МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ**

**ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ**

**ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра компьютерных образовательных технологий

Направление: нейротехнологии и программирование

Разработка программного обеспечения информационной системы

Выполнил студент

Группы P3222

Шимохина Т.А.

Проверил:

Ефимчик Е.А.

Санкт-Петербург

2019 г.

Оглавление

[Введение 3](#_Toc534984564)

[Основная часть 3](#_Toc534984565)

[Анализ задачи и обзор аналогов 3](#_Toc534984566)

[Задачи 3](#_Toc534984567)

[Обзор аналогов 3](#_Toc534984568)

[Расширение задания 4](#_Toc534984569)

[Роли пользователей 4](#_Toc534984570)

[Прецеденты использования 5](#_Toc534984571)

[Проектирование базы данных 5](#_Toc534984572)

[Список сущностей, свойств и связей между ними с описанием 5](#_Toc534984573)

[ER – диаграмма 6](#_Toc534984574)

[Дата-логическая модель 7](#_Toc534984575)

[Проектирование пользовательских интерфейсов 7](#_Toc534984576)

[Реализация базы данных 9](#_Toc534984577)

[Реализация веб-приложения 9](#_Toc534984578)

[Пользовательские интерфейсы 9](#_Toc534984579)

[Подключение к БД 9](#_Toc534984580)

[Сущности и методы 9](#_Toc534984581)

[Контроллеры 10](#_Toc534984582)

[Запуск приложения 11](#_Toc534984583)

[Документация 11](#_Toc534984584)

[Заключение 11](#_Toc534984585)

[Список источников 11](#_Toc534984586)

Введение

По итогу курсовой работы будет создано многопользовательское веб-приложение с подключением базы данных.

Основная часть

Анализ задачи и обзор аналогов

# Задачи

Необходимо разработать информационную систему с элементами социальной сети в виде веб-приложения. Должны быть реализованы:

1. возможность комментирования;
2. добавление/удаление друзей;
3. публикация информации (по типу постов или твитов).

# Обзор аналогов

Рассмотрим некоторые примеры систем, на основе выделенных особенностей которых будет строиться будущее приложение.

Первое приложение - Joe’s Goals, позволяет ставить задачи и отмечать их выполнение, однако отсутствует возможность поделиться ею с друзьями и комментировать.

Второе приложение – Steemol. Также присутствует возможность постановка целей и их распределение по категориям. Формируется список активных пользователей, но пользователю доступны только свои задачи.

Третье приложение – Twitter. Здесь нас интересует система твитов, списков друзей, комментирования и оценивания.

Результаты сравнения – в таблице 1 (обозначения: “+” – присутствует,

“-” – отсутствует, “+-” – присутствует частично).

Таблица 1 – Сравнение аналогов

|  |  |  |  |
| --- | --- | --- | --- |
|  | Joe’s Goals | Steemool | Twitter |
| Постановка целей | + | + | - |
| Возможность комментирования | + | - | + |
| Система оценивания | - | - | +- |
| Разделение целей по времени исполнения | + | + | - |
| Добавление друзей | - | - | +- |
| Формирование рейтинга | - | +- | - |

# Расширение задания

Будем разрабатывать веб - приложение. Пользователь может устанавливать цели (на день, неделю, месяц и т.д.) и время ее выполнения. По истечении установленного срока необходимо «закрыть» цель, т.е. выложить некое доказательство ее выполнения. Это может быть фото, ссылка на сторонний ресурс.

Другие пользователи могут оценить качество ее выполнения (10-ти балльная система). Все оценки и время выполнения (опоздал/в срок) учитываются в общем рейтинге.

Пользователи могут добавлять друзей.

# Роли пользователей

Пользователи приложения могут быть:

* Гостем;
* Администратором;
* Участником (зарегистрированным пользователем);

Действия, доступные для каждого участника приложения, представлены на рис. 1.

