Министерство науки и высшего образования Российской федерации
ФГБОУ ВО «Рыбинский государственный авиационный
технический университет имени П. А. Соловьева»

Институт информационных технологий и систем управления
Кафедра математического и программного обеспечения электронных вычислительных средств

Направление подготовки
09.04.01 Информатика и вычислительная техника

КУРСОВАЯ РАБОТА

по дисциплине
СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ПРОМЫШЛЕННОЙ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

на тему:

ПРОЕКТИРОВАНИЕ СИСТЕМЫ СТРУКТУРИЗАЦИИ И ХРАНЕНИЯ ДАННЫХ ПЕРСОНАЛА ПРЕДПРИЯТИЯ

Пояснительная записка

Студент группы ИВМ-22 Щербаков М.И.
Руководитель, доцент: Волков М. Л.
Консультант: Петров Н. С.

Рыбинск 2022

Оглавление

Введение

На всех современных предприятиях ведется учет сотрудников. При этому получаемый в результате массив данных требует жесткой структуризации. Бумажный документооборот постепенно заменяется электронным в виду эффективности, надежности и быстродействия последнего.

Целью выполнения курсового проекта является знакомство с системами хранения и обработки баз данных и создание приложения, демонстрирующего возможности электронной системы структуризации.

В виду особенностей выбранного программного обеспечения, разрабатываемая система может дополнятся и видоизменяться, усложняясь и адаптируюсь под нужды конкретного предприятия.

1 Выбор и обоснование программного обеспечения

В качестве основы серверной части в рамках проходимой дисциплины был выбран объектно-ориентированный язык программирования *Java*. Данный язык достаточно широко используется в сфере проектирования серверных приложений и веб-приложений.

В качестве системы управления базой данных (СУБД) выбрана *PostgreSQL*, поддерживающая БД неограниченного размера, надежных механизмов транзакций, поддержки наследования и легкой расширяемости, что позволит в дальнейшем расширить разрабатываемую систему или использовать ее в качестве интегрируемой подсистемы.

Разработка серверной части проводилась в среде *IntelliJ IDEA* с использованием фреймворка *Quarkus (рисунок 1.1)*.

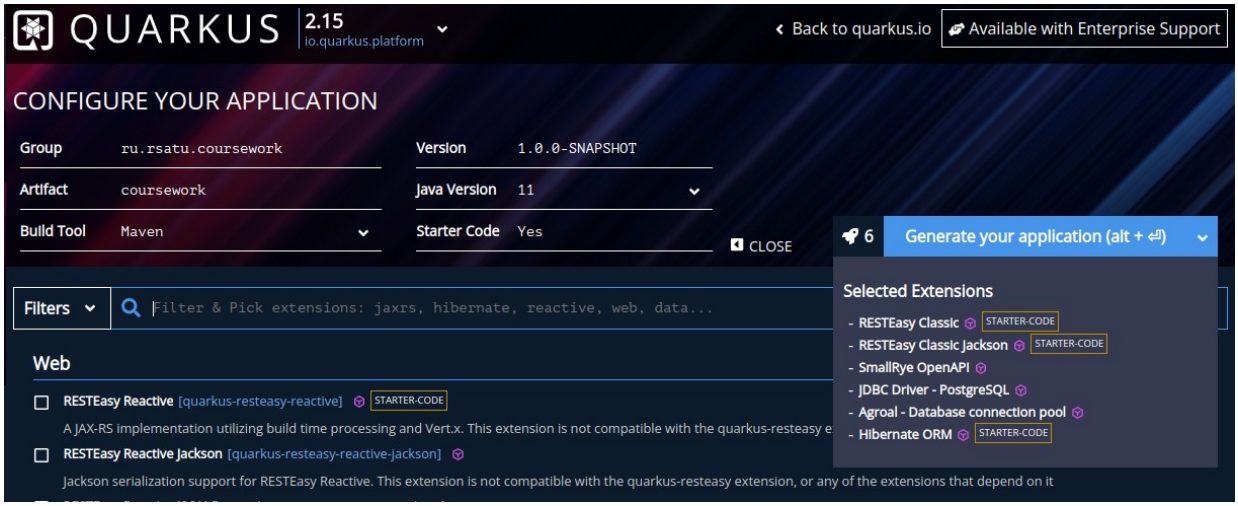

Рисунок 1.1 – Выбор зависимостей проекта на *Quarkus*

Клиентская часть выполнена в виде веб-страницы на *HTML* с использованием *JavaScript* и *Vue.js*.

