На главной странице был разработан кастомный компонент, основывающийся на компоненте Text. При нажатии по нему вызывается событие, которое меняет цвет текста на случайный цвет при клике по компоненту. Результат работы компонента представлен на рисунке 6.

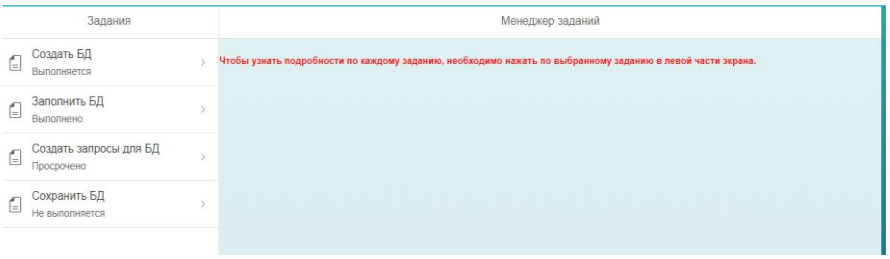


Рисунок 6 – Главная страница после нажатия по компоненту

В ходе работы написано приложение «Менеджер заданий» для управления заданиями с использованием технологий: HTML, CSS, JavaScript, OpenUI5 [1–2]. Приложение хранит следующую информацию: название, статус, приоритет, заказчика, дату получения и окончания задания и статус выполнения задания в процентах. Также приложение выполняет все основные действия с записями: добавление, удаление, изменение. Приложение способно выполнять валидацию вводимых данных, интернационализацию i18n и адаптировано под маленькие экраны. Разработанное приложение протестировано.

Приложение создано при помощи среды разработки Visual Studio Code.

## Литература

1 Официальная документация OpenUI5 – OpenUI5 Documentation : [Электронный ресурс] // URL: https://openui5.hana.ondemand.com/. – Дата доступа: 28.01.2019.

2 SAP UI5 Tutorial : [Электронный ресурс] // URL: https://www. tutorialspoint. com/sap_ui5 – Дата доступа: 25.02.2019.

**УДК 004.67**

***М. Н. Гавриленко***

## РАЗРАБОТКА СИСТЕМ ОБРАБОТКИ И АНАЛИЗА ДАННЫХ С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТОВ BIG DATA

*В статье рассматриваются вопросы актуальности использования новых технологий BigData для анализа и обработки информации. Функциональные возможности использования подобных технологий и языков программирования рассматриваются на примере разработки проекта по обработке различных данных в реальном времени, а также отображения информации в Angular приложении. Рассматриваются перспективы использования брокера-сообщений Kafka, а также Spark фреймворк для обработки и хранения информации.*

Актуальность создания новых проектов и приложений в настоящее время трудно переоценить, ведь последнее время все данные, которые нас окружают, хранятся и записываются куда-либо, будь то облачные хранилища или обычные данные на

домашнем компьютере. Увеличение информации в больших объемах, быстрый рост и появление новых информационных технологий требуют использования новых современных инструментов и методов, отвечающих современному уровню технических средств обработки большого количества данных. Огромные корпорации нуждаются в анализе своих данных и отображении их на Интернет ресурсах. В связи с этим началась разработка и использование новых технологий, которые удовлетворяли бы основным функциям: быстроте обработки данных, качеству и простоте использования.

На сегодняшний день становится популярным изучение и применение технологий BigData для анализа и обработки большого количества информации, которая может быть как структурируемой (данные JSON формата, CSV, и др.) так и не структурируемой (файлы логирования).

Apache Spark (рисунок 1) позволяет читать файлы данных различных форматов и объемов, также позволяет читать Streaming данные (во время загрузки данных). Реализация Spark написана на Scala – языке программирования, который включает в себя как объектно-ориентированную парадигму, так и функциональную. Нативно Spark поддерживает Scala, Python и Java. Основным понятием в Spark'е является RDD (Resilient Distributed Dataset), который представляет собой Dataset, над которым можно делать преобразования двух типов: трансформация и действия.