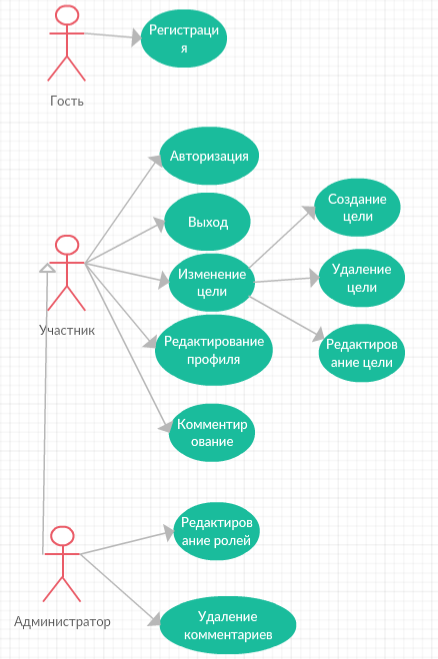


Рисунок 1 – Диаграмма использования

# Прецеденты использования

1. Регистрация (смотри таблицу 2);
2. Авторизация (смотри таблицу 3);
3. Редактирование профиля (смотри таблицу 4);
4. Редактирование целей
   1. Создание цели (смотри таблицу 5);
   2. Выполнение цели (смотри таблицу 6);
   3. Удаление цели (смотри таблицу 7);
5. Комментирование (смотри таблицу 8);
6. Оценивание (смотри таблицу 9);
7. Удаление профиля (смотри таблицу 10);
8. Изменить роль пользователя (смотри таблицу 11);
9. Удаление комментария (смотри таблицу 12);

Проектирование базы данных

# Список сущностей, свойств и связей между ними с описанием

Список сущностей со связями и свойствами представлен в таблице 13.

Таблица – Сущность, ее свойства и связи

|  |  |  |
| --- | --- | --- |
| Сущности | Свойства | Связи |
| User  (Пользователь) | Nickname  Avatar (фото)  Email  Username  Password | Добавление в друзья (П – П)  Начисление очков (П) |
| Goal  (Цель) | Text  DateOfCreation (дата создания)  SetTime (установленная дата выполнения)  DateOfComp(дата выполнения)  Evidence (доказательство выполнения) | Комментирование (П – Ц)  Авторство (П – Ц)  Оценивание (П – Ц) |

# ER – диаграмма

ER – диаграмма представлена на рисунке 2.

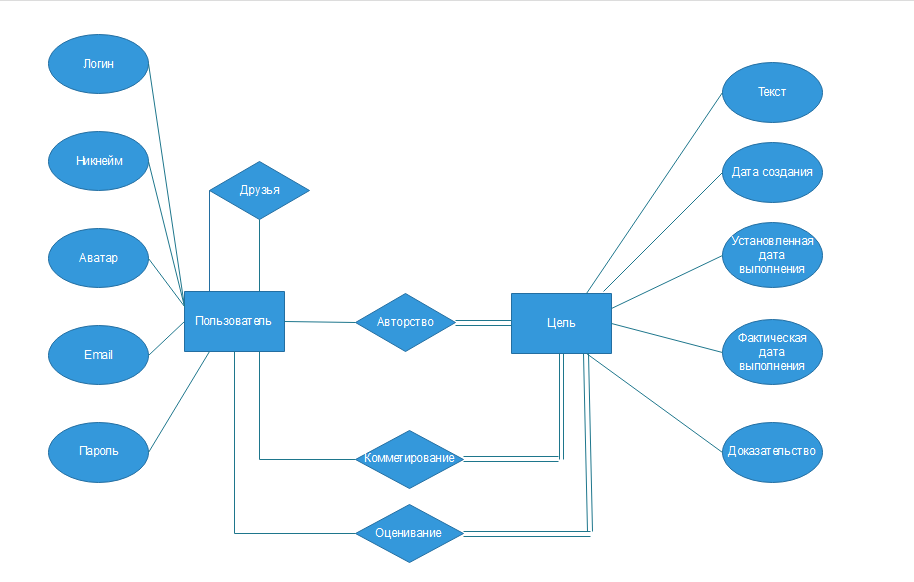


Рисунок – 2 – ER-диаграмма

# Дата-логическая модель

Дата-логическая модель представлена на рисунке 3.