Организация контроля доступа выполнена с помощью *Keycloak*.

2 Описание Java-сущностей

В таблице 3.1 представлено описание сущностей (таблиц БД).

Сущность	Назначение
Users	Список сотрудников и их данные: ИД, имя, внутренний телефон и номер сектора, где работает сотрудник

В таблице 3.2 представлено описание полей сущностей *Users* (пользователи).

Таблица 3.2 – Описание полей сущности *Users*

Имя поля	Тип	Назначение
id	*Long*	Идентификационный номер пользователя
name	*String*	ФИО сотрудника
phone	*Long*	Дополнительная информация (внутренний телефон)
sector_id	*Long*	Номер сектора (отдела), к которому привязан сотрудник

3 Back-end разработка приложения

Задача back-end разработки сводится к проектированию серверной части. Для этого были реализован ряд классов, осуществляющих доступ к БД. Модели данных описываются в пакете *pojo* («*plain-old-Java-object*», т.е. простые классы), классы для работы с БД находятся в пакете *service*.

Описание таблиц производится с помощью классов *Entity*. Взаимодействие с клиентской частью приложения осуществляют классы *Dto*. Бэкенд не может отдавать данные из репозитория как есть, поэтому *Entity* необходимо преобразовать в *Dto*, для чего разрабатываются *mapper*-классы.

Получение и обработку данных из БД берут на себя классы *repository* и *service*.

Классы, реализующие интерфейсы и взаимодействующие с клиентской частью, помещены в пакет *resource*.

Дерево проекта представлено на рисунке 4.1.

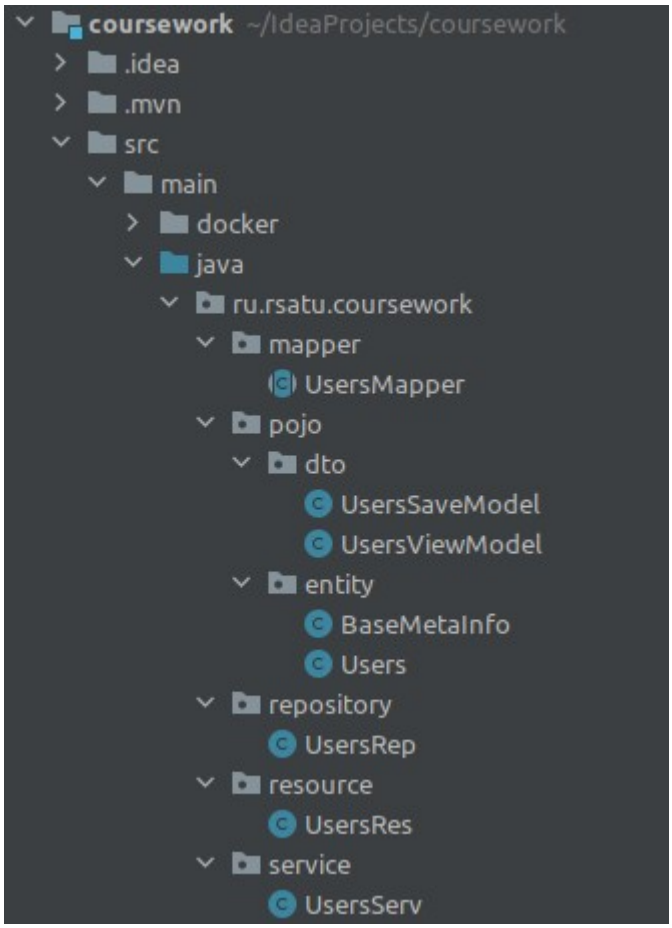

Рисунок 4.1 – Каталог проекта

Параметры соединения с базой данных и параметры подключения к *Keycloak* описываются в *application.properties* – рисунок 4.2.

```
# Datasource
quarkus.datasource.db-kind=postgresql
quarkus.datasource.jdbc.url=jdbc:postgresql://localhost:5432/postgres
quarkus.datasource.username=coursework
quarkus.datasource.password=coursework

quarkus.hibernate-orm.database.generation=update
quarkus.hibernate-orm.log.sql=true

# Keycloak
quarkus.keycloak.devservices.enabled=false
quarkus.oidc.auth-server-url=http://127.0.0.1:8180/realms/coursework

quarkus.swagger-ui.always-include=true
```

