УДК 004.422.81


С.Ф. Маслович, Р.О. Сеглин
Гомельский государственный университет имени Ф. Скорины

РАЗРАБОТКА МИКРОСЕРВИСА ОТОБРАЖЕНИЯ РАСПИСАНИЯ ЗАНЯТИЙ В УНИВЕРСИТЕТЕ

Во время разработки микросервиса применялись на практике концепция MVC-проектирования приложений, современные языки программирования Java, JavaScript, а также фреймворки Spring и AngularJS. Были проведены исследования в области взаимодействия сервера с клиентом, в ходе которых было получено достаточно сведений для внедрения архитектурного стиля взаимодействия компонентов распределенного приложения в сети REST в разработанный микросервис. При помощи JavaScript была разработана система взаимодействия с базой данных, предоставляемой Google Firebase. Результатом разработки стало полноценное функционирующее веб-приложение со множеством возможностей, в первую очередь с возможностью показа расписания университета, использующее в качестве базы данных виртуальную, что дает возможность показа расписания в режиме реального времени.

Разработка микросервиса, концепция MVC, архитектурный стиль REST, виртуальная база данных, язык программирования Java, язык программирования JavaScript, фреймворки.

Цифровизация охватывает все больше сфер жизни общества, что связано с постоянной потребностью сделать информацию максимально доступной для каждого человека. Растет актуальность цифровых форматов разного рода информации, в частности расписания занятий университета.

На данный момент существует большое количество цифровых инструментов, которые позволяют создавать и отображать расписание для занятий университета, например настольные приложения, андроид-приложения, веб-сайты. Микросервис, рассматриваемый в данной статье, представляет собой веб-сайт. Данный подход был использован из-за ряда преимуществ, таких как отображение расписания занятий в реальном времени, а также огромные возможности этих языков при достаточной простоте их изучения (в отличие от обычных языков программирования, например C или C++).

В данном веб-приложении использована трехуровневая архитектура как частный случай многоуровневой – модульная клиент-серверная архитектура, которая состоит из слоя представления, слоя приложения и слоя данных.

Первым шагом в разработке веб-приложения является создание Maven-проекта в приложении IntelliJ IDEA. После этого с помощью Spring Initializr генерируется пустой проект с внедренной зависимостью Spring BOOT в pom.xml [1]. Этот проект позже распаковывается в ранее созданный проект в IntelliJ. После чего создается минимальный набор классов и помещается в каталоге src/main/java/com/unished/unisheduleapp (подкаталоги com/unished/unisheduleapp ранее генерируются в Spring Initializr). В данном случае в конечном подкаталоге уже существует сгенерированный ранее класс UnisheduleAppApplication, содержащий метод main, который используется Spring Boot для запуска сервлет-контейнера Tomcat и приложения. Далее необходимо создать конфигурацию запуска Spring Boot приложения в IntelliJ. Для этого через Edit configurations создается новая конфигурация типа Application, где в графе Main class выбирается UnisheduleAppApplication, содержащий метод main. Так, при нажатии на Run (или сочетании клавиш Shift+F10) приложение запускается по адресу http://localhost:8080. После этого происходит работа со слоем представления, который и будет отображать интерфейс микросервиса. Для этого создается index.html в каталоге src/main/resources/static, который будет автоматически запускаться при открытии страницы микросервиса. После этого происходит наполнение содержимым этого файла с использованием HTML-разметки и таблицей каскадных стилей CSS. Также используется свободный набор инструментов для создания сайтов и веб-приложений – Twitter Bootstrap. При наполнении index.html создаются четыре контейнера, второй и третий из которых представляют собой прямоугольные окна (второй контейнер служит для хранения «шапки» страницы микросервиса, а в третьем будет выводиться расписание и происходить поиск по группам/преподавателям); первый и четвертый контейнеры не содержат никакой информации и служат лишь для разграничения второго и третьего контейнеров от остальной страницы (рис. 1).

Рис. 1. Структура контейнеров

Для реализации слоя приложения в данном микросервисе используются файлы js, в которых описаны функции для получения и вывода информации. Эти функции определены в контроллере AngularJS; также используются Java-классы со Spring-аннотациями, которые разделяют каждый слой приложения. Для реализации слоя приложения используются класс контроллера (помечен аннотацией @RestController; представляет собой часть слоя представления), который непосредственно взаимодействует со слоем представления, а также класс сервиса (помечен аннотацией @Service; представляет собой бизнес-логику), который вычисляет даты на текущую неделю, а также конфигурацию текущей недели (под чертой/над чертой). Вычисленные данные при помощи контроллера отправляются через GET-запрос в слой представления, где AngularJS с помощью того же запроса получает их, а позже либо использует эти данные для заполнения расписания занятий, либо выводит на страницу веб-приложения в приемлемом для пользователя виде. Контроллер AngularJS захватывает всю главную страницу микросервиса, что позволяет минимизировать количество строк кода, используя, к примеру, только директивы [2]. Так, для отображения списка групп и факультетов используется отдельная директива, код которой расположен в отдельном html-файле. Аналогично используются директивы для отображения поля поиска преподавателя, а также отображения таблиц с расписанием. Когда пользователь выбирает группу или преподавателя, активизируется работа маршрутизатора AngularJS, который позволяет сделать веб-приложение одностраничным (делает возможным переходы на разные страницы без перезагрузки главной страницы). При выборе группы маршрутизатор выводит содержимое страницы с таблицей для группы (содержимое директивы). Аналогично при выборе преподавателя. В то же время при поиске расписания по преподавателям при вводе фамилии каждый раз вызывается функция фильтрации, которая из списка всех преподавателей находит тех, в фамилии которых содержится строка, введенная пользователем. Отфильтрованные данные передаются на веб-страницу списком, в котором пользователь может выбрать необходимого ему преподавателя. Фильтрация начинает работать при вводе трех и более символов. Веб-приложение использует данные, полученные из виртуальной базы данных Google Firebase Realtime Database, которая обновляется в режиме реального времени [3]. Соединение с базой данных происходит в отдельном js-файле, где полученные данные (в формате JSON) сразу заносятся в глобальную переменную, которая позже используется в контроллере AngularJS. Соединение с базой данных требует некоторого времени, поэтому для ожидания получения данных в контроллере предусмотрен таймер – в это время для пользователя будет отображаться индикатор загрузки.