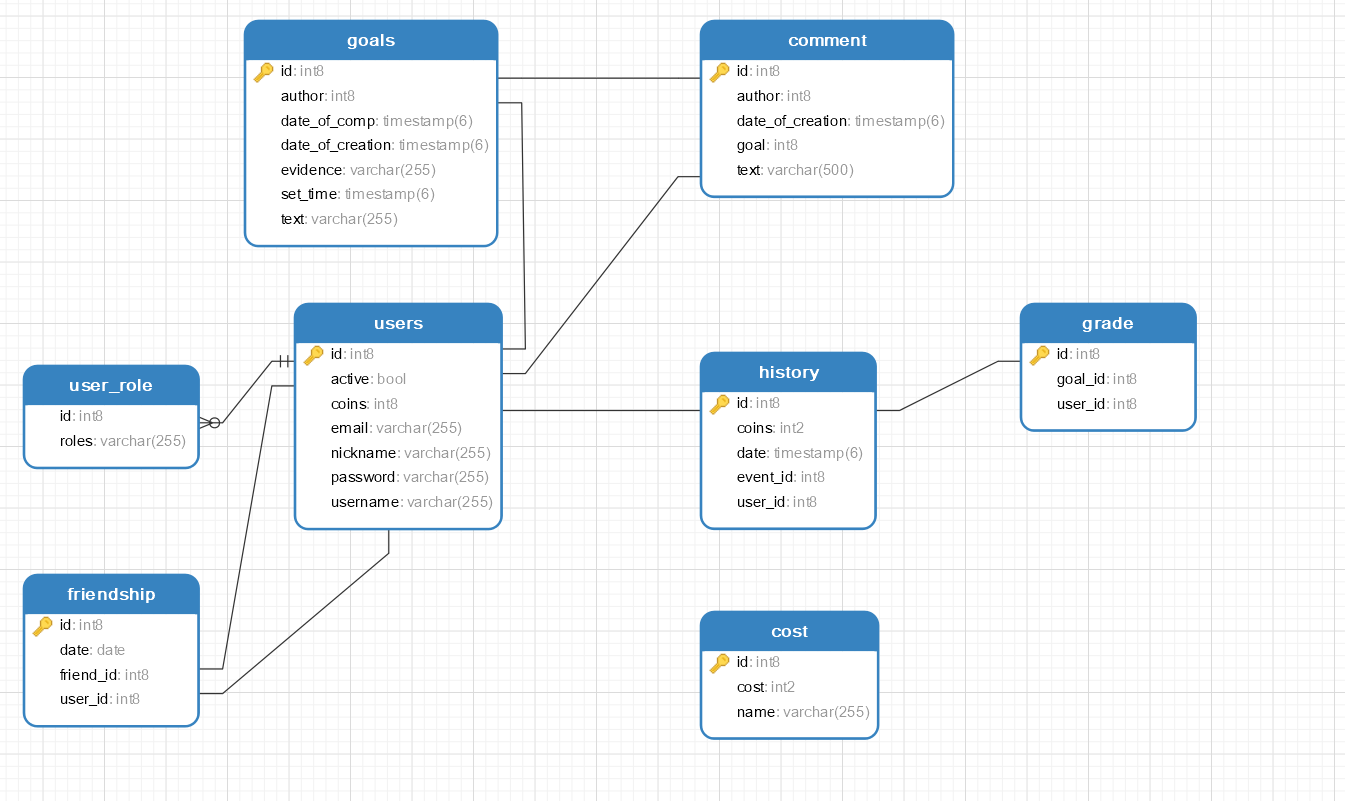


Рисунок – 3 – Дата-логическая модель

Проектирование пользовательских интерфейсов

На данном этапе были разработаны шаблоны страниц проектируемого приложения.

Диаграмма переходов (State Diagram) представлена на рисунках 4.1. На каждой странице есть панель меню для переходов на страницы. Пример таких переходов отражен на рисунке 4.2.

