УДК 333.71

А. Г. Майоров

РАЗРАБОТКА ПРОГРАММНОГО ИНТЕРФЕЙСА ПО УПРАВЛЕНИЮ БАЗОЙ ДАННЫХ РАСПИСАНИЯ УЧЕБНОГО ЗАВЕДЕНИЯ С ПРИМЕНЕНИЕМ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

Описывается процесс разработки программного интерфейса для управления, импорта и экспорта информации из базы данных учебного заведения. Разработана структура объектно-реляционной базы данных. Спроектирован и реализован программный интерфейс для управления базой данных на основе микросервисной архитектуры с использованием языка Java и его фреймворка Spring Boot. Данное приложение позволяет хранить и генерировать данные для других микросервисов которые предоставляют возможность отображать их на любом устройстве.

Когда цифровые технологии проникают все глубже в нашу жизнь, очень важно не отставать от этой тенденции и тоже пытаться дополнять или вовсе заменять все больше аспектов своей работы данными на цифровых носителях или хранящимися в сети интернет. Вот несколько очевидных причин для этого: данные будут доступны из любого места и в любое время, все что требуется – доступ в интернет и цифровое устройство, которые сейчас есть у каждого. Любое изменение будет сразу же отображено на всех устройствах, просматривающих информацию в данный момент. Удобство манипулирования данными, когда информация хранится на бумажных носителях, единственный способ взаимодействия с ней – это чтение. При работе с данными в электронном формате получаем безграничные возможности: поиск по ключевым словам, переиспользование данных, изменение или их корректировка без трудоемких операций, анализ и другое. Поэтому создание программного интерфейса для базы данных учебного заведения является важным шагом для упрощения учебного процесса и жизни университета в целом. Разберемся что же предлагает данное приложение.

Программный интерфейс является ключевым звеном в цепочке обработки и манипулирования цифровыми данными. Он является посредником между базой данных и приложениями, которые оперируют информацией из нее. Информация, которая хранится в базе данных является структурированной, но она абсолютно бесполезна в своем первозданном виде.

Разработанное приложение предоставляет удобный интерфейс посредством HTTP запросов позволяющий производить с базой любые операции, такие как чтение, удаление, создание, обновление и всевозможные виды поиска по имеющимся данным. Стоит отметить, что данные полученные из данного приложения являются не предназначенными для непосредственной выдачи пользователю, интерфейс возвращает данные в виде JSON объекта [1], который должен быть обработаны следующим звеном в цепочке микросервисов, который и отобразит их пользователю на его устройстве.

Далее более детально о микросервисной архитектуре, используемой при разработке данного программного продукта и цепочке микросервисов, которые могут быть с ним связаны.

Микросервисная архитектура – это подход, при котором единое приложение строится как набор небольших сервисов, каждый из которых работает в собственном процессе и коммуницирует с остальными используя легковесные механизмы, как правило HTTP. Эти сервисы построены вокруг отдельных бизнес–потребностей и развертываются независимо с использованием полностью автоматизированной среды.

Существует абсолютный минимум централизованного управления этими сервисами. Сами по себе эти сервисы могут быть написаны на разных языках и использовать разные технологии хранения данных. Данный подход позволяет разрабатывать легко масштабируемые приложения.

На текущий момент, данный программный интерфейс будет использован тремя другими микросервисами [2]:

– android клиент;
– ios клиент;
– web-приложение.

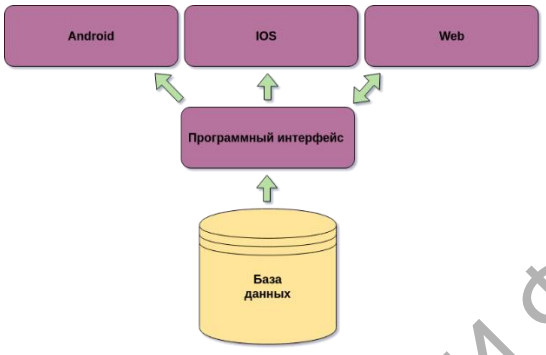

Рисунок 1 – Схема взаимодействия микросервисов

При разработке программного интерфеса требовалось учесть особенности сервиса для работы с расписанием, который оперировал такими понятиями как event (событие – конференция, выступление приглашенного гостя) и lesson (урок – обычное занятие, лекционное или практическое). Сервис расписания должен оперировать этими сущностями как взаимозаменяемыми, но при обычной стратегии применяемой к большинству нормализованных баз данных, это было бы трудоемким и сложнореализуемым, для этого мной было принято решение использовать стратегию single table при создании схемы для этой таблицы.