Рисунок 4.2 – Параметры подключения к БД

4 Front-end разработка приложения

Front-end разработка сводится к проектированию клиентского приложения, в данном случае реализованного в виде веб-страницы средствами *HTML*, *JavaScript* и *Vue*.

После авторизации пользователь получает доступ к чтению содержимого БД (и записи, если пользователь авторизован как администратор).

На вкладке сотрудники расположена таблица с выводом некоторого количества строк. Навигация по остальной части таблицы осуществляется путем выбора страницы.

5 Организация контроля доступа

На рисунке 6.1 представлены настройки *realm*, примененные к проекту.

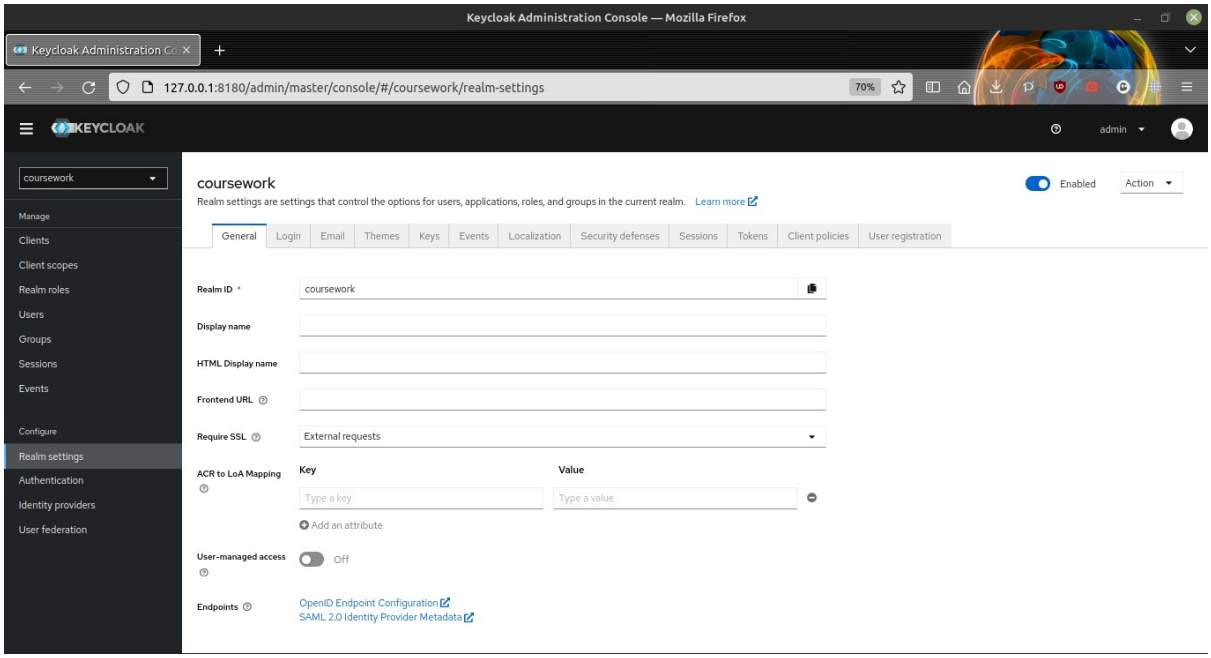

Рисунок 6.1 – Настройка *realm*

На рисунках 6.2 – 6.4 представлены список ролей пользователей и доступные им действия (чтение и запись данных).

