УДК 004.77

Кононович Т. Д., Козлова Е. И.

ПОТОКОВАЯ ПЕРЕДАЧА МЕДИАКОНТЕНТА ДЛЯ УДАЛЕННЫХ УСТРОЙСТВ ОТОБРАЖЕНИЯ ИНФОРМАЦИИ

Белорусский государственный университет, Минск, Беларусь

В работе представлено приложение для системы потоковой передачи медиаконтента для удаленных устройств отображения информации. При проектировании приложения выбран язык программирования Java, а также фреймворки построения Web-приложений Spring Web и Spring Data для работы с базами данных.

В настоящее время существует большое количество устройств для отображения информации. Для упрощения работы со всеми устройствами создаются различные системы управления, совершенствуются системы контроля. Целью данной работы является создание системы потоковой передачи медиаконтента для удалённых устройств отображения информации. Представлено решение ряда задач для достижения поставленной цели – создать устойчивый канал данных для передачи медиаконтента на устройства отображения информации, разработать систему удалённого управления доступом устройств отображения информации к каналам данных, создать структурированное хранилище данных об устройствах отображения информации.

В качестве системы потоковой передачи медиаконтента используется сервер и устройства отображения информации. Центром системы является сервер, на котором присутствует приложение, управляющее ею, написанное на языке программирования Java. Для правильной организации системы необходимо выбрать устройства вывода, которые способны принимать запросы от приложения на сервере. Самым распространенным способом взаимодействия и передачи данных в сети Интернет является REST API, который позволяет использовать протокол HTTP. Протокол HTTP поддерживается любым браузером, то есть подойдет любое устройство, которые может запускать его. В качестве такого устройства, выбран телевизор со встроенной операционной системой Android, которая поддерживает работу браузера.

Реализовать подобную систему решено на языке объектно-ориентированного программирования Java, который имеет встроенный функционал для потоковой передачи информации любого типа в виде потока байт (рисунок 1).

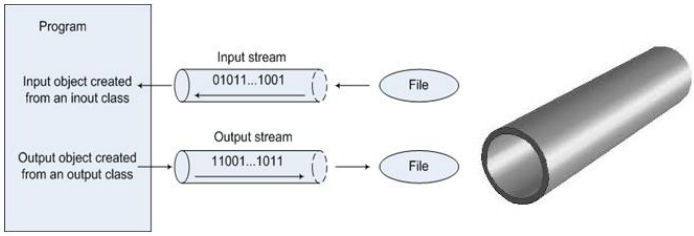

Рисунок 1 – Чтение и запись информации из файла с помощью потоковой передачи данных

Для того чтобы информация об устройствах и медиаконтенте сохранялась, создается хранилище данных, в котором будет хранится вся необходимая информация для корректной работы системы, а именно об устройствах, который включены в эту систему, данные о медиаконтенте и на какое устройство оно должно транслироваться. Стоит отметить, что реляционные базы данных удобны для работы с текстовой информацией, но хранить медиаконтент будет FTP-сервер, ссылка на расположение медиаконтента в нем будет храниться в базе данных.

Для удобства взаимодействия базы данных и приложения на Java существует технология ORM, которая позволяет строить интерпретации таблиц в базе данных с классами сущностей, написанных на Java. Самой удобной архитектурой такой приложения является MVC, схема которой приведена на рисунке 2.

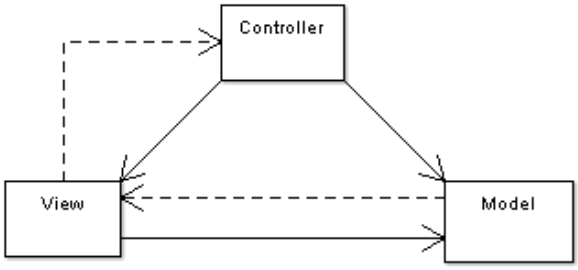

Рисунок 2 – Схема MVC архитектуры

Опишем каждую компоненту данной архитектуры в соответствии с задачами данной работы. Компонент «Модель» соответствует всей логике, связанной с данными, с которой работает пользователь. Это могут быть либо данные, которые передаются между компонентами View и Controller, либо любые другие данные, связанные с бизнес-логикой. Компонент View используется для всей логики пользовательского интерфейса приложения. В данном случае, пользовательский интерфейс – это медиаконтент, переданный на браузер устройства отображения информации. Контроллеры действуют как интерфейс между компонентами модели и представления для обработки всей бизнес-логики и входящих запросов, манипулирования данными с использованием компонента модели и взаимодействия с представлениями для вывода окончательного результата.

