**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**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  
дисциплина: Технологии Web-программирования  
тема: «Разработка и проектирование базы данных»

Выполнила: ст. группы ПВ-41  
Щетинина П.Г.  
Проверил: Картамышев С.В.

Белгород 2020

**Цель работы:** 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).

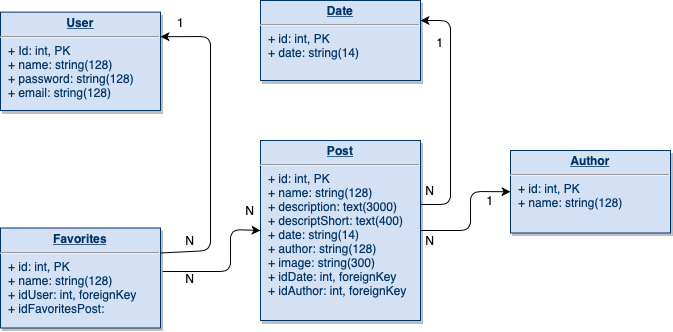
3. Разработать структуру базы данных.

4. Разработать соответствующие модели в приложении.

5. В отчёт приложить схему базы данных, а так же код одной из моделей (на своё усмотрение).

**Выполнение работы**

Для сайта новостей о мероприятиях была разработана следующая структура базы данных:



В файле конфигурации db.php и файле docker-compose были прописаны следующие настройки базы данных:

<?php  
  
return [  
 'class' => 'yii\db\Connection',  
 'dsn' => 'mysql:host=mysql;dbname=bd\_web',  
 'username' => 'user',  
 'password' => 'password',  
 'charset' => 'utf8',  
];

mysql:  
 image: mysql  
 command: --default-authentication-plugin=mysql\_native\_password --character-set-server=utf8 --collation-server=utf8\_general\_ci  
 environment:  
 - MYSQL\_ROOT\_PASSWORD=password  
 - MYSQL\_DATABASE=bd\_web  
 - MYSQL\_USER=user  
 - MYSQL\_PASSWORD=password

Для создания таблиц с помощью команды docker-compose run php yii migrate/create create\_product\_table были созданы миграции:

<?php  
  
use yii\db\Migration;  
  
*/\*\*  
\* Handles the creation of table `{{%product}}`.  
\*/*class m201221\_093250\_create\_post\_table extends Migration  
{  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeUp()  
 {  
 $this->createTable('{{%post}}', [  
 'id' => $this->primaryKey()->notNull(),  
 'name' => $this->string(250)->notNull()->comment('Название'),  
 'img' => $this->string()->notNull()->comment('Ссылка на изображение'),  
 'description' => $this->text()->notNull()->comment('Описание'),  
 'descriptionShort' => $this->text()->notNull()->comment('Краткое описание'),  
 'idDate' => $this->integer()->notNull()->comment('Дата поста'),  
 'idAuthor' => $this->integer()->notNull()->comment('Имя автора'),  
 ]);  
  
 $this->addForeignKey('fk-post-idDate', '{{%post}}', 'idDate', '{{%date}}', 'id', 'NO ACTION', 'NO ACTION');  
 $this->addForeignKey('fk-post-idAuthor', '{{%post}}', 'idAuthor', '{{%author}}', 'id', 'NO ACTION', 'NO ACTION');  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeDown()  
 {  
 $this->dropForeignKey('fk-post-idDate', '{{%post}}');  
 $this->dropForeignKey('fk-post-idAuthor', '{{%post}}');  
 $this->dropTable('{{%post}}');  
 }  
}

