Исходный код оборачивается в 2 докер образа. Один из которых основное приложение, содержащее spring boot приложение, второй содержит скала-спарк код для выполнения ETL преобразований. Далее эти образы помещаются в любое хранилище образов. Приложение запускается в kubernates кластере в отдельном пространстве имен на основе, сохраненного ранее, образа. Каждое создание или изменение сущности на клиенте влечет за союой изменение соответствующей сущности в Kubernetes. Так при создании нового проекта, создается новое пространство имен, при создании джоба – configmap, при создании пайплайна – workflow template. Взаимодействие с образом содержащим спарк код происходит при запуске джоба или пайплайна. При этом создается под с контейнером и как переменные окружения добавляются данные из configmap, созданной на этапе конфигурации джоба пользователем. Такой под работает как управляюзий узел для исполнителей. Их количество может быть изменено пользователем.

При запуске пайплайна создается workflow с ссылкой на ранее созданные workflow template. При необходимости, меняются параметры, влиящие на быстродействие и последовательность выполнения.

Н. А. Коноплич
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **В. В. Васькевич**, ст. преподаватель

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ МЕДИЦИНСКОГО КОНСУЛЬТАНТА С ИСПОЛЬЗОВАНИЕМ СУБД POSTGRESQL И NODE.JS

Веб-приложения – это приложения, с которым можно взаимодействовать с помощью браузера, которое состоит из Frontend и Backend части. В данной статье будет представлен мой опыт разработки такого приложения.

В разработке такого приложения есть три важных этапа: разработка клиентской части (Frontend), разработка серверной части (Backend) и создание базы данных (Database). Для разработки клиентской части я выбрал Java-Script библиотеку React, для разработки серверной части я выбрал модуль Node.js – Express, а для создания базы данных была выбрана СУБД PostgreSQL.

Первоочередной задачей является создание базы данных. Для создания медицинского консультанта понадобилось две таблицы, в одной из которых находятся симптомы (symptoms), с соответствующими

РЕПОЗИТОРИЙ ГГУ ИМЕНИ Ф. СКОРИНЫ

им id, а во второй название болезни(disease), и массив из id, каждому из которых соответствует определенный симптом.

Далее была разработана серверная часть, где нужно уделить внимание нескольким моментам, а именно создание маршрутов, по которым будут передаваться HTTP-запросы, и разработка кода обработки входных данных. Благодаря модулю Express разработка маршрутов не составляет труда, достаточно только указать метод HTTP-запроса, URL, на который будут отправляться данные и скрипт, обрабатывающий эти данные. С кодом обработки данных сложнее, т.к. пришлось разработать метод сортировки данных в несколько шагов:

1. Определение количества совпавших id симптомов введенных клиентов, с массивом id симптомов соответствующей им болезни.
2. Присвоение объекту с соответствующим индексом ключа, содержащего в себе количество совпадений для данного объекта.
3. Создание массива из количеств совпадений, из которого после выбирается наибольшее значение.
4. Создание цикла for, который перебирает объекты с соответствующим индексом, в котором так же содержится цикл, который отбирает из массива объектов объект с наибольшим количеством совпадений, после чего объекты с наибольшим количеством совпадений будут добавлены в новый массив, который будет содержать в себе отсортированные объекты, после завершения работы внутреннего цикла, из массива количеств совпадений будет удалено наибольшее из чисел совпадений, далее будет определено новое количество совпадений, после чего внешний цикл повторится, до тех пор, пока не будут отобраны все объекты.
5. Полученный массив отсортированных объектов будет отправлен в JSON-формате как ответ от сервера.

После того как создание базы данных и серверной части были завершены, я приступил к разработке клиентской части. Страница взаимодействия с клиентом имеет довольно простую структуру, на ней представлены три формы для ввода данных, в первую из которых клиент должен ввести свою температуру, во второй артериальное давление, а в третьей оставшиеся симптомы клиента, после введения которых данные отправятся на сервер, подвергнуться сортировке, после чего будут представлены клиенту. На рисунках 1 и 2 представлен внешний вид страницы для ввода данных и ответ от сервера.

