Санкт-Петербургский государственный университет
ЦИТИК

ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ
ПО ПРОФИЛЯМ:
“Автоматизация научных исследований”
на тему:
«Аналитика данных электронного расписания»

Выполнили бакалавры 3 курса:

Докиенко Денис Александрович _____
(подпись)

Красноперов Егор Андреевич _____
(подпись)

Левин Сергей Дмитриевич _____
(подпись)

Павлова Екатерина Денисовна _____
(подпись)

Сокол Милена Денисовна _____
(подпись)

Преподаватель-руководитель: Севрюков Сергей Юрьевич, старший преподаватель кафедры ТП, ПМ-ПУ

_____
(подпись)

Представитель организации-партнера: Тремасов Евгений Вячеславович, УСИТ, инженер группы технической поддержки

_____
(подпись)

Научный руководитель ЦИТИК: Иван Станиславович Блеканов, доцент кафедры технологии программирования, заведующий кафедрой технологии программирования ПМ-ПУ

_____
(подпись)

Санкт-Петербург
2019

СОДЕРЖАНИЕ

Введение

В данный момент имеются проблемы с формированием различных аналитических запросов к базе данных электронного расписания СПбГУ. В связи с этим диспетчерам расписания не был обеспечен максимально возможный уровень удобства работы. Основные недостатки существующего в данный момент решения: низкая скорость выполнения запросов, высокая нагрузка на базу данных, малое количество используемых аналитических срезов.

На основе данной проблематики необходимо разработать решение, которое способствовало бы диспетчерам расписания в выполнении их рутинных задач и лишенное указанных выше недостатков существующей системы.

Постановка задачи

Изначальная постановка задачи:

На основе данных электронного расписания и свободного ПО с открытым исходным кодом создать сервис кэширования данных и аналитическое Web приложение. Примеры аналитических срезов: занятость аудиторий, занятость преподавателей, занятость студентов, доступные аудитории по заданным критериям.

Но в процессе работы над проектом понимание задачи изменялось. Были проведены беседы и с диспетчерами, и с заказчиком, чтобы улучшить понимание требуемого продукта.

В ходе общения с диспетчерами были выявлены аналитические срезы и часть проблем, с которыми они сталкиваются в процессе работы. Также команда была ознакомлена с процессом составления расписания. Кроме прочего, было понятно, что данные обновляются в режиме реального времени, а не раз в некоторый продолжительный промежуток времени.

После обсуждения проблематики с УСИТ, удалось выяснить как работает нынешнее решение и каковы его недостатки. Была получена информация о базе, о форме ее хранения и о принципе, по которому осуществляется взаимодействие. Также было сказано о форме представления данных — веб приложение.

Итоговая постановка задачи:

На основе имеющихся данных и ПО с открытым доступом создать веб приложение, которое, имея определенный набор аналитических срезов, способствовало бы более простому составлению расписания, посредством упрощения способа получения необходимой диспетчерам информации. Также необходимо избегать обращения к нынешней базе данных, чтобы снизить нагрузку на нее. Помимо указанных выше требований одним из пожеланий заказчика была адаптивность дизайна веб-приложения под мобильные устройства.

Формирование проектной команды

1. Левин Сергей — teamlead, project manager был выбран по общему согласию. Причина — возможность и желание, а также развитые лидерские качества и высокий уровень общего коммуникативного развития.
2. Докиенко Денис — работа с серверами, а также изредка помощь project manager-у в выполнении его обязанностей. Был выбран по общему согласию, так как имеет опыт системного администрирования и хорошо развитые социальные навыки.
3. Красноперов Егор — работа в выбранном инструментарии (ELK) и в целом одна из ведущих ролей в разработке backend части проекта. Был выбран по общему согласию.
4. Сокол Милена — работа с базой данных. Была выбрана по общему согласию, т.к. имела желание и самый высокий уровень компетенции в работе с базой данных и ее анализе.
5. Павлова Екатерина — работа с выбранным инструментарием (ELK) и в целом одна из ведущих ролей в разработке backend части проекта. Выбрана по общему согласию.

Стоит заметить, что распределение по ролям не было жестким. В той или иной мере каждый помогал коллегам в выполнении их обязанностей, а во многих случаях конечное решение разрабатывалось при непосредственном участии каждого. Особенно сильно это было заметно во время аналитической части проекта, когда выбрать инструментарий и разработать схему конечного решения невозможно без постоянного обсуждения вопросов всей команды.