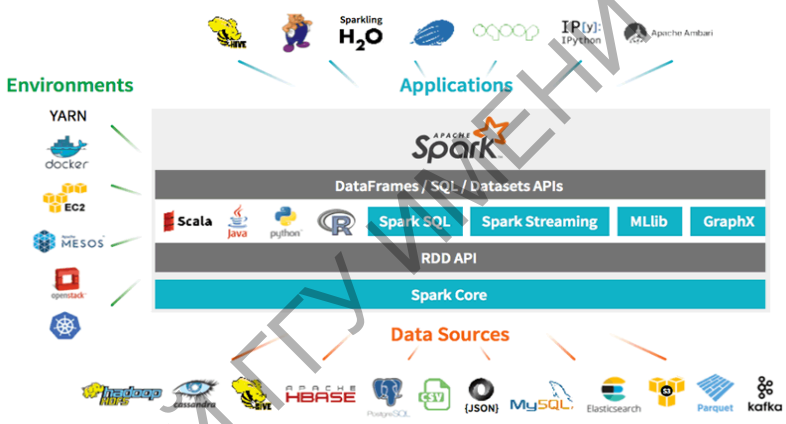


Рисунок 1 – Схема использование Apache Spark

Результатом применения операции трансформации RDD является новый RDD. Как правило, это операции, которые каким-либо образом преобразовывают элементы данного исходного набора данных. Действия применяются тогда, когда необходимо материализовать результат – как правило, сохранить данные на диск, либо вывести часть данных в консоль.

Загружать данные в Spark можно двумя путями: непосредственно из локальной программы с помощью функции .parallelize(data); из поддерживаемых хранилищ (например, hdfs) с помощью функции .textFile(path). Стоит отметить, что Scala и, в частности, Spark поддерживают огромное количество операций, которые можно проводить с набором данных, таких как сортировка, группировка, различное агрегирование данных и многое другое, что позволяет определить свою стратегию обработки, анализа и интерпретации данных [1].

После каких-либо действий можно также и записать полученные данные в какой-либо формат файлов: csv, parquet, avro, текстовый файл и др. Плюсом таких технологий является то, что вы можете прочитать файлы из какого-либо хранилища в любом формате, и после обработки записать файл. Таким образом, работа с данными стала проще, быстрее и более интереснее.

Достаточно популярным на сегодняшний день является еще один инструмент хранения и обработки данных Apache Kafka – распределённый программный брокер сообщений. Написан он на языках программирования Scala и Java. Спроектирован как распределённая, горизонтально масштабируемая система, обеспечивающая наращивание пропускной способности как при росте числа и нагрузки со стороны источников, так и количества систем-подписчиков. Подписчики могут быть объединены в группы. Поддерживается возможность временного хранения данных для последующей пакетной обработки.

На рисунке 2 изображена схема работы инструмента ApacheKafka, который состоит из поставщика сообщений или информации (Producer), которые записываются в определенный пакет (Topic), и потребителя, который читает или получает данную информацию из определенного пакета Topic. Основным плюсом данного инструмента является то, что получатель может подписаться на определенный топик, и при изменении его информации, получает их тут же в режиме реального времених [2].

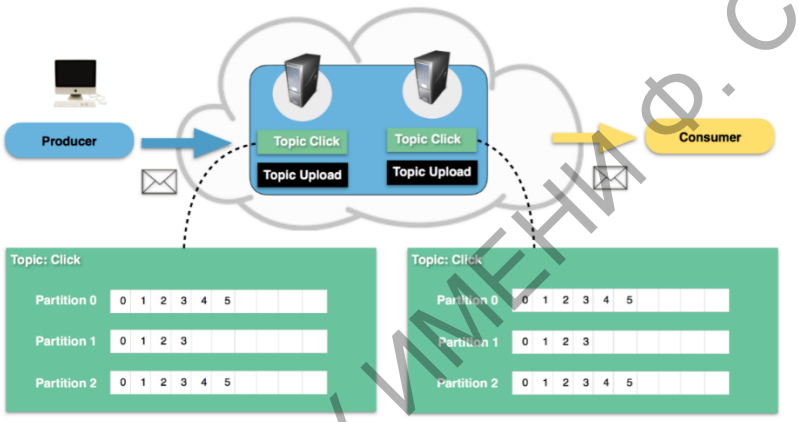


Рисунок 2 – Схема работы ApacheKafka

Примером использования данных инструментов служит разработанная система обработки информации логов (рисунок 3).

