

ФГБОУ ВО Уральский государственный горный университет

Инженерно-экономический факультет

Кафедра информатики

Курсовой проект

По дисциплине «Технологии программирования»

На тему «Разработка приложения для "Формирование статистики по оценкам в заданиях"»

Выполнил:

Студентка гр. ИНФ-20-4

Грицюк Н.В.

Проверила:

ст. преп. каф. информатики

Волкова Е.А.

Екатеринбург, 2023г

Оглавление

[1. Постановка задачи 3](#_Toc41836027)

[1.1 Характеристика объекта 3](#_Toc41836028)

[1.2 Потенциал внедрения 3](#_Toc41836029)

[1.3 Краткое техническое задание 3](#_Toc41836030)

[1.3.1 Назначение и цели создания системы 3](#_Toc41836031)

[1.3.1.1 Назначение системы 3](#_Toc41836032)

[1.3.1.2 Цели создания системы 3](#_Toc41836033)

[1.3.2 Требования к системе 3](#_Toc41836034)

[1.3.2.1 Требования к системе в целом 3](#_Toc41836035)

[1.3.2.1.1 Требования к структуре и функционированию системы 3](#_Toc41836036)

[2. Проектирование системы 4](#_Toc41836037)

[2.1 Моделирование системы 5](#_Toc41836038)

[2.2 Проектирование БД 6](#_Toc41836039)

[2.3 Проектирование интерфейса 7](#_Toc41836040)

[3. Разработка системы 8](#_Toc41836041)

[3.1 Выбор средств реализации 8](#_Toc41836042)

[3.2 Этапы реализации 8](#_Toc41836043)

[3.3 Реализация 8](#_Toc41836044)

[3.3.1 Листинг и описание классов 8](#_Toc41836045)

[3.3.2 Интерфейс приложения 11](#_Toc41836046)

# Постановка задачи

Целью этого курсового проекта является создание WEB-приложения для формирования статистики по оценкам в заданиях.

Приложение должно отвечать требованиям к многоуровневой структуре (т.е. иметь как минимум уровень БД, бизнес-логики и, собственно, интерфейс), а также обязательно иметь возможности разграничения доступа (администратор, группы пользователей).

# 1.1 Характеристика объекта

Объектом информатизации является WEB-приложение, которое обрабатывает БД.

# 1.2 Потенциал внедрения

Данное приложение разрабатывается для студентов и преподавателей, которые хотят сэкономить время на создании статистики по общим баллам в заданиях.

В дальнейшем данное приложение будет очень полезно высшим и средним учебным заведениям. Так же его можно будет применить для других различных школ и заведений. Для этого можно быстро разработать бизнес-логику для данных задач.

# 1.3 Краткое техническое задание

Согласно ГОСТ 34.602-89

## 1.3.1 Назначение и цели создания системы

## 1.3.1.1 Назначение системы

Данное приложение предназначена для формирование статистики и предоставления её пользователю.

## 1.3.1.2 Цели создания системы

- обеспечение сбора и первичную обработку исходной информации, необходимой для пользователей.

- предоставление пользователям удобной платформы для формирования статистики с оценками.

## 1.3.2 Требования к системе

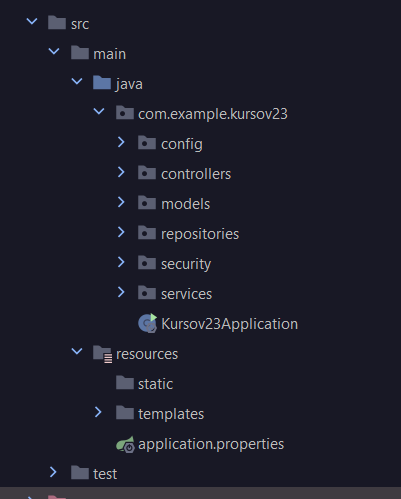
## 1.3.2.1 Требования к системе в целом

## 1.3.2.1.1 Требования к структуре и функционированию системы

Источниками данных для Системы должны быть СУБД MY SQL.