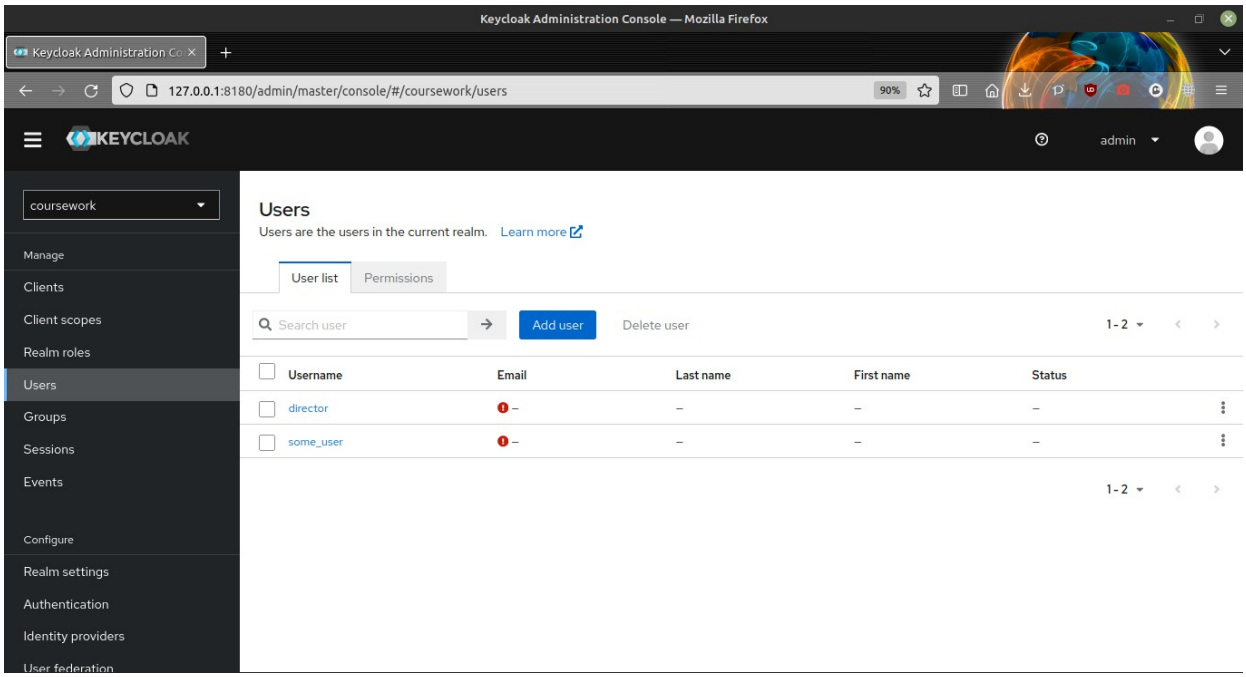

Рисунок 6.2 – Роли пользователей

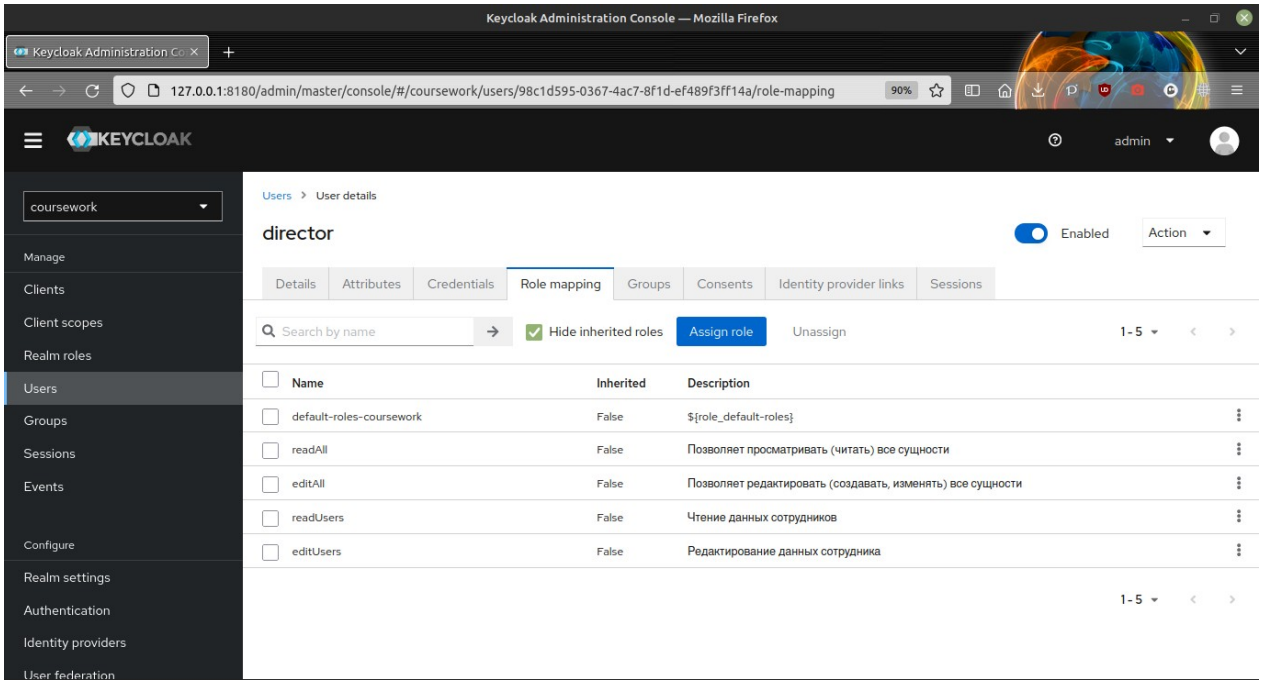

Рисунок 6.3 – Действия, доступные пользователю «*director*» (чтение и запись)

