3] [CoordinatedShutdown(akka://sbt-web)] Starting coordinated shutdown from JVM shutdown hook
```

System Performance/Scalability

Iteration	Kafka producer push data to Topic Time (in ms)	Spark Streaming Read Time interval from Topic (in seconds)	Stream Data Size range received	Time to Display/ update graphs on UI (in seconds)
1	2000 ms (low input traffic)	2 sec	1-2 records	3 seconds
2	1000 ms	4 sec	4-5 records	5 seconds
3	500 ms	5 sec	9-10 records	6 seconds
4	100 ms	6 sec	55-60 records	7 seconds
5	50 ms (high input traffic)	6 sec	110-115 records	7 seconds
6	20 ms(very high input traffic)	10 sec	450+ records	15-20 seconds

Right Configuration?

Iteration	Kafka producer push data to Topic Time (in ms)	Spark Streaming Read Time interval from Topic (in seconds)	Stream Data Size range received	Time to Display/ update graphs on UI (in seconds)
5	50 ms (high input traffic)	6 sec	110-115 records	7 seconds
6	20 ms(very high input traffic)	10 sec	450+ records	15-20 seconds

Until the 5th iteration configuration, the back-end and front-end works gracefully.

But, for 6th iteration, the data was being handled gracefully in the backend, but the UI part was taking some time to process. It took some time to display the analytics.

Conclusion: Configurations set for iteration 5 seems ideal for our acceptance criteria and our application can handle 110-115 records gracefully every 6 seconds.

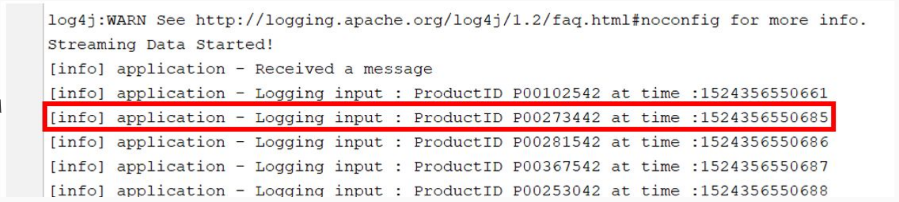
Acceptance criteria

- 85% of the time, Spark Streaming will clean the data received, process it and generate/update the dashboard within 10 sec

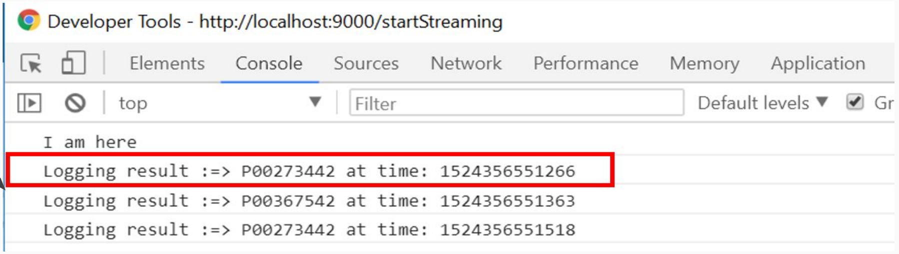
1524356551266 – 1524356550685 = 605 Milliseconds
~ 0.6 seconds

1524356552174-1524356550728 = 1446
 ~ 1.5 seconds

Time range ~ 0.6 to 5.3
Criteria Met!

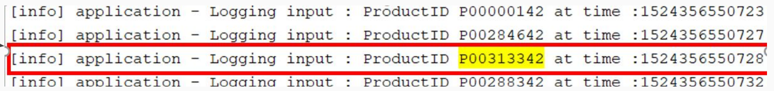


```
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Streaming Data Started!
[info] application - Received a message
[info] application - Logging input : ProductID P00102542 at time :1524356550661
[info] application - Logging input : ProductID P00273442 at time :1524356550685
[info] application - Logging input : ProductID P00281542 at time :1524356550686
[info] application - Logging input : ProductID P00367542 at time :1524356550687
[info] application - Logging input : ProductID P00253042 at time :1524356550688
```

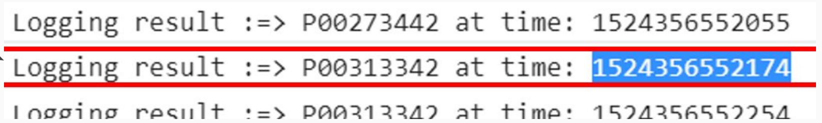


Developer Tools - http://localhost:9000/startStreaming

```
I am here
Logging result :=> P00273442 at time: 1524356551266
Logging result :=> P00367542 at time: 1524356551363
Logging result :=> P00273442 at time: 1524356551518
```



```
[info] application - Logging input : ProductID P00000142 at time :1524356550723
[info] application - Logging input : ProductID P00284642 at time :1524356550727
[info] application - Logging input : ProductID P00313342 at time :1524356550728
[info] application - Logging input : ProductID P00288342 at time :1524356550732
```



```
Logging result :=> P00273442 at time: 1524356552055
Logging result :=> P00313342 at time: 1524356552174
Logging result :=> P00313342 at time: 1524356552254
```

Future Scope

- Recommendation of product to Users with the help of Machine learning Algorithm

- User’s Dashboard to display Analytics of the products purchased

Thank You!