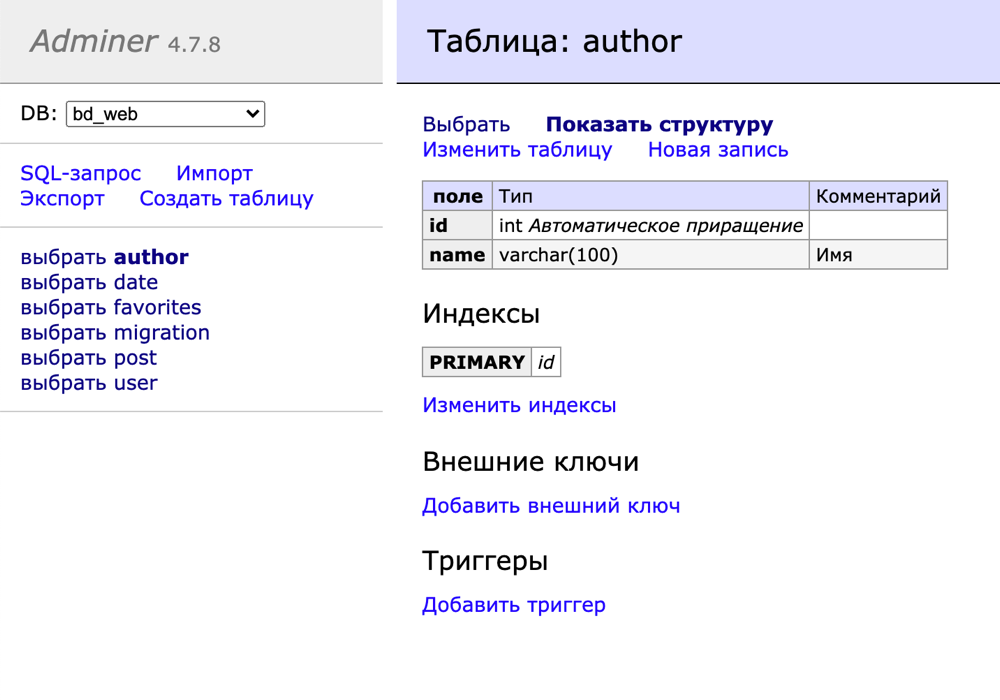
<?php  
  
use yii\db\Migration;  
  
*/\*\*  
 \* Handles the creation of table `{{%user}}`.  
 \*/*class m201221\_093520\_create\_user\_table extends Migration  
{  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeUp()  
 {  
 $this->createTable('{{%user}}', [  
 'id' => $this->primaryKey(),  
 'name' => $this->string(150)->notNull()->comment('Имя'),  
 'password' => $this->string(50)->notNull()->comment('Пароль'),  
 'email' => $this->string(100)->comment('Электронная почта'),  
 ]);  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeDown()  
 {  
 $this->dropTable('{{%user}}');  
 }  
}

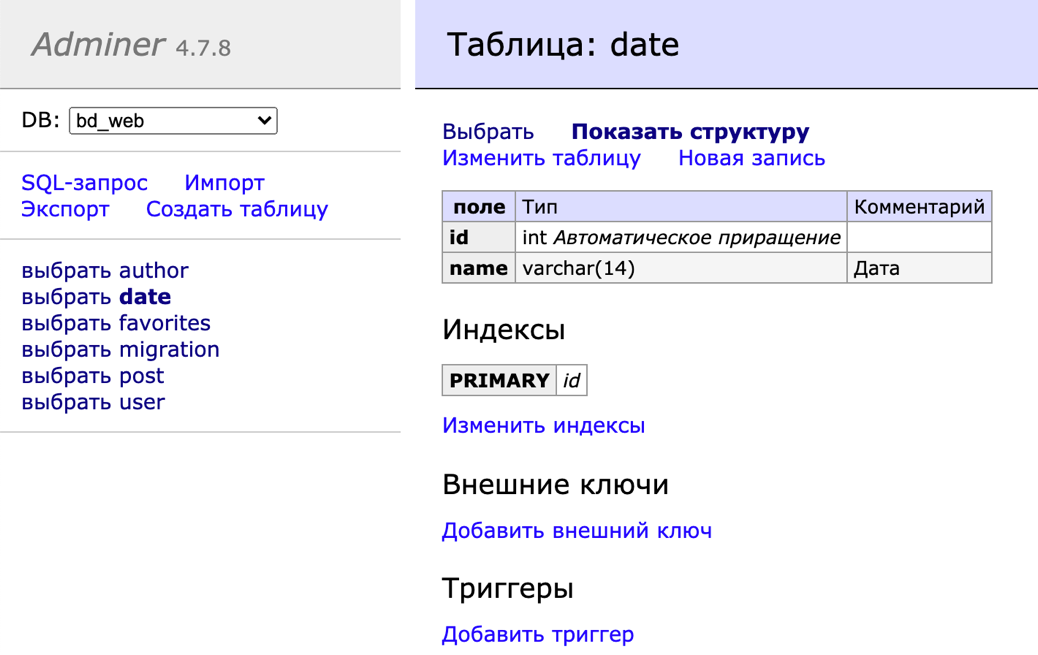
<?php  
  
use yii\db\Migration;  
  