Данную модель архитектуры на языке программирования Java помогает создать фреймворк Spring Web. В качестве реализации ORM технологии удобен фреймоворк Spring Data, который будет настраивать взаимодействие между сущностями, написанными на Java и таблицами в базе данных PostgreSQL. Для простой настройки работы приложения на сервере будет использоваться фреймворк Spring Boot, который автоматизирует запуск приложения на сервере.

Описание структуры базы данных, которая будет храниться в себе все вышеперечисленное, представлено на рисунке 3.

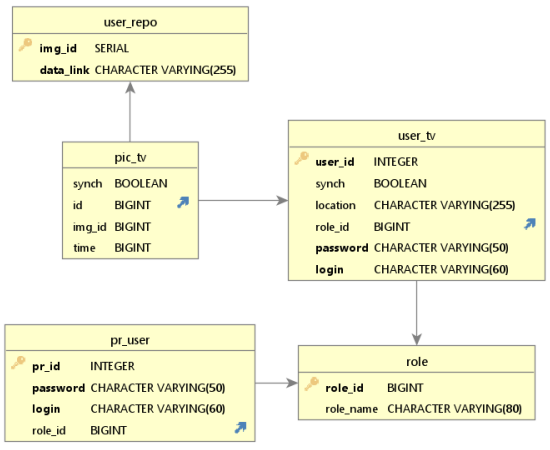

Рисунок 3 – Архитектура базы данных приложения

Краткое опишем содержимое каждой таблицы. Таблица role хранит в себе все возможные роли пользователя или устройства в системе. Таблица pr_user хранит в себе информацию о привилегированных пользователях с различным уровнем доступа, которые могут ка-

ким-либо образом влиять на работу устройства или информацию с помощью системы. Таблица user_tv хранит в себе информацию о пользователях (устройствах), которые отображают информацию. Таблица user_repo хранит в себе информацию о медиаконтенте (изображениях), который может отображаться на устройствах. Таблица pic_tv хранит в себе расписание отображения конкретного медиаконтента на конкретном устройстве. Также эта таблица обеспечивает связь типа «многие ко многим» между таблицей user и user_repo.

Создав базу данных устройств, необходимо реализовать контроллеры в коде программы, которые будут проводить все необходимые манипуляции над устройствами, а также данными, которые на них будут отправляться. Для удобства написания кода программы, а также для соблюдения бизнес-логики написания кода была разработана модель взаимодействия, представленная на рисунке 4.

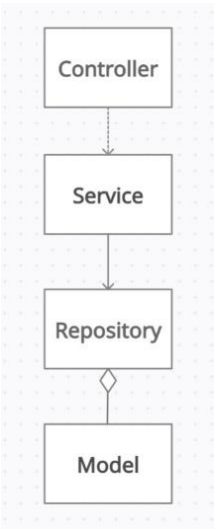

Рисунок 4 – Архитектура взаимодействия компонентов контроллер-модель

В состав модели входят Контроллер – обработчик запросов, модель – представление данных, их функции описаны выше, Репозиторий – хранилище однотипных сущностей (моделей), которые используются для удобного доступа к данным и их обработки. Сервис – инструмент для работы над репозиторием и данными в нем, его включение в модель необходимо, так как репозиторий с точки зрения бизнес-логики не может производить действия над самим собой.

Предложенная система управления потоковой передачи данных на устройства вывода информации может быть интегрирована исходя из поставленных для этой системы задач в проект "Цифровой факультет" факультета радиофизики и компьютерных технологий БГУ [4].

Список литературы

1. Блинов, И. Н., Романчик, В. С. Java from EPAM : учеб.-метод. пособие / И. Н. Блинов, В. С. Романчик. — Минск: Четыре четверти, 2020. — 560 с.

2. Paraschiv, E. Build your API with Spring / E. Paraschiv [Electronic resource]. – 2020. - Mode of access: https://www.baeldung.com/rest-api-spring-guide. - Date of access: 8.04.2023.

3. Рогов, Е. PostgreSQL изнутри / Е. Рогов. – 3-е изд. – Москва : ДМК Пресс, 2022. – 660с.

4. Ю. И. Воротницкий и др. Цифровая интеллектуальная среда факультета / Ю. И. Воротницкий, К. В. Козадаев, Е. И. Козлова, И. А. Шалатонин, Е. А. Головатая, А. М. Соболь // материалы II Междунар. науч.-практ. конф., Минск, 23–24 апреля 2020 г. / редкол.: В. В. Скакун (отв. ред.) [и др.]. – Минск : БГУ, 2020. – С.19 – 22.