

происходит вызов функции `translate`, где введённый для перевода текст сохраняется в строковой переменной. В массив строк `words` заносятся слова языка, с которого ведётся перевод. Сам массив не сохраняет строки по отдельности, а лишь сохраняет адрес, с которого начинается строковый массив при помощи вызова функции `readWords`. В данную функцию передаётся строковая переменная, хранящая код языка. В функции происходит поиск данного кода и возврат адреса строкового массива, хранящего слова в соответствии с кодом языка. Затем введённый для перевода текст сравнивается с каждым словом из массива строк. При совпадении слов происходит сохранение позиции слова, а в массив строк `words` заносятся слова перевода. После происходит проверка на количество слов: если массив строк со словами для перевода содержит количество элементов менее чем сохраненная позиция, происходит вывод на экран устройства сообщения об отсутствии перевода. Такая же ситуация происходит, если введённое слово не найдено в исходном массиве строк.

Отлавливание нажатия на клавиши происходит при помощи метода `setOnEditorActionListener`. В конструкторе данного метода создаётся новый класс `OnEditorActionListener`, определяющий интерфейс, который вызывается, когда в редакторе выполняется действие. Результатом окончания служит идентификатор действия `actionId`, который принимает значения битов `IME_ACTION_DONE`, ключевое действие которых выполняет операцию «Готово» и, как правило, означает, что нет ничего более для ввода, и `IME` будет закрыта.

**М.Н. Коваленко** (УО «ГГУ им. Ф. Скорины», Гомель)  
Науч. рук. **В.Н. Леванцов**, старший преподаватель

## **РАЗРАБОТКА МУЗЫКАЛЬНОГО WEB-СЕРВИСА**

Проект представлен web-сервисом «MusicChart», основной задачей которого является предоставление возможности прослушивания треков. Реализует шаблон проектирования `Model-View-Controller`. Проект написан на языке программирования `Java`, включая `HTML`, `CSS`, `JavaScript`; и использует технологию построения интерактивных пользовательских интерфейсов web-приложений `AJAX`.

Проект состоит из 74 классов, 11 интерфейсов, 15 перечислений, которые в общем состоят из 4126 строчек кода; содержит каскадную таблицу стилей, 4 файла `JavaScript`, 2 вспомогательных файла плагинов, 25 файлов `JSP`.

Любая страница предоставляет ряд возможностей, среди которых общими являются: авторизация, регистрация и переход на страницу треков, исполнителей и жанров. Страницы с треками и актёрами имеют ограничение в 15 элементов на страницу. Вследствие чего возможен постраничный переход. На странице тегов ограничение отсутствует. В проекте выделяется три основных сценария: «трек», «актёр» и «тег».

Главная страница представлена треками и реализует сценарий «трек». При выборе трека начинается его проигрывание в плеере, постоянно находящимся внизу экрана. В неактивном состоянии плеер прозрачен. Плеер использует два вспомогательных файла и предоставляет возможности воспроизведения, паузы, остановки, перемотки, скачивания трека; настройки громкости. По окончании проигрыша трека происходит автоматическое воспроизведение следующего.

Вторая страница содержит список исполнителей и реализует сценарий «актёр». При выборе исполнителя открывается страница, представленная его треками и реализующая сценарий «трек».

Третья страница содержит список жанров. На данной странице предоставлен выбор поиска информации: «по трекам» либо «по актёрам». При выборе жанра в зависимости от вида поиска открывается страница с треками либо исполнителями данного жанра со сценариями «трек» либо «актёр» соответственно.

Авторизованным пользователям могут предоставляться роли пользователя либо администратора. По умолчанию при регистрации пользователю устанавливается в соответствие роль пользователя. Разница между ролями заключается в предоставлении прав доступа к базе данных администратору. Вне зависимости от роли все авторизованные пользователи имеют собственный плейлист, разделённый по трекам, исполнителям и жанрам; и соответственно могут добавлять туда понравившиеся треки, исполнителей либо жанры.

Каждый элемент содержит рейтинг. Рейтинг увеличивается на единицу при добавлении пользователем трека, исполнителя либо жанра в свой плейлист. При удалении рейтинг снижается на единицу.

