**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИИ**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 № 3  
**«Разработка и проектирование базы данных.»**

Выполнил: ст. группы ПВ-41

Иванов Денис Александрович

Белгород 2019

**Цель:** Изучить основы работы 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.

**Выполнение**

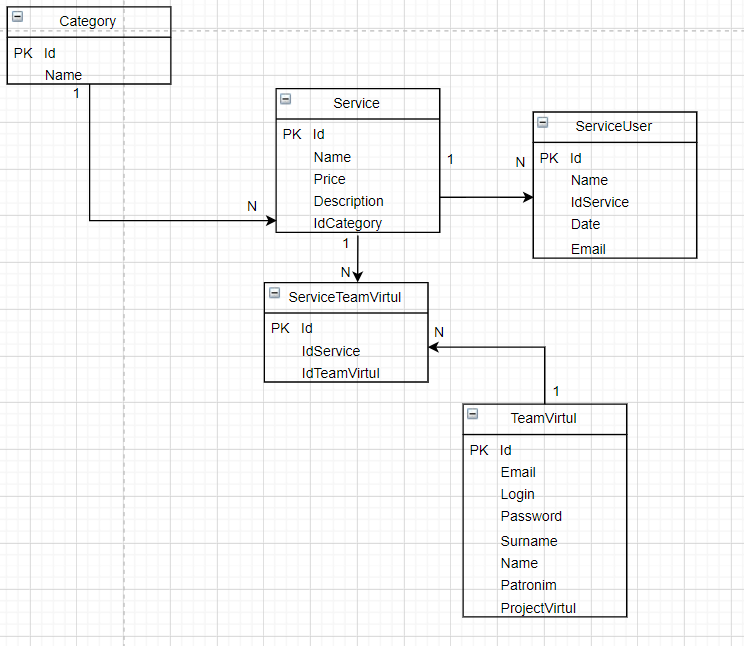
Active Record это M в [MVC](https://ru.wikipedia.org/wiki/Model-View-Controller) - модель - которая является слоем в системе, ответственным за представление бизнес-логики и данных. Active Record упрощает создание и использование бизнес-объектов, данные которых требуют персистентного хранения в базе данных. Сама по себе эта реализация паттерна Active Record является описанием системы ORM (Object Relational Mapping).

Active Record предоставляет нам несколько механизмов, наиболее важными из которых являются способности для:

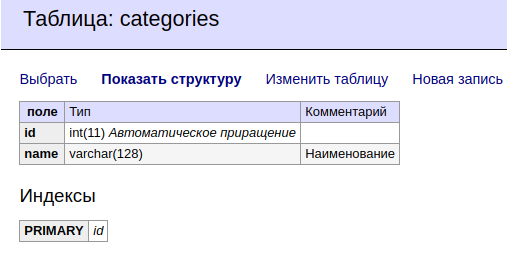
* Представления моделей и их данных.
* Представления связей между этими моделями.
* Представления иерархий наследования с помощью связанных моделей.
* Валидации моделей до того, как они станут персистентными в базе данных.
* Выполнения операций с базой данных в объектно-ориентированном стиле.

[Object Relational Mapping](https://ru.wikipedia.org/wiki/ORM) (объектно-реляционное отображение), обычно упоминающееся как аббревиатура ORM, это техника, соединяющая сложные объекты приложения с таблицами в системе управления реляционными базами данных. С помощью ORM, свойства и взаимоотношения этих объектов приложения могут быть с легкостью сохранены и получены из базы данных без непосредственного написания выражений SQL, и, в итоге, с меньшим суммарным кодом для доступа в базу данных.

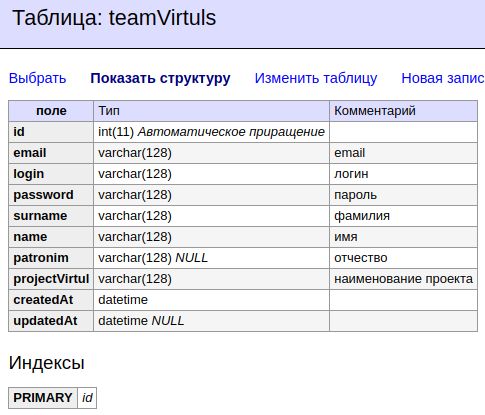