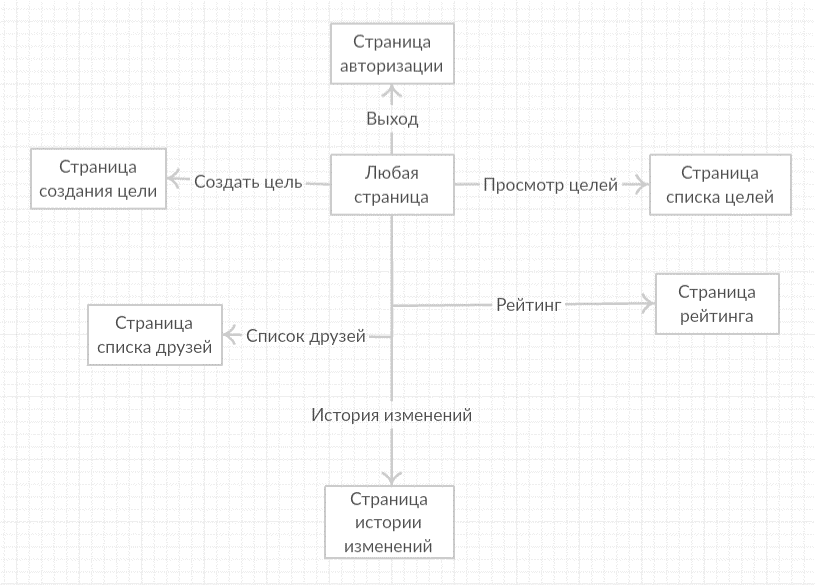


Рисунок – 4.2 – Дополнительная диаграмма состояний

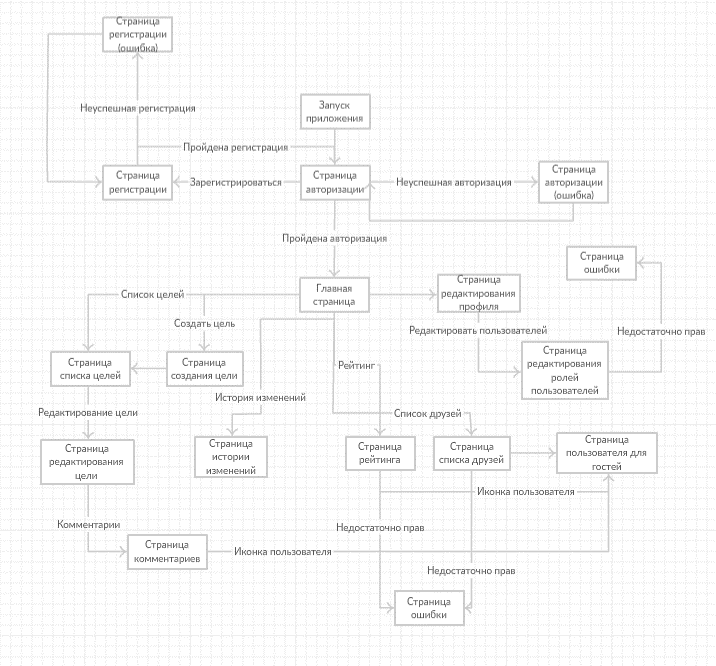


Рисунок – 4.1 – Основная диаграмма состояний

Были спроектированы следующие макеты страниц (все они представлены в приложении):

1. Главная страница приложения (рисунок 5);
2. Главная страница приложения (Ошибка) (рисунок 6);
3. Страница регистрации (рисунок 7);
4. Страница регистрации (Ошибка) (рисунок 8);
5. Главная страница пользователя (рисунок 9);
6. Редактирование профиля (рисунок 10);
7. Создание цели (рисунок 11);
8. Редактирование цели (рисунок 12);
9. Список целей (рисунок 13);
10. Список друзей (рисунок 14);
11. Рейтинг пользователей (рисунок 15);
12. Страница пользователя для гостей (рисунок 16);
13. Комментарии (рисунок 17);
14. Редактирование ролей пользователей (рисунок 18);
15. История изменений (рисунок 19);
16. Ошибка прав доступа пользователя (рисунок 20).

Реализация базы данных

Так как в проекте для работы с базами данных используется Spring Data Jpa, требуется лишь частичная ее реализации в виде скрипта SQL (run.sql в приложении).

Реализация веб-приложения

Приложение разработано на языке Java EE. Архитектура приложения основана на Spring MVC: существуют контроллеры, слушающие запросы и передающие представления.

При разработке приложения были использованы:

* Spring Boot;
* Spring Boot Security;
* Spring Boot Data Jpa;
* Thymeleaf;
* PostgreSQL;
* Devtools;
* Tomcat;
* Bootstrap.

# Пользовательские интерфейсы

Пользовательские интерфейсы реализованы с помощью Thymeleaf (html – шаблоны) и Bootstrap (стилизация). Все, что касается непосредственно шаблонов, хранится в директории resources. Стили и иконки – в static/res, шаблоны страниц – в templates.