Данный проект содержит ряд фильтров, отслеживающих корректность вводимых данных. Процессом авторизации пользователю предлагается ввести «Логин» и «Пароль». При отсутствии совпадения пары значений выводится соответствующее сообщение.

Регистрация предоставляет три поля: «Логин», «Пароль», «Повтор пароля». Регистрация пройдёт успешно при одновременном выполнении двух условий: пользователя с введённым логином не существует, пароли в соответствующих полях идентичны.

Интерфейс сервиса представлен на рисунке 1.

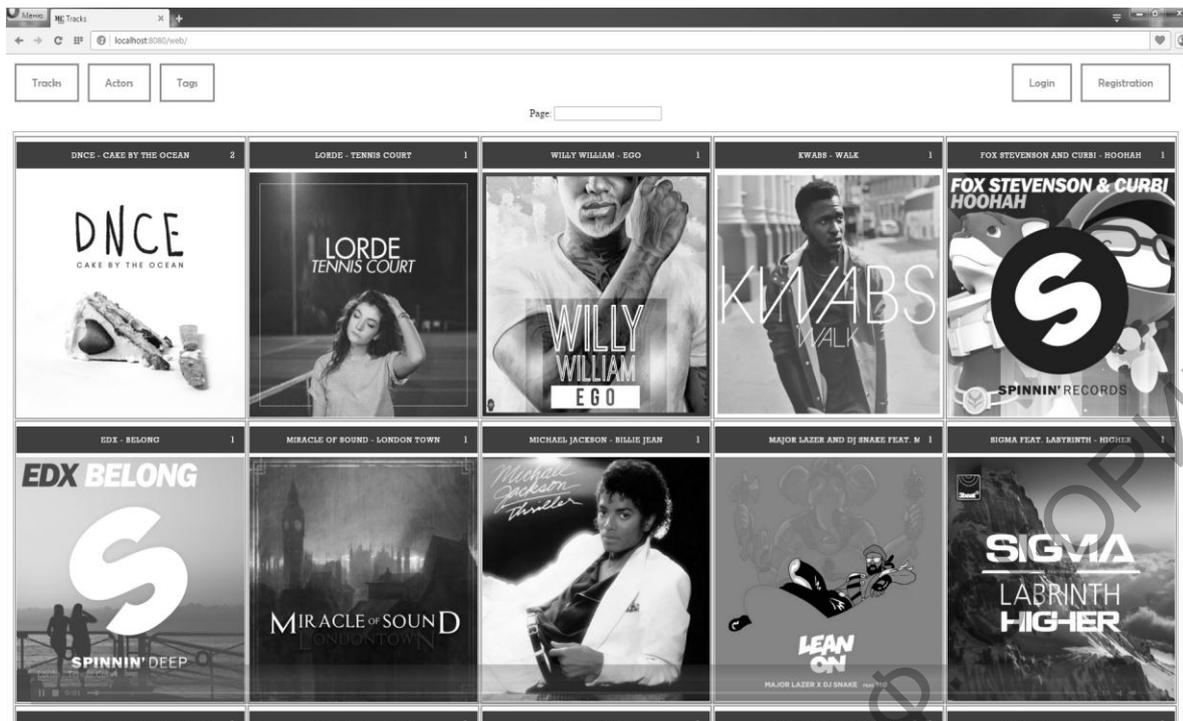


Рисунок 1 – Интерфейс сервиса

В качестве локального сервера использовался «Denwer». Контейнер сервлетов представлен «Arache Tomcat». Web-интерфейс для администрирования базы данных представлен «phpMyAdmin».

Вся информация хранится в базе данных «MusicChart». Схема данных представлена на рисунке 2.