# Проектирование системы

Данный проект имеет следующую структуру данных (рис. 1)



(рис.1)

Папка config содержит в себе служебные классы, такие как разграничение доступа и доступ данным.

В папке controllers написаны классы, предназначенные для непосредственной обработки запросов от клиента и возвращения результатов.

Models содержит в себе описание сущностей БД и их связи.

В папке repositories содержатся интерфейсы обработки данных.

Папка service содержит в себе классы с логикой обработки данных.

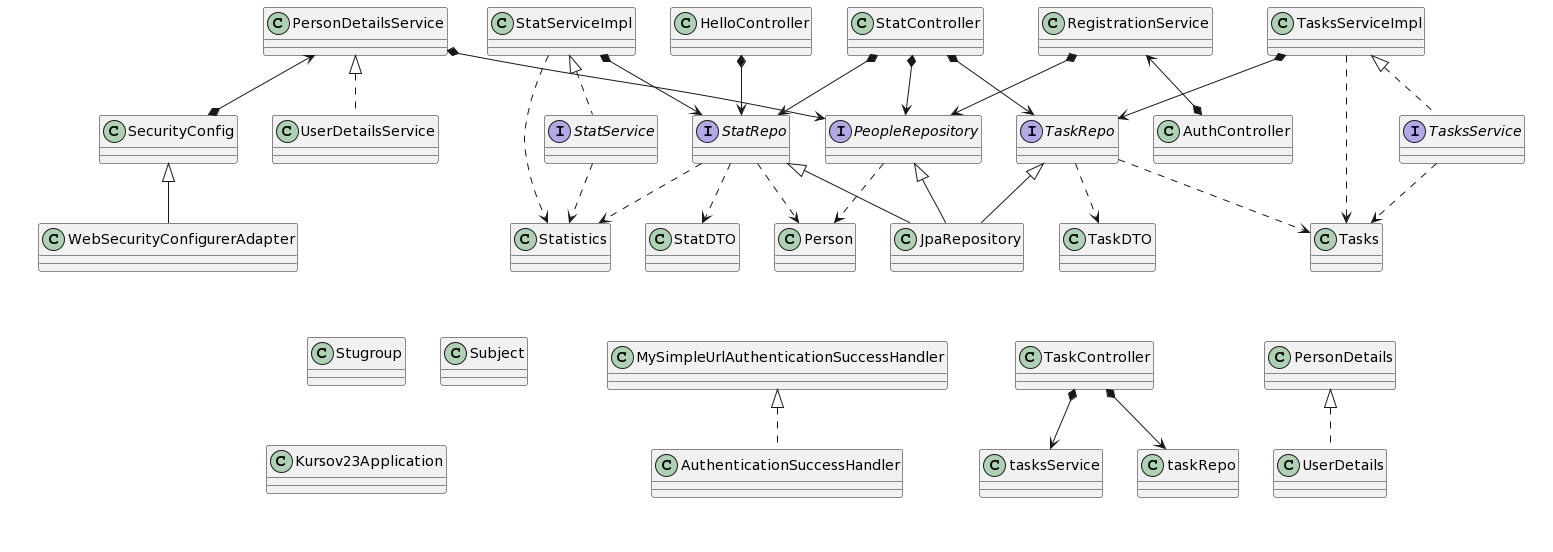
В папке security содержатся классы для работы с spring security.

Папка services содержит в себе классы для работы с БД.

В resoureces находятся шаблоны и статические элементы, которые предназначены для визуализации данных.