Принятое решение позволяет хранить в одной таблице сразу несколько типов сущностей, что в свою очередь облегчает процесс взаимодействия с ними.

Данная стратегия достигается путем хранения идентификатора, в качестве отдельного поля в таблице, где значением в поле является тип сущности (рисунок 2).

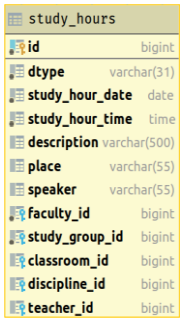

Рисунок 2 – Таблица хранения сущностей расписания

Для реализации программного кода приложения был использован язык программирования Java. На данный момент он является наилучшим выбором для реализации крупных Enterprise проектов по следующим причинам:

1 огромное количество библиотек и фреймворков позволяющих ускорить разработку;
2 строгая типизация языка;
3 обратная совместимость версий;
4 постоянные обновления безопасности;
5 множество готовых решений в сфере разработки крупных проектов.

Для быстрой реализации и развертывания проекта был использован популярный фреймворк Spring Boot. Он предоставляет разработчику возможность не тратить свое время на создание конфигурационных файлов и настройку проекта, потому что внутри себя он содержит так называемый starter – набор настроек и инструментов, установленных по умолчанию и сразу готовых к запуску, но при необходимости предоставляющих простые механизмы по их перенастройке.

Литература

1 Spring Boot – быстрый старт // Сайт с документацией Spring Framework. [Электронный ресурс] – Режим доступа: http://spring-projects.ru/projects/spring-boot/ – Дата доступа: 13.04.19

2 Уоллс, К. Spring в действии / К. Уоллс. – М.: ДМК: Пресс, 2015. – 752 с.

УДК 004.416.6

А. И. Маховик

РАЗРАБОТКА ВНЕШНЕГО ОТЧЕТА В 1С: БУХГАЛТЕРИЯ ДЛЯ БЕЛАРУСИ 2.1 ДЛЯ ОАО «ГОМЕЛЬОБЛАВТОТРАНС»

В статье рассматривается алгоритм разработки внешнего отчета в 1С: Бухгалтерия для Беларуси 2.1, включающий в себя функционал платформы 1С: Предприятие 8.3, такой как управляемые формы, построение запросов и использование различных процедур, функций и свойств элементов с учетом специфики автотранспортного предприятия. Результатом данной работы является внешний отчет, позволяющий пользователю получить сведения по перевозкам, осуществленным по договорным ставкам за заданный период времени.

В современном мире экономический анализ и ведение бухгалтерского учета имеет достаточно сложную структуру. В связи с этим появляется необходимость разрабатывать специальные программы для ведения и учета бухгалтерии. Наиболее распространенной системой автоматизации учета являются программные продукты фирмы «1С». Часто в связи со спецификой производства предприятию необходима доработка типовых решений.

Так, у предприятия-заказчика ОАО «Гомельоблавтотранс» возникла потребность в доработке типовой конфигурации 1С: Бухгалтерия для Беларуси 2.1 в виде разработки отчета, позволяющего пользователю получить сведения по перевозкам, осуществленным по договорным ставкам за заданный период времени. Данный отчет должен содержать гаражный номер машины, маршрут следования, тоннаж перевезенного груза, общее расстояние по путевому листу, расстояние с грузом и без груза, сведения о заказчике, дату выгрузки, сумму дохода, ставку по клиенту (в валюте), общую ставку (в белорусских рублях по курсу на дату выгрузки путевого листа) и доход за 1 км. Отчет должен формироваться в разрезе путевых листов с возможностью выбора вида перевозки.