Используемый инструментарий

Прежде всего, необходимо было снизить нагрузку на основную базу. Также всем участникам проекта было бы проще работать, имея какую-то единую финальную версию проекта. В связи с этим работа велась на виртуальном сервере, предоставленном СПбГУ. Для работы на сервере использовался VPN-клиент **OpenVPN**, так как это решения было рекомендовано системным администратором университета.

Для работы с самой базой был выбран продукт **Microsoft SQL server 2017**, так как основная база также работает на основе данного решения, а следовательно такой выбор поможет избежать проблем с совместимостью. Однако проблема доступа к базе из стороннего приложения не обошли нас стороной.

На сервере в итоговом решении была развернута операционная система **Microsoft Windows server 2016**. Эта система обеспечивает простое взаимодействие с базой данных, так как MS SQL ориентирован в основном на работу с Windows. После выбора ELK встал вопрос о его совместимости, так как все компоненты системы разработаны в первую очередь под операционные системы Unix-семейства. Одно из важных затруднений заключалось в том, что ответственный за работу с серверами имеет куда меньший опыт работы с ОС Windows, чем с Unix. Вероятно, стоило разделить базу данных и ELK, расположив их на виртуальных машинах с разными операционными системами.

В ходе многочисленных исследований, в качестве основного решения был выбран ELK stack. Данный продукт является кроссплатформенным и объединяет все необходимые части реализации проекта. Для формирования полного понимания специфики конечного решения стоит рассмотреть каждый компонент решения в отдельности: **logstash, elastic search и kibana.**

Logstash — компонент, который занимается сбором данных из основной базы, помещая нужную выборку в Elastic Search. Выгрузка данных осуществляется в том числе через стандартные SQL запросы, что безусловно было плюсом. Также запросы обрабатываются довольно быстро и есть возможность автоматически обновлять данные при изменении базы данных, хоть с этой часть и возникли некоторые трудности.

Elastic Search — хранение и систематизация данных, которые передаются из logstash. Является нереляционной базой данных, которая хранит внутри себя информация в формате json. Это значительно упрощает взаимодействие с веб

приложениями, но наличие Kibana и так лишало нас этой необходимости в явном виде.

Kibana — визуализация данных, полученных из elastic search в виде web приложения, что сильно упрощает работу команды, предоставляя качественный и удобный адаптивный frontend ценой малых усилий по настройке и конфигурированию системы. Имеет различные способы визуализации и работы с данными. Однако имеет не слишком простой интерфейс.

Данный стек технологий хорошо адаптирован для работы с большим потоком данных и его части изначально созданы для простого взаимодействия друг с другом.

Для систематизации работы и распределения задач использовался GitHub. Это наиболее популярное решение, имеющие множество достоинств. В контексте данного проекта это знакомство всей команды с функционалом выбранной системы.

Анализ хода проекта

Изначально затруднения возникли на этапе понимания и формализации поставленной задачи. Для решения этого вопроса было проведено множество встреч с заказчиком в различных форматах, также были привлечены диспетчеры для обозначения видения данного решения с их стороны, кроме этого была проведена беседа с командой прошлого года.

В ходе анализа было выявлено, что необходимо создать некое приложение, которое больше подходит именно для аналитики, а под “умным кэшом” подразумевается хранение данных, которое направлено на ускорение выполнения соответствующих запросов диспетчеров.

Вторым этапом были встречи и коммуникации с представителями как вуза, так и заказчика для получения необходимой информации и ресурсов (сервера, копии базы данных). В ходе двух встреч с диспетчерами были определены основные пожелания по аналитическим срезам и общее видение, хотя конкретики в этих встречах было мало, выводы сделаны были, в том числе выявлена частота и скорость обновления данных в базе на примере работы в реальном времени.

После этого последовала работа с полученными данными и ресурсами. С сервером изначально были проблемы, а база данных имела очень сложную и непонятную структуру. Поэтому была задействована помощь со стороны специалистов, а именно: со стороны системного администратора и ведущего разработчика УСИТ.

Следующим этапом — был подбор инструментария, который соответствовал бы поставленным требованиям.

Первый вариант — использования OLAP куба (MOLAP) , вместе с django и bootstrap, однако после уточнения сроков и частоты обновления данных, в ходе бесед с диспетчерами и заказчиком, это решение было отброшено.