Данные записываются в ApacheKafkaв определенный топик, который уже прослушивается пользователем Consumer’ом.При поступлении информации в топик, начинает работать Spark-приложение, которое использует свой функционал Streamingобработки и анализа данных: получает их; производит манипуляции; дальше отдает ее на REST-приложение, написанное на Javaи используемое для получения и отправки данных на UIклиенту. В данной ситуации наиболее оптимальным является создание UI на Angular – популярном JS фреймворке, который имеет достаточно инструментов для работы с данными, который будет напрямую общаться с backend’ом, написанным на Java, что позволит данным обновляться постоянно с определенным промежутком времени.

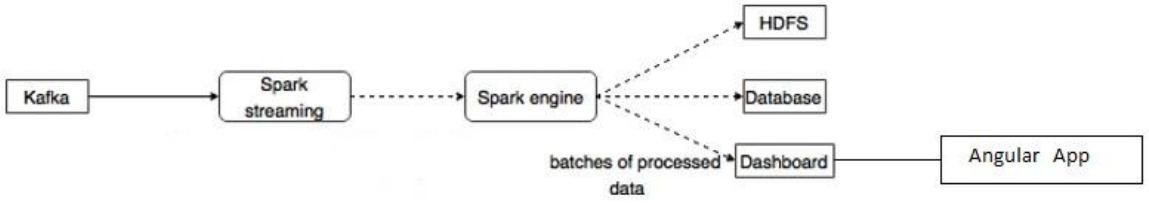


Рисунок 3 – Схема работы приложения

В процессе работы системы анализа данных фиксируется, систематизируется, обрабатывается и анализируется информация, которая отображается в виде графиков (рисунок 4).

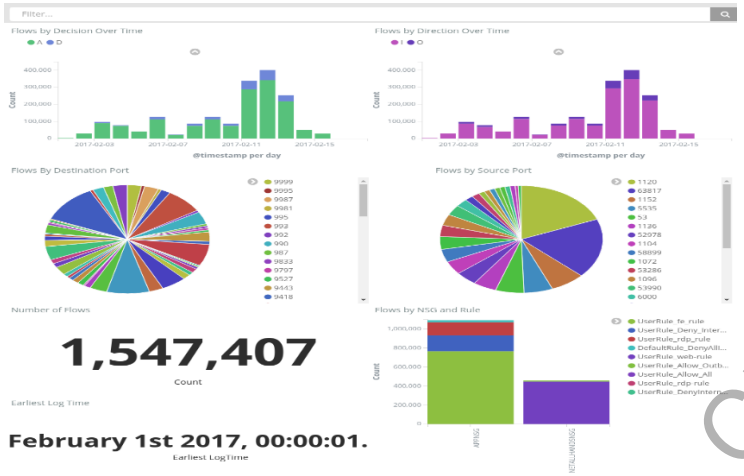


Рисунок 4 – Интерфейс UIприложения с обработанными данными

Разработка системы с использованием языка Java, который превратился из просто универсального языка в целую платформу и экосистему, объединяющую различные технологии, используемые в ряде задач: от создания десктопных приложений до написания крупных веб-порталов и сервисов, обеспечит доступность системы на обычных ПК, планшетах, смартфонах и гарантирует его популярность и востребованность как в среде новичков, так и программистов, изучающих новый вид программного обеспечения.

Представленное приложение отвечает всем современным требованиям разработки программного обеспечения. Данные инструменты используются во многих крупных ITкомпаниях и позволяют решить поставленные задачи.

## Литература

1 Sandy, Ryza Advanced Analytics with Spark: Patterns for Learning from Data at Scale / Ryza Sandy, Uri Laserson, Sean Owen, Josh Wills.Publisher: O'Reilly Media. 2015. – 276 с.

2 Narkhede, NKafka: TheDenitiveGuide / N. Narkhede, G. Shapira, T. Palino. – Publisher: O'Reilly Media. 2017. – 322 с.

**УДК 638.51-77**

***П. В. Гаврилик***

## ПРОГРАММНЫЕ СРЕДСТВА ПЛАНИРОВАНИЯ И ОРГАНИЗАЦИИ СИСТЕМЫ ДОСТАВКИ ГРУЗОВ

*В статье описаны программные средства разработки системы планирования и организации доставки различных категорий грузов множеством транспортных средств. Разработанная система используется в работе интерактивного сервиса грузоперевозок «от отправителя к заказчикам» с учетом множества условий транспортировки доставки.*