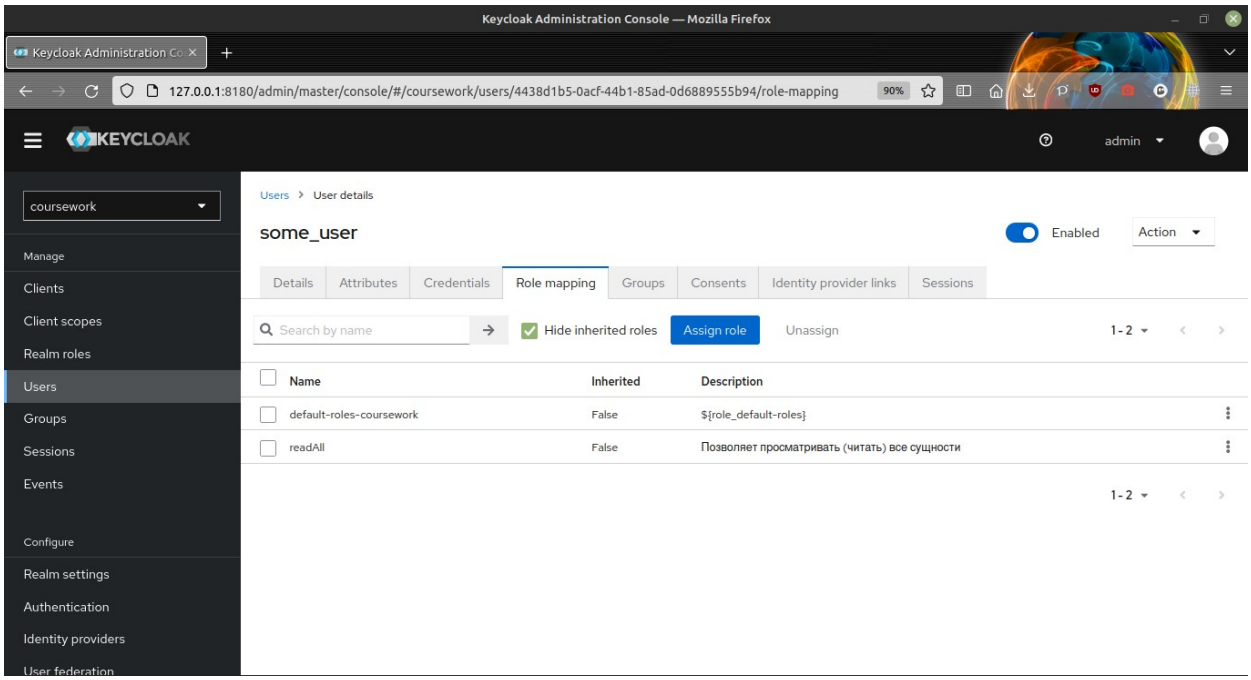

Рисунок 6.4 – Действия, доступные пользователю «*some_user*» (чтение)

Таким образом, с помощью настроек ролей *Keycloak* можно ограничить доступ пользователей к определенным сущностям проекта.