# Моделирование системы

UML диаграмма показана ниже (рис. 2)

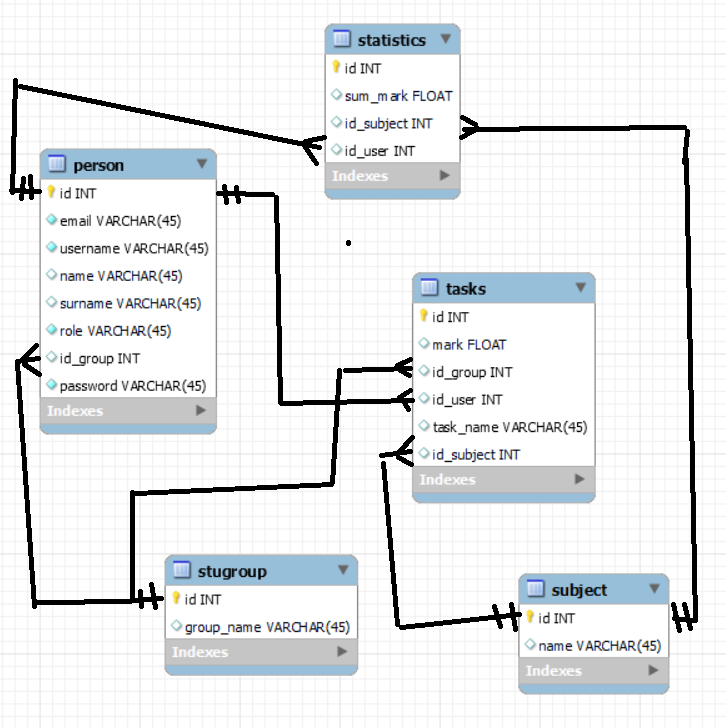


(рис. 2)

# Проектирование БД

В качестве сервера баз был выбран MYSQL версии 8.0.

ER-диаграмма, описывающая структуру БД представлена на (рис. 5)



(рис. 5)

# Проектирование интерфейса

Интерфейс приложения должен содержать навигационную панель. В основной части должен содержаться листинг той или иной информации. В качестве оформления интерфейса будет использован Bootstrap.

# Разработка системы

# Выбор средств реализации

В качестве языка программирования был выбран java, использующее Sping-boot. Такое решение было выбрано, так как на этом языке наиболее удобнее делать web-приложение, также код выполняется достаточно быстро и занимает мало места.

Базой данных будет служить бесплатный сервер баз данных MySQL.

Для оформления HTML страниц и скриптов был выбран Bootstrap, потому что им удобно пользоваться и не нужны дополнительные настройки CSS.

# Этапы реализации

Реализация была разделена на несколько этапов:

* Проектирование структуры БД
* Добавление зависимостей для работы со Spring
* Разработка листинга создания заданий
* Разработка логики добавления оценок и редактирования/удаления заданий
* Разработка листинга статистики
* Разработка логики для формирования статистики
* Реализация разграничений доступа
* Оформление интерфейса
* Разработка логики создания заданий

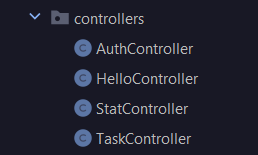
Проект сейчас находится на начале последнего этапа. Изначальный план по разработке приложения «Формирование статистики по оценкам» был выполнен. Поэтому последний этап находится ещё в разработке. Также данный проект можно дополнять ещё многими функционалами, например, чат между пользователями, просмотр решения в заданиях и другое.

# Реализация

# 3.3.1 Листинг и описание классов

Общая структура приложения была приведена на (рис.1).

В данный момент приложение имеет 4 контроллера (рис. 6)



(рис.6)

* AuthController отвечает за все действия связанные с заходом на главную страницу.
* HelloController отвечает за все действия связанные с аутентификацией.
* StatController отвечает за все действия связанные с просмотром и формированием статистики.
* TaskController отвечает за простое создание и взаимодействие с заданиями.

В листинге 1 показан метод формирования статистики.