*/\*\*  
\* Handles the creation of table `{{%category}}`.  
\*/*class m201221\_093608\_create\_author\_table extends Migration  
{  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeUp()  
 {  
 $this->createTable('{{%author}}', [  
 'id' => $this->primaryKey(),  
 'name' => $this->string(100)->notNull()->comment('Имя'),  
  
 ]);  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeDown()  
 {  
 $this->dropTable('{{%author}}');  
 }  
}

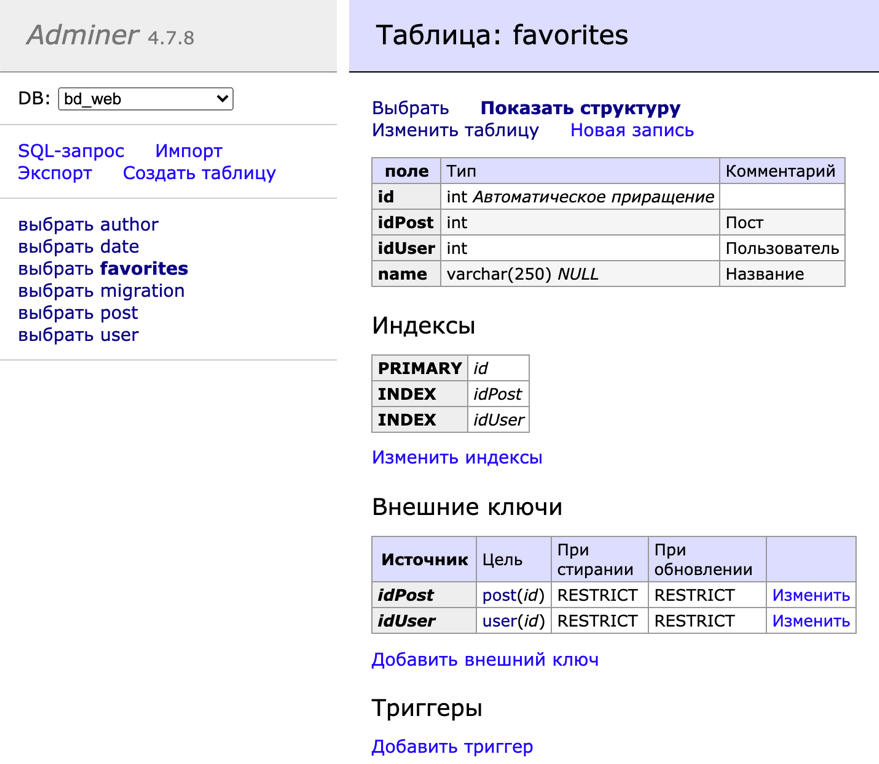
<?php  
  
use yii\db\Migration;  
  
*/\*\*  
\* Handles the creation of table `{{%category}}`.  
\*/*class m201221\_093706\_create\_date\_table extends Migration  
{  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeUp()  
 {  
 $this->createTable('{{%date}}', [  
 'id' => $this->primaryKey(),  
 'name' => $this->string(14)->notNull()->comment('Дата'),  
  
 ]);  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeDown()  
 {  
 $this->dropTable('{{%date}}');  
 }  
}