6 Демонстрация работы

Для работы проекта необходимо запустить докер-контейнер с базой данных на порту, прописанном в параметрах подключения сервера, контейнер *keycloak*, сервер приложения и клиентскую часть приложения (рисунок 7.1).

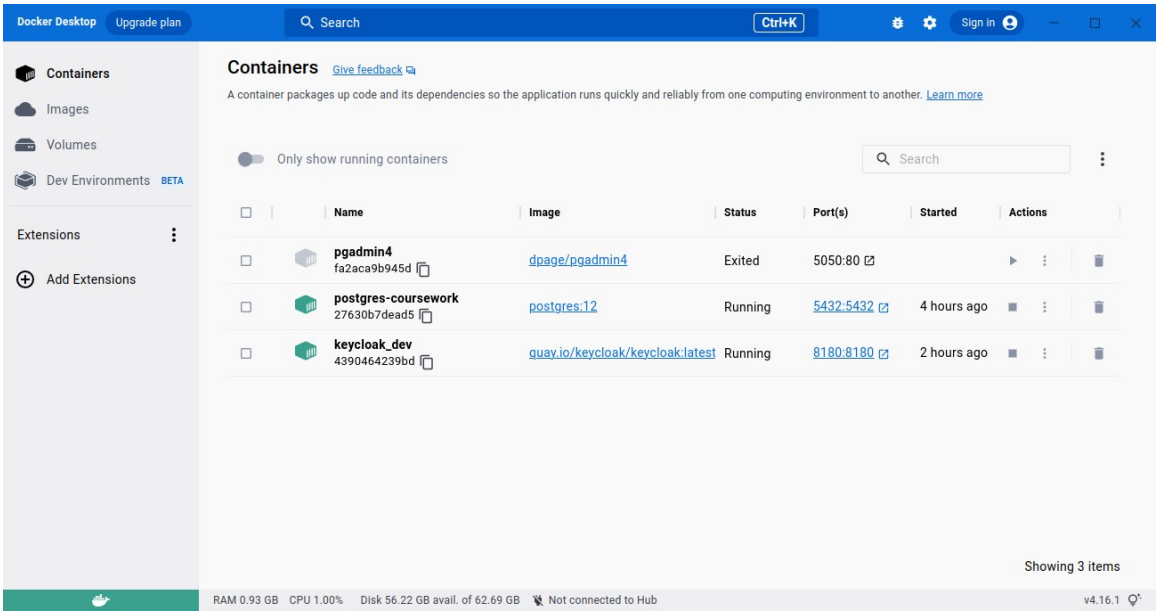

Рисунок 7.1 – запуск *docker*-контейнеров

На рисунке 7.2 представлена вкладка авторизации *Keycloack*. На рисунке 7.3 представлена вкладка Список сотрудников.