Листинг 1

@GetMapping("/showStatistics")

public String showStatistics(Model model) {

List<Tasks> tasks = taskRepo.finTasksByGrSub(id\_gr, id\_su);

List<Person> person = peopleRepository.findAllUserByGroup(id\_gr);

System.out.println(tasks.size());

System.out.println(person.size());

if (!person.isEmpty()){

for (int i = 0; i < person.size(); i++){

float count = 0.0f;

for (int j = 0 ; j < tasks.size(); j++){

if ((tasks.get(j).getId\_user() == person.get(i).getId())

&& (tasks.get(j).getId\_group() == person.get(i).getGroup())){

count = count + tasks.get(j).getMark();

}

}

Statistics st = statRepo.finStatBySuUs(person.get(i).getId(), id\_su);

if (st != null){

st.setSum\_mark(count);

statRepo.save(st);

} else {

Statistics newStat = new Statistics();

newStat.setSum\_mark(count);

newStat.setId\_subject(id\_su);

newStat.setId\_user(person.get(i).getId());

statRepo.save(newStat);

}

}

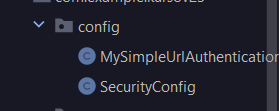
}

model.addAttribute("statList1", statRepo.finStatBySuGr(id\_su, id\_gr));

return "showStatistics";

}

В папке config содержится 2 класса (рис. 7)



(рис. 7)

* MySimpleUrlAuthenticationSuccessHandler служит для перенаправления пользоателя на ту или иную страницу после входа в систему (это зависит от роли).
* SecurityConfig служит за разграничения доступа пользователей.

В листинге 2 представлен класс WebSecurityConfig

Листинг 2

**package com.example.kursov23.config;**

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

private final PersonDetailsService personDetailsService;

@Autowired

public SecurityConfig(PersonDetailsService personDetailsService) {

this.personDetailsService = personDetailsService;

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeRequests()

.antMatchers("/admin/\*\*").hasRole("ADMIN")

.antMatchers("/login", "/error").permitAll()

.anyRequest().hasAnyRole("USER", "ADMIN")

.and()

.formLogin().loginPage("/login")

.loginProcessingUrl("/process\_login")

.successHandler(myAuthenticationSuccessHandler())

.failureUrl("/login?error")

.and()

.logout()

.logoutUrl("/logout")

.logoutSuccessUrl("/login");

}

// Настраивает аутентификацию

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(personDetailsService);

}

@Bean

public AuthenticationSuccessHandler myAuthenticationSuccessHandler(){

return new MySimpleUrlAuthenticationSuccessHandler();

}

@Bean

public PasswordEncoder getPasswordEncoder(){

return NoOpPasswordEncoder.getInstance();

}

}

На данный момент в папке service находится только 6 классов:

* PersonDetailsService, который нужен для работы с пользователем.
* RegistrationService, отвечающий за регистрацию.
* StatService для обращения к бд.
* StatServiceImpl, который нужен для работы с статистикой.
* TasksService для обращения к бд.
* TasksServiceImpl, который нужен для работы с заданями

В листинге 3 показан метод добавления пользователя.

Листинг 3

@Service

public class RegistrationService {

private final PeopleRepository peopleRepository;

@Autowired

public RegistrationService(PeopleRepository peopleRepository) {

this.peopleRepository = peopleRepository;

}

@Transactional

public void register(Person person) {

person.setPassword(person.getPassword());

person.setRole("ROLE\_ADMIN");