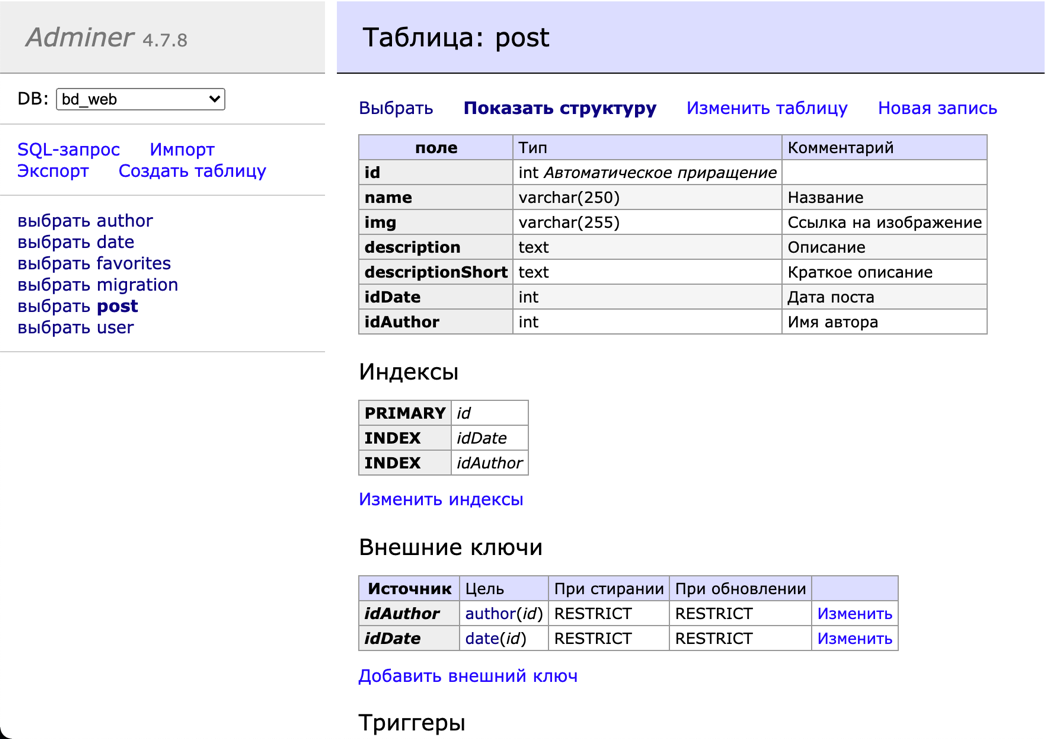
<?php  
  
use yii\db\Migration;  
  
*/\*\*  
\* Handles the creation of table `{{%order}}`.  
\*/*class m201221\_212146\_create\_favorites\_table extends Migration  
{  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeUp()  
 {  
 $this->createTable('{{%favorites}}', [  
 'id' => $this->primaryKey(),  
 'idPost' => $this->integer()->notNull()->comment('Пост'),  
 'idUser' => $this->integer()->notNull()->comment('Пользователь'),  
 'name' => $this->string(250)->comment('Название'),  
 ]);  
  
 $this->addForeignKey('fk-favorites-idPost', '{{%favorites}}', 'idPost', '{{%post}}', 'id', 'NO ACTION', 'NO ACTION');  
 $this->addForeignKey('fk-favorites-idUser', '{{%favorites}}', 'idUser', '{{%user}}', 'id', 'NO ACTION', 'NO ACTION');  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function safeDown()  
 {  
 $this->dropForeignKey('fk-favorites-idPost', '{{%favorites}}');  
 $this->dropForeignKey('fk-favorites-idUser', '{{%favorites}}');  
 $this->dropTable('{{%favorites}}');  
 }  
}