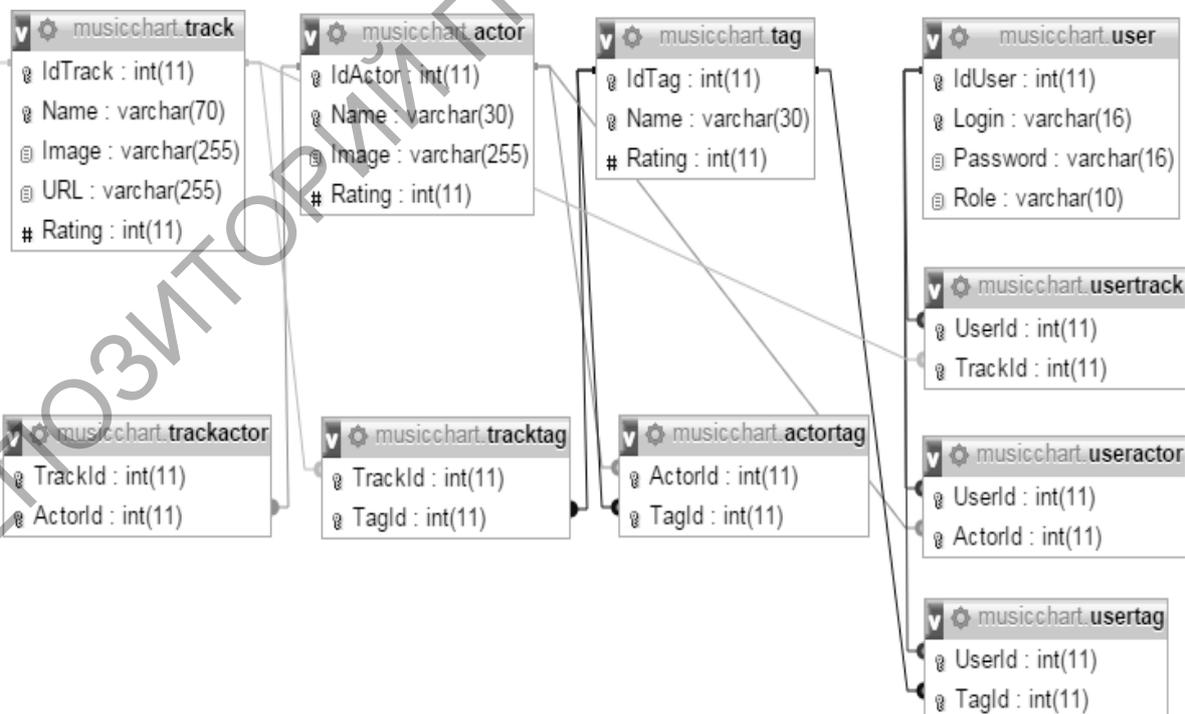


Рисунок 2 – Схема данных

База данных состоит из 10 таблиц, которые условно можно разделить на три логические части: данные («Track», «Actor», «Tag»), связи («TrackActor», «TrackTag», «ActorTag»), пользователь («User», «UserTrack», «UserActor», «UserTag»).

**Н.Н. Коваленко** (УО «ГГУ им. Ф. Скорины», Гомель)

Науч. рук. **А.И. Кучеров**, старший преподаватель

## **КОНФИГУРАЦИЯ ПРОЕКТА МОНИТОРИНГА ВЫПОЛНЕНИЯ ПРАКТИЧЕСКИХ ЗАДАНИЙ УНИВЕРСИТЕТА НА NODE JS**

Node JS – программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API, подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере. В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

Целью данного проекта являлось написание проекта, который упорядочит взаимодействие между обучающимися и преподавателями, Приложение позволяет:

- 1 Вести учёт лабораторных, практических и прочих работ.
- 2 Генерировать различные списки и отчёты об успеваемости.
- 3 Занимать очередь на сдачу какой-либо работы онлайн.
- 4 Записываться в группы к определённым преподавателям.
- 5 Регистрировать новых преподавателей.
- 6 Регистрировать новых обучающихся.

Для реализации вышеописанного функционала потребовалось сконфигурировать сервер на Node JS с использованием библиотек Express JS и Sequelize JS. Sequelize JS выступает в качестве объектно-реляционного преобразователя. Он связывает поля классов, описанных на ECMAScript, с реальными таблицами базы данных. Благодаря этому на сервере мы можем обращаться к таблицам и информации в них с помощью методов list(), save(), update() и т. д.

Библиотека Express JS в приложении служит помощником в предоставлении серверного API. Как только пошлём подобный запрос на сервер, на сервере на него отреагирует соответствующий обработчик, написанный с помощью Express JS. Он выглядит примерно следующим образом: