МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ **«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №4  
по дисциплине: Технологии Web-программирования

«Разработка и проектирование базы данных»

Выполнил: ст. гр. ПВ-41

Мишина Ю.А.

Проверил: Картамышев С. В.

Белгород 2020г.

**Цель**: изучить основы взаимодействия web-приложения с базой данных. Спроектировать базу данных для хранения информации приложения (страницы, пользователи и т.п.).

**Задание к лабораторной работе:**

1. Выбрать подходящую СУБД.
2. Изучить методы взаимодействия web-приложения с базой данных (ORM, Active Record).
3. Разработать структуру базы данных.
4. Разработать соответствующие модели в приложении.
5. В отчёт приложить схему базы данных, а также код одной из моделей (на своё усмотрение).

**Выполнение**

В данной лабораторной работе мы будем реализовывать бекенд приложение для работы с базой данных с тремя сущностями

Язык программирования: Java 8

Фреймворк: Spring, Spring Boot, JPA, Hibernate

Система сборки: Maven

База данных: PostgeSQL

После установки java8, maven и PostgeSQL

Создаём новый проект с такой структурой:

pom.xml – файл в котором будут находится все необходимые нам зависимости

App – класс, точка запуска приложения (настройка конфигурации и многое другое)

entity – пакет, в котором будут лежать наши сущности

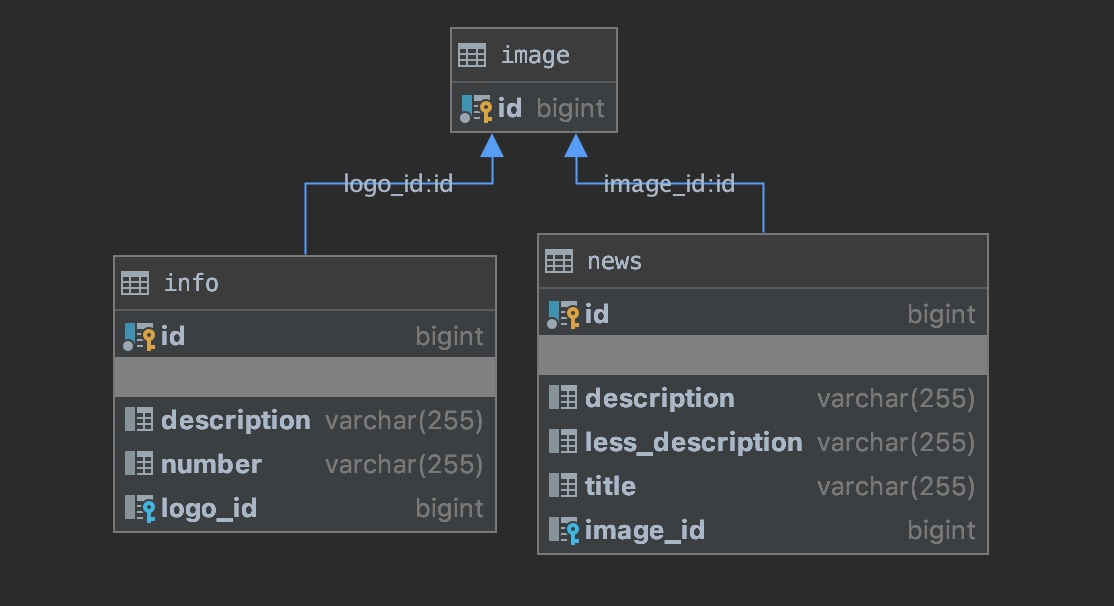
repository – пакет, в котором будут лежать интефейсы для работы с базой данных.

**Подключение необходимых библиотек**

Нам понадобиться библиотека для работы с PostgreSQL и ORM JPA

<dependencies>  
 . . .  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.postgresql</groupId>  
 <artifactId>postgresql</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
 </dependencies>

**Структура базы данных**

****

**Создание сущностей**

Используя аннотации Hibernate для создания и настройки сущностей задаём имя колонок, связи многие ко многим.

@Data  
@Entity  
@Table(name = "news")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
@EqualsAndHashCode  
public class News {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 @Column(name = "id")  
 Long id;  
  
 @Column(name = "title")  
 String title;  
  
 @Column(name = "less\_description")  
 String lessDescription;  
  
 @Column(name = "description")  
 String description;  
  
 @OneToOne  
 Image image;  
}

@Data  
@Entity  
@Table(name = "info")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
@EqualsAndHashCode  
public class Info {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 @Column(name = "id")  
 Long id;  
  
 @Column(name = "number")  
 String number;  
  
 @OneToOne  
 Image logo;  
  
 @Column(name = "description")  
 String description;  
}

@Data  
@Entity  
@Table(name = "image")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
@EqualsAndHashCode  
public class Image {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 @Column(name = "id")  
 Long id;  
}

**Создание JPA репозиториев**

Мы будем использовать JPA интерфейы, которые под капотом используют ORM Hibernate.   
  
В этих интерфейсах уже реализованы базовые CRUD операции, а нам больше и не нужно. Так же при запуске приложение hibernate самостоятельно по методанным создаст таблицы в базе данных.  
  
Чтобы создать репозиторий всего лишь нужно создать интерфейс и наследоваться от одного из JPA интерфейсов.

@Repository  
public interface NewsRepository extends JpaRepository<News, Long> {  
}

Так же мы поступаем с остальными сущностями.

**Работа с PostgeSQL**

Чтобы работать с БД необходимо в конфигурации прописать настройки, куда подключаться.

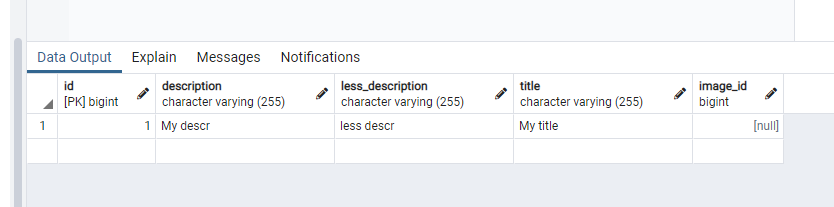
server:  
 port: 8080  
  
spring:  
 datasource:  
 url: jdbc:postgresql://localhost:5432/postgres  
 username: postgres  
 password: 2514  
 jpa:  
 generate-ddl: true  
 show-sql: false

**Демонстрация работы**

Чтобы продемонстрировать работу приложения был написан тест, который сохраняет данные в базу.

class NewsRepositoryTest extends BaseTest {  
  
 @Autowired  
 private NewsRepository newsRepository;  
  
 @Test  
 void testAddAndReadNewsFromDB() {  
 newsRepository.deleteAll();  
  
 News news = News.*builder*()  
 .description("My descr")  
 .lessDescription("less descr")  
 .title("My title")  
 .build();  
  
 newsRepository.save(news);  
  
 List<News> res = newsRepository.findAll();  
 *assertEquals*("My descr", res.get(0).getDescription());  
 }  
  
}

**Результат**

****