Так как, при обновлении данных необходимо перестраивать куб, что занимает какое-то время, а значит сервис становиться недоступным. Количество обновлений данных и необходимость почти мгновенного обновления данных, не позволяет перестраивать куб через заданные промежутки времени (раз в сутки/час) .

Второй вариант — использовать OLAP куб в комбинации с другой аналитической базой, оставив иные части неизменными. Как дополнение к OLAP кубу рассматривалось несколько вариантов, основные из них: использование триггера, если в данный момент куб перестраивается, то запрос перенаправляется в копию базы, либо

же в базу, реализованной посредство ROLAP.

ROLAP — хранит некоторые срезы уже сформированными, а при отсутствии необходимых обращается к обыкновенной реляционной базе. Данное решение помогло бы ускорит процесс выдачи запросов и работало, как минимум, не медленнее, чем обращение непосредственно к базе. Но от этой реализации было решено отказаться ввиду сложности ее реализации и ограниченности имеющегося времени.

Третий вариант — рассматривались различные CMS (*Content management system) ,* но большинство из них не были продуктами Open Source. При более детальном рассмотрении этот вариант оказался не совсем тем, что подходило бы для решения данной задачи.

Четвертый вариант — используется сейчас. Это ELK стек и MS SQL server 2017. Причины, почему был выбран именно данный набор инструментов описаны выше.

Далее наступил сам процесс разработки. Программные решения, которые использовались в ходе реализации, были в новинку для всех членов команды, поэтому сразу полноценно приступить к разработке не являлось возможным. Происходило изучение ELK stack одновременно с разработкой и формированием выбранных запросов из базы посредством SQL. Данные из основной базы были переданы в elastic search посредством logstash. А сам ES связан с kibana. Решение было незавершенным, но дальнейшую работу было принято осуществлять на сервере.

Финальный этап — развертка на сервере имеющегося проекта с дальнейшими доработками. Было установлено все основное ПО на сервере, развернута копия базы, а также налажено взаимодействие между компонентами. Во время развертки возникло довольно большое количество проблем, например: доступ к базе, необходимость установки дополнительного ПО (java 8, JDK и т.д.). Также было введено решение для обновления данных в ES, при обновлении копии базы, хранимой на сервере вместе с ELK. Также имеется доступ к kibana с удаленного устройства, которое находится в пределах сети law.spbu.

Полученные результаты

В ходе работы было получено приложение, которое содержит информацию исключительно для требующихся аналитических срезов, что помогает работать ему значительно быстрее основной базы. Оно имеет адаптивный дизайн под различные устройства, а доступ к нему можно получить с любого устройства, подключенного к соответствующему VPN.

Однако пока не реализовано обновление копии БД, которая является реляционной. Для этого предполагается триггер со стороны сервера с основной базой.

Полученный продукт выполняет предъявленные требования: исключает обращение к основной базе данных с целью получения аналитики, взаимодействия происходят с аналитическим средством, имеется адаптивный дизайн и возможность доступа с удаленных устройств.

Ниже вы можете ознакомиться с полученными результатами в виде скриншотов. На рисунке 1 — пример полученной визуализации, которая отражает занятость факультета ПМ-ПУ по месяцам:

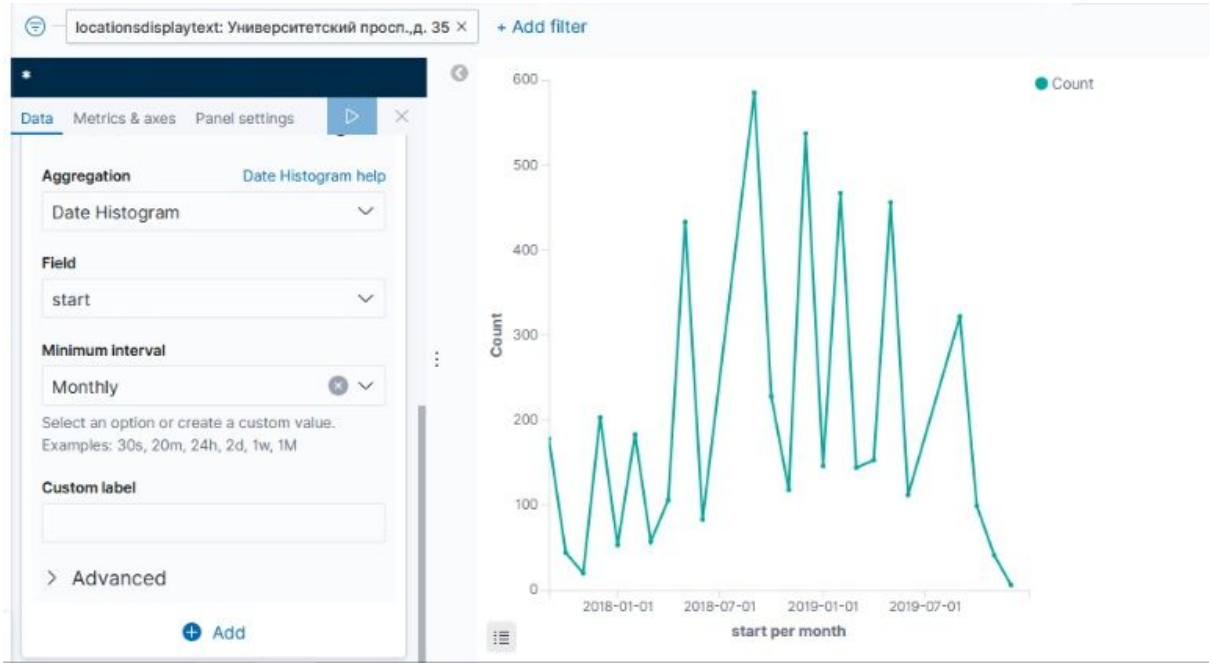

Рисунок 1 — занятость факультета

Пример, который показывает адаптивность дизайна и возможность удаленного доступа можно увидеть на рисунке 2.

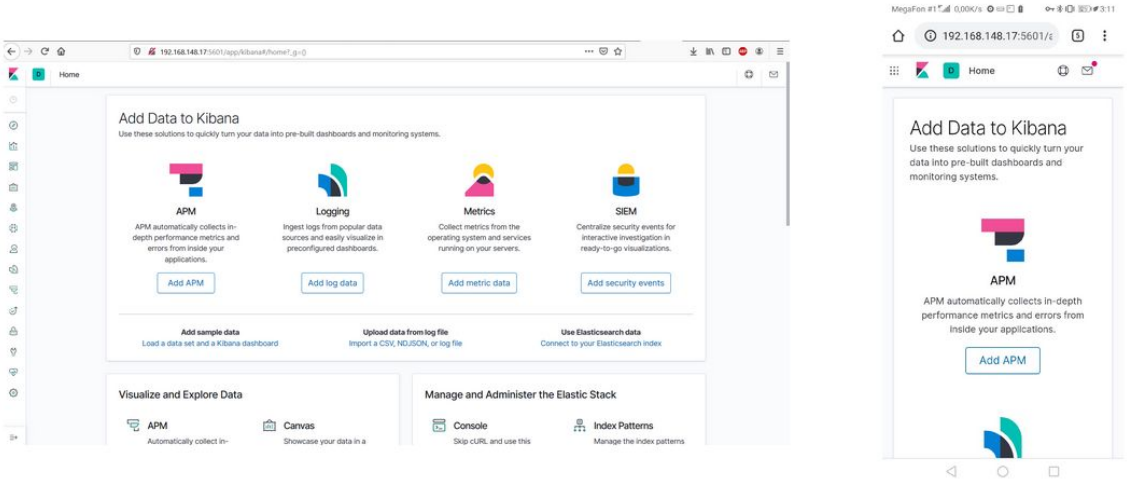

Рисунок 1 — занятость факультета

Более подробно с проделанной работы можно ознакомиться на GitHub страничке проекта:

https://github.com/sergere15/analysis_of_SBPU_timetable

Выводы

Выбранное решение было одним из 4 рассматриваемых вариантов, описанных выше. Как основной инструмент используется ELK stack. Полученный результат удовлетворяет основным требованиям, однако требует доработок. Его внедрение возможно на практике, но, из-за довольно сложного интерфейса kibana, не представляется возможным использование этого решения диспетчерами.

После некоторых улучшений, а именно: введения триггера для взаимодействия с основной базой, упрощения интерфейса путем создания нового WEB интерфейса, написания подробной документации, — это решение вполне может быть введено как основное.