Ниже представлена симуляция POST-запроса на сервер, посредством программы Postman, где в тело запроса передаются данные в json-формате, представляющие из себя ключ name, с массивом из произвольно взятых симптомов и ответ от сервера с массивом из объектов содержащими в себе соответствующие данные о болезни.

РЕПОЗИТОРИЙ ГГУ ИМЕНИ Ф. СКОРИНЫ

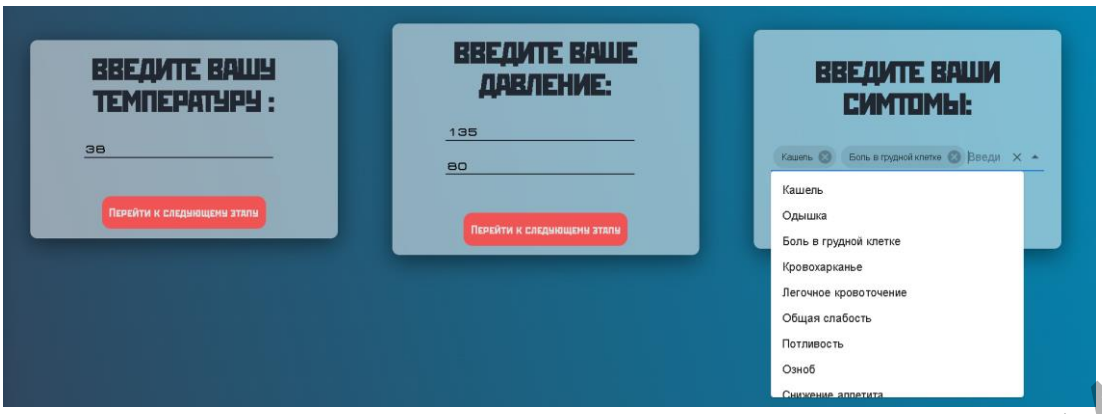

Рисунок 1 – Внешний вид страницы для ввода данных клиента

Рисунок 2 – Внешний вид страницы результата обработки данных

Для простоты восприятия в эти объекты передается только название болезни, соответствующие этой болезни id, массив из совпавших id и количество совпадений (рисунок 3).

Рисунок 3 – Ответ от сервера

РЕПОЗИТОРИЙ ГГУ ИМЕНИ Ф. СКОРИНЫ

Как можно увидеть из рисунка 3, в объекте определенной болезни содержится четыре ключа, в данном случае name, symptoms_id, coincidences и intersection где:

– name это – название болезни;

– symptoms_id – это массив из id соответствующих симптомам для данной болезни;

– coincidences – это количество совпадений симптомов, введенных пользователем с симптомами болезни;

– intersection – это массив из совпавших id;

Из ответа от сервера, можно увидеть, что алгоритм сортировки работает исправно и соответствует поставленным задачам.

Заключение:

Разработка веб-приложения, сложный и трудоемкий процесс, требующий обширных знаний в области программирования и API, умения работы с серверами, знания языка JavaScript.

В процессе разработки веб-приложения пришлось столкнуться с рядом задач, такими как определение архитектуры базы данных, созданием SQL-запросов, подходящими под поставленные задачи, выбором подходящего HTTP-запроса, а также создание алгоритма сортировки данных.

А. Д. Корнодуд
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **Е. И. Сукач**, канд. техн. наук, доцент

АВТОМАТИЗАЦИЯ РАСЧЕТА НАДЕЖНОСТИ СТРУКТУР-ТРЕХПОЛЮСНИКОВ

Структура и особенности функционирования реальных систем графовой структуры столь разнообразны, специфичны и сложны, что моделирование и анализ их характеристик надёжности возможны лишь с применением средств автоматизации расчетов, реализующих стандартизованные расчетные процедуры для вычисления интенсивностей отказов элементов систем, средних времен их восстановления, модули поддержки качественных процедур выявления видов и последствий отказов.

Большинство известных программных комплексов, реализующих расчет надежности систем графовой структуры реализуют универсальные методы, при этом как правило полностью отсутствуют специальные методики, учитывающие специфику объектов. Поэтому актуальна

РЕПОЗИТОРИЙ ГГУ ИМЕНИ Ф. СКОРИНЫ