# Подключение к БД

Подключение к БД осуществляется с помощью Spring Data Jpa и соответствующими настройками в файле application.properties.

# Сущности и методы

Взаимодействие с базой данных осуществляется с помощью создания классов – сущностей в директории Entities с соответствующей аннотацией. Для каждой сущности (кроме Enum Role, так как она заполняется посредством сущности User) созданы интерфейсы – репозитории в директории Repositories для сохранения и поиска в БД. Для удобства был создан UserService в директории Services, так как логика работы с UserRepository встречается очень часто.

Для “прослушивания” приходящих запросов созданы классы – контроллеры в директории Controllers, получающие запросы, реализующие соответствующую логику и возвращающие пользователю запрашиваемые страницы. Некоторые из них имеют аннотации для запрета доступа для пользователей с недостаточными правами.

Как правило, именно контроллеры взаимодействуют со всеми представленными классами. Примерная схема приведена на рисунке 21.

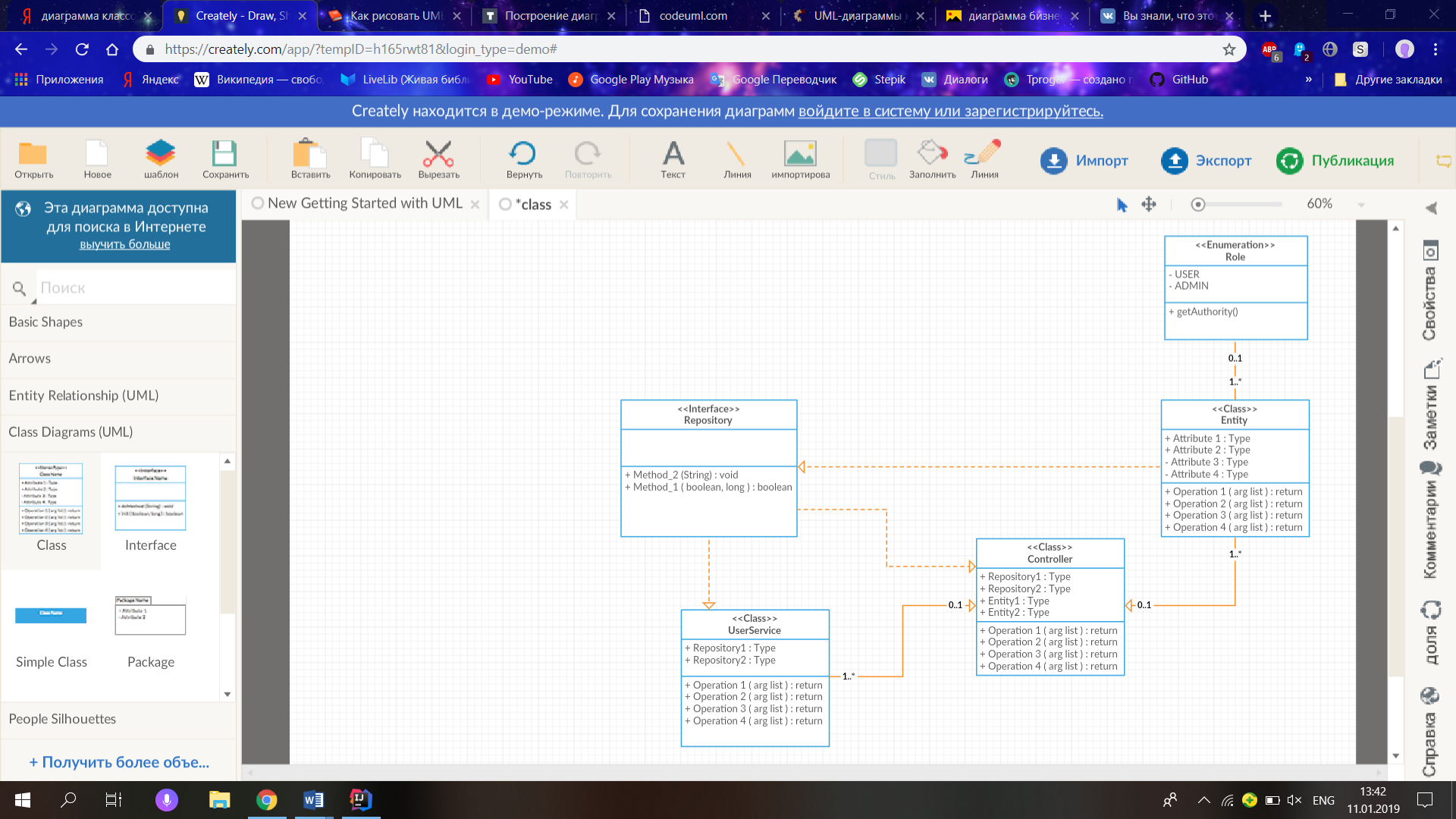


Рисунок - 21 – Взаимодействие классов приложения

# Контроллеры

В таблице 14 приведены контроллеры приложения и логика, которую они реализуют.

Таблица – Контроллер с обработкой

|  |  |
| --- | --- |
| *Контроллер* | *Что реализует* |
| CommentControler | Обработка действий с комментариями. |
| ExceptionController | Обработка ошибки 403. |
| FriendsController | Обработка добавления в друзья и просмотр списков. |
| GoalsController | Обработка создания и редактирования цели. |
| HistoryController | Выдача истории изменения количества баллов. |
| MainController | Обработка информации из профиля текущего пользователя. |
| RaitingController | Формирование рейтинга пользователей. |
| RegistrationController | Обработка регистрации нового пользователя. |
| UserController | Отвечает за изменение ролей пользователей. |
| ViewUserController | Выдача информации о пользователей для гостей. |

# Запуск приложения

1. Установите PostgreSQL, создайте базу данных «my-application». Установите для нее логин: «postgres», пароль: «root» (без кавычек).
2. Запустите SQL-скрипт, прилагаемый в архиве.
3. Установите JRE с сайта <https://www.java.com/ru/download/>.