Далее рассмотрим принцип работы микросервиса. При запуске веб-приложения пользователю необходимо перейти по адресу localhost:8080 в браузере, после чего начнется процесс загрузки интерфейса (в этот момент микросервис получает данные из виртуальной базы данных; процесс загрузки обычно занимает до 4–5 секунд). Затем пользователю становится доступно меню выбора действий: просмотр расписания по группам или преподавателям. Далее пользователь может либо выбрать интересующую его группу (перед выбором группы необходимо выбрать и факультет, к которому относится искомая группа), либо ввести фамилию преподавателя. После того как пользователь, к примеру, выбрал группу, микросервис автоматически показывает расписание для этой группы, содержащееся в 6 отдельных таблицах, представляющих дни недели (рис. 2). Каждая таблица состоит из 5 столбцов: порядковый номер занятия, наименование дисциплины, время проведения занятия, квалификация и ФИО преподавателя, аудитория. Над каждой таблицей указана дата и день недели, а перед первой таблицей указаны группа, для которой выведено расписание, и конфигурация текущей недели (под чертой/над чертой). Микросервис распознает, является ли текущая неделя над чертой/под чертой и, соответственно, выводит расписание с необходимой конфигурацией. Если же пользователю необхо-

димо узнать расписание для конкретного преподавателя, он может ввести его фамилию (если есть преподаватели-однофамильцы, то и инициалы) в соответствующее поле ввода. После ввода первых трех символов пользователю предоставляется полный список похожих фамилий (по крайней мере тех, в которых содержится подпоследовательность введенных пользователем символов), из которых он может выбрать фамилию интересующего его преподавателя, после чего микросервис загружает актуальное расписание на текущую неделю для этого преподавателя.

Рис. 2. Фрагмент страницы с расписанием занятий для группы ПМ-31

Для преподавателя, так же как и в расписании для групп, отобразятся шесть таблиц, представляющих дни недели, но вместо столбца «Преподаватель» в таблицах расписания указана группа, у которой будет занятие у этого преподавателя, а также перед первой таблицей указан преподаватель, расписание которого было выведено. Пользователь может в любой момент, просматривая расписание для группы, просмотреть расписание для преподавателя, и наоборот.

Разработанный микросервис отображения расписания занятий университета – отличный способ узнать актуальное расписание необходимой группы или необходимого преподавателя. В данном проекте есть возможность посмотреть на использование в связке языков программирования и фреймворков, таких как: JavaScript, HTML, CSS, Bootstrap, Java, Spring Framework, AngularJS. Дальнейшая разработка предполагает внедрение дополнительного функционала, например показ расписания по аудиториям с возможностью отображения свободных аудиторий, система учетных записей.

Литература

1. Spring QuickStart Guide. – URL: https://spring.io/quickstart. – Текст : электронный (дата обращения: 11.11.2020).

2. AngularJS Tutorial. – URL: https://www.w3schools.com/angular/default.asp. – Текст : электронный (дата обращения: 2.03.2021).

3. Firebase Documentation. – URL: https://firebase.google.com/docs/web/setup?hl=uk. – Текст : электронный (дата обращения: 11.03.2021).

S.F. Maslovich, R.O. Seglin
Gomel State University named after Francisk Skorina

DEVELOPMENT OF MICROSERVICE FOR DISPLAYING UNIVERSITY CLASS SCHEDULE

During the development of the microservice, the concept of MVC application design, modern programming languages Java, JavaScript, as well as the Spring and AngularJS frameworks were applied in practice. Research was conducted in the field of server-client interaction, during which enough information was obtained to implement an architectural style of interaction of components of a distributed application in the REST network in the developed microservice. Since an important component of such microservice is a database, a system of interaction with the database provided by Google Firebase was developed using JavaScript. The result of the development was a fully functioning web application with many features, primarily with the ability to display the university schedule, using a virtual database as a database, which makes it possible to display the schedule in real time.

Microservice development, MVC concept, REST architectural style, virtual database, Java programming language, JavaScript programming language, frameworks.