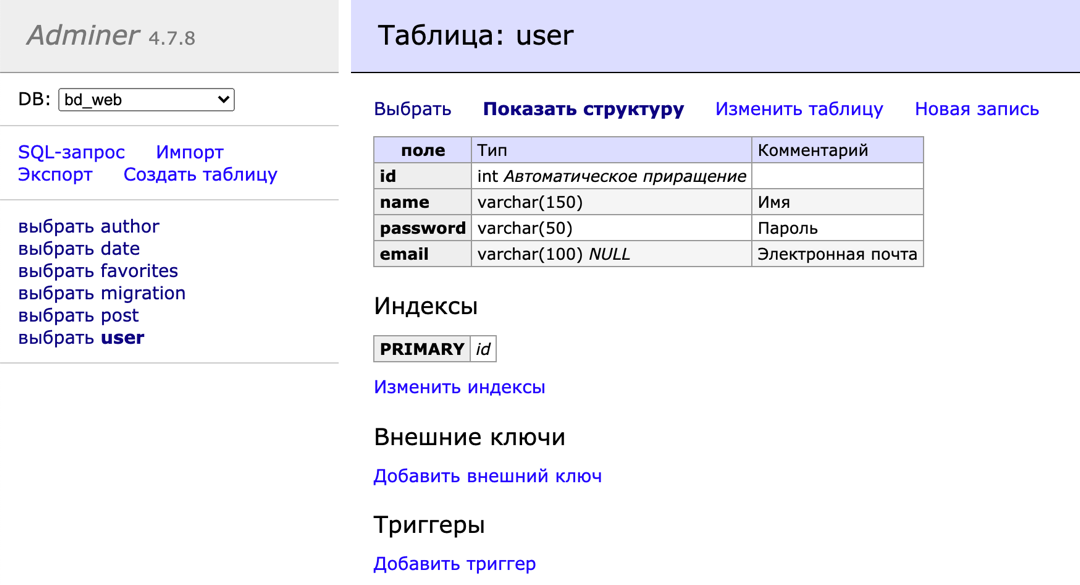
После применении команды docker-compose run php yii migrate/up проверили в adminer созданные таблицы:











Далее с помощью генератора моделей yii code generator были сгенерированы соответствующие модели. Для примера приведем код модели Post.php:

<?php  
  
namespace app\models;  
  
use Yii;  
  
*/\*\*  
 \* This is the model class for table "post".  
 \*  
 \* @property int $id  
 \* @property string $name Название  
 \* @property string $img Ссылка на изображение  
 \* @property string $description Описание  
 \* @property string $descriptionShort Краткое описание  
 \* @property int $idDate Дата поста  
 \* @property int $idAuthor Имя автора  
 \*  
 \* @property Favorites[] $favorites  
 \* @property Author $idAuthor0  
 \* @property Date $idDate0  
 \*/*class Post extends \yii\db\ActiveRecord  
{  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public static function tableName()  
 {  
 return 'post';  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function rules()  
 {  
 return [  
 [['name', 'img', 'description', 'descriptionShort', 'idDate', 'idAuthor'], 'required'],  
 [['description', 'descriptionShort'], 'string'],  
 [['idDate', 'idAuthor'], 'integer'],  
 [['name'], 'string', 'max' => 250],  
 [['img'], 'string', 'max' => 255],  
 [['idAuthor'], 'exist', 'skipOnError' => true, 'targetClass' => Author::className(), 'targetAttribute' => ['idAuthor' => 'id']],  
 [['idDate'], 'exist', 'skipOnError' => true, 'targetClass' => Date::className(), 'targetAttribute' => ['idDate' => 'id']],  
 ];  
 }  
  
 */\*\*  
 \* {@inheritdoc}  
 \*/* public function attributeLabels()  
 {  
 return [  
 'id' => 'ID',  
 'name' => 'Название',  
 'img' => 'Ссылка на изображение',  
 'description' => 'Описание',  
 'descriptionShort' => 'Краткое описание',  
 'idDate' => 'Дата поста',  
 'idAuthor' => 'Имя автора',  
 ];  
 }  
  
 */\*\*  
 \* Gets query for [[Favorites]].  
 \*  
 \* @return \yii\db\ActiveQuery  
 \*/* public function getFavorites()  
 {  
 return $this->hasMany(Favorites::className(), ['idPost' => 'id']);  
 }  
  
 */\*\*  
 \* Gets query for [[IdAuthor0]].  
 \*  
 \* @return \yii\db\ActiveQuery  
 \*/* public function getIdAuthor0()  
 {  
 return $this->hasOne(Author::className(), ['id' => 'idAuthor']);  
 }  
  
 */\*\*  
 \* Gets query for [[IdDate0]].  
 \*  
 \* @return \yii\db\ActiveQuery  
 \*/* public function getIdDate0()  
 {  
 return $this->hasOne(Date::className(), ['id' => 'idDate']);  
 }  
}

**Вывод:** в результате выполнения лабораторной работы изучили основы взаимодействия web-приложения с базой данных, спроектировали базу данных для хранения информации приложения (продукты, категории, начинки для продуктов, заказы, пользователи) и сгенерировали соответствующие модели.