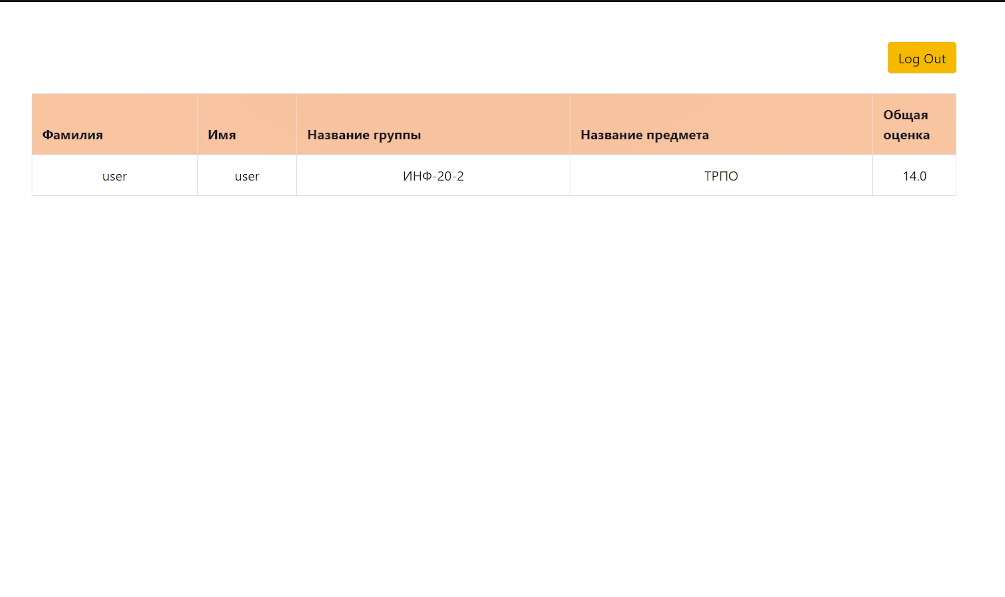
peopleRepository.save(person);

}

}

# 3.3.2 Интерфейс приложения

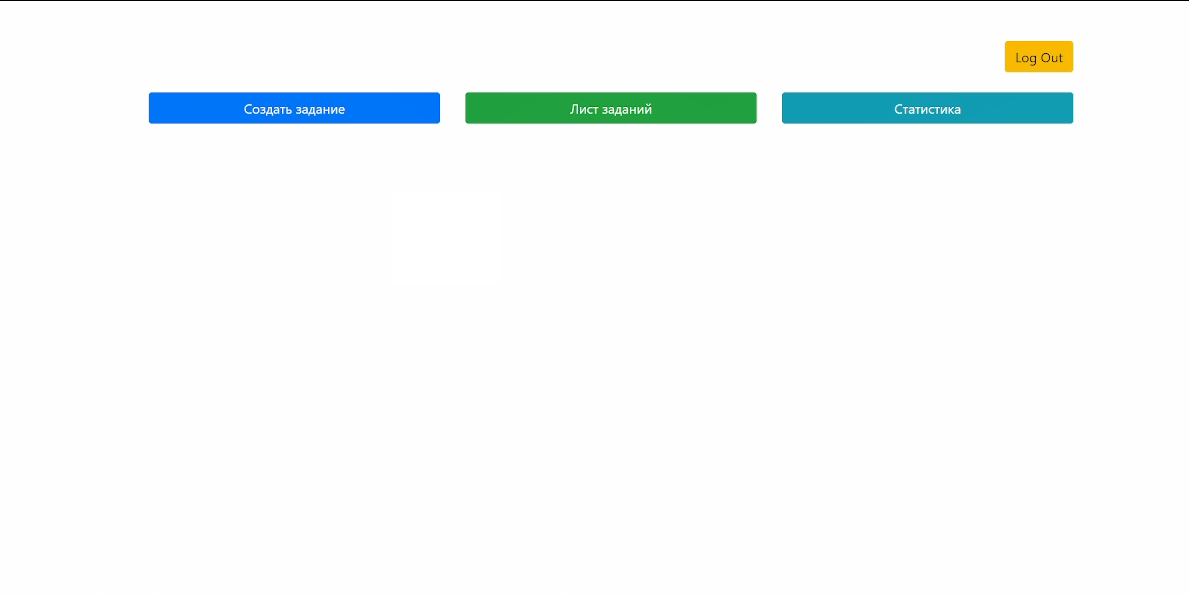
На (рис. 8) показана страница со статистикой.