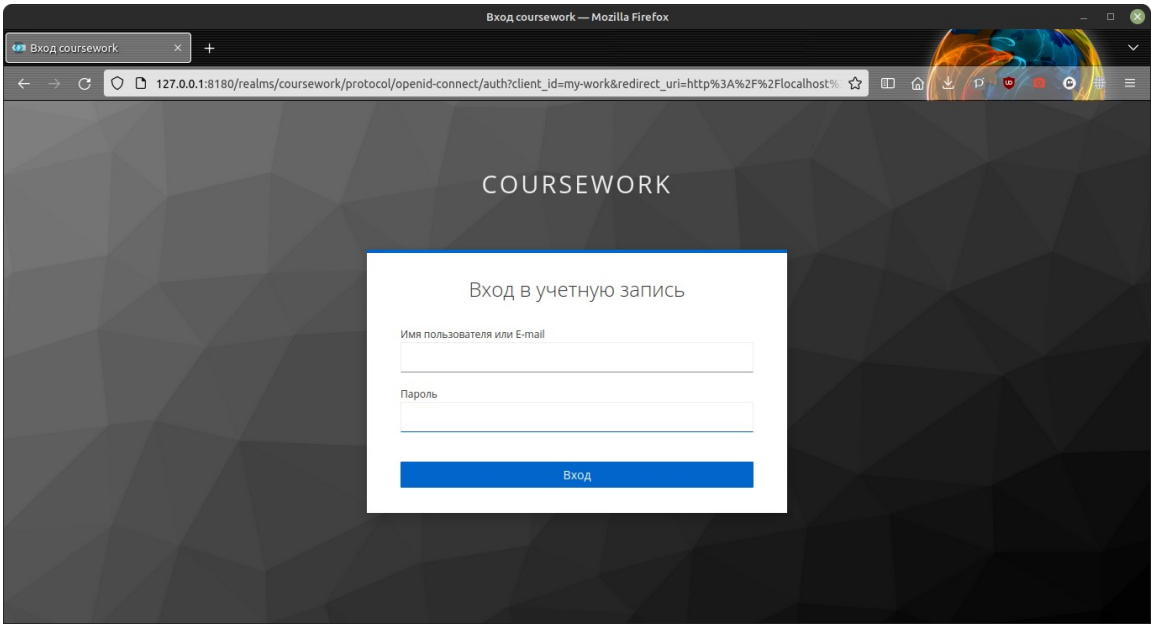

Рисунок 7.2 - Вкладка авторизации *Keycloack*

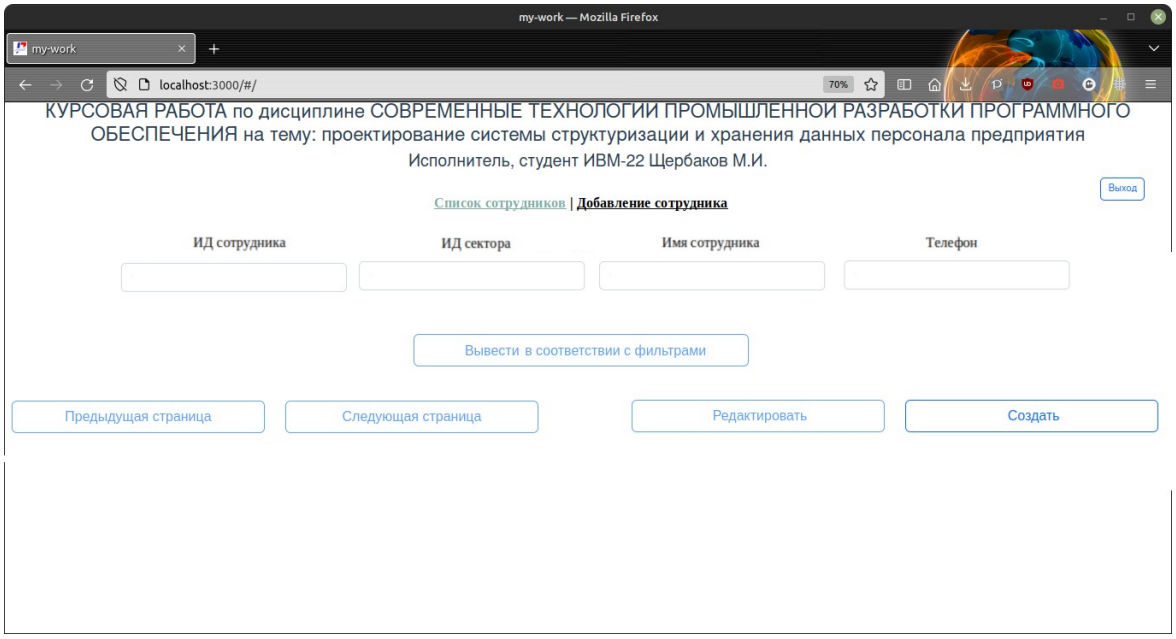

Рисунок 7.3 - Вкладка «Список сотрудников» с возможностью провести фильтрацию, вывести список согласно установленному фильтру, редактировать и создать записи

7 Заключение

В процессе выполнения курсовой работы были получены навыки в организации серверного приложения, проведения *front-end* и back-end разработки.

Были изучены методы работы с базами данных и контейнерами. Изучены методы построения системы авторизации и контроля доступа. Получены навыки в разработки веб-приложений.

В дальнейшем разработанное приложение может дополняться в зависимости от потенциальной сферы применения. Также данное приложение может стать вызываемым модулем другой системы структуризации данных.