4. Создайте на диске C папку «avatars».
5. Скачайте Tomcat (<https://tomcat.apache.org/download-90.cgi>), установите логин и пароль. Запомните их.
6. Скачайте Gradle (<https://gradle.org/install/>) и установите соответствующие переменные среды.
7. Запустите командную строку, перейдите в главную директорию «course\_work» и введите команду «gradle bootRun». Нажмите Enter.
8. Откройте браузер и перейдите по адресу «localhost:8080». Приложение запущено.
9. Изначально создан пользователь admin с паролем admin (рекомендуется сменить пароль после первого запуска) с правами администратора. В дальнейшем права администратора могут быть даны другим пользователям. Не рекомендуется удалять пользователя admin!
10. По окончанию работы закройте консоль. Приложение будет остановлено.

Документация

Отчет по курсовой работе содержит пояснительную записку, приложение A и приложение B отдельными документами, SQL – скрипт для выполнения перед запуском, а также диск с реализованным приложением.

Заключение

По итогам курсовой работы было разработано многопользовательское приложение Goal App с возможностью взаимодействия пользователей.

Были изучены технологии сборки проектов (Maven и Gradle), изучена работа Spring Boot и частично его подсистем (MVC, Security, Data Jpa).

Для создания шаблонов был выбран Thymeleaf ввиду подходящего способа обработки моделей, для стилизации – Bootstrap, функций которого оказалось достаточно для данного проекта.

Список источников

* Компания Pivotal / Guides for Spring / URL = [https://spring.io/guides / 11.01.2019 14:00](https://spring.io/guides%20/%2011.01.2019%2014:00);
* Software Foundation Apache / Thymeleaf Tutorial: Using Thymeleaf / URL = [https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html /](https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html%20/) 11.01.2019 14:00;
* Gradle Inc / Docs for Gradle / URL = [https://docs.gradle.org/current/userguide/userguide.html / 11.01.2019 14:00](https://docs.gradle.org/current/userguide/userguide.html%20/%2011.01.2019%2014:00);
* Bootstrap Team / Bootstrap Documentation / URL = <https://getbootstrap.com/docs/4.2/getting-started/introduction/> / 11.01.2019 14:00;
* Влад Мержевич / Справочник по HTML / URL = <http://htmlbook.ru/html> / 11.01.2019 14:00;