(рис. 8)

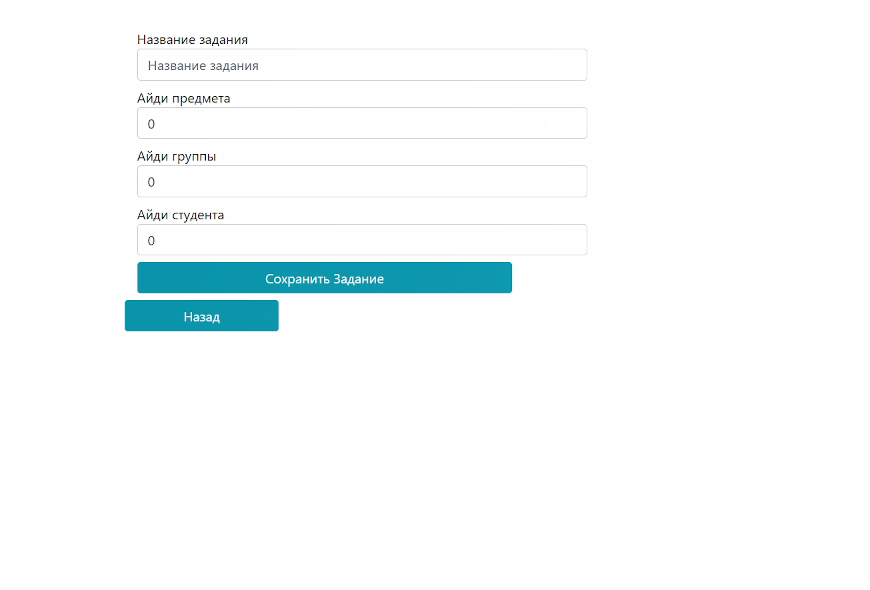
Как можно видеть мы зашли за пользователя с именем user и просматривать существующую статистику по предметам.

Как видим на (рис. 9) у пользователя Admin куда больше возможностей. Он имеет право создавать и редактировать задания, а так же просматривать интересующую статистику. Теперь нажмём на кнопку редактировать и отредактируем эту опечатку.

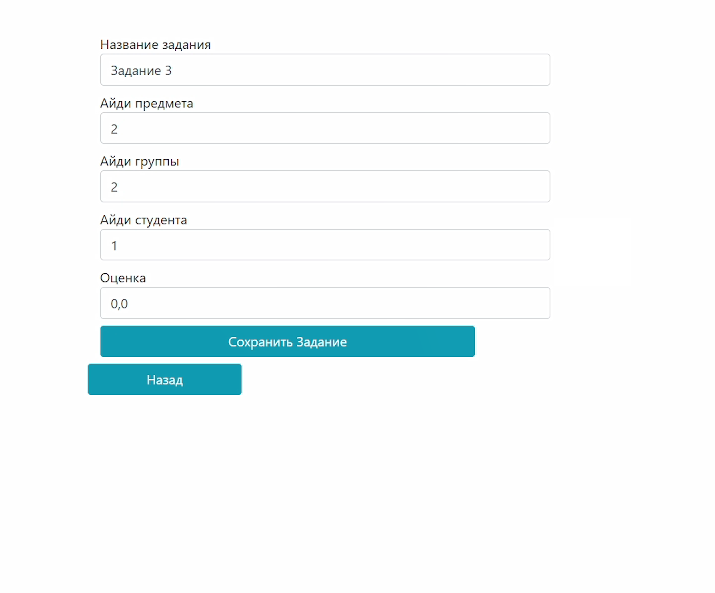


(рис.9)

Для редактирования и добавления заданий используется такая форма ввода (рис. 10, 11). После нажатия кнопки «Сохранить задание» применяются все изменения, которые были записаны в форме.



(рис. 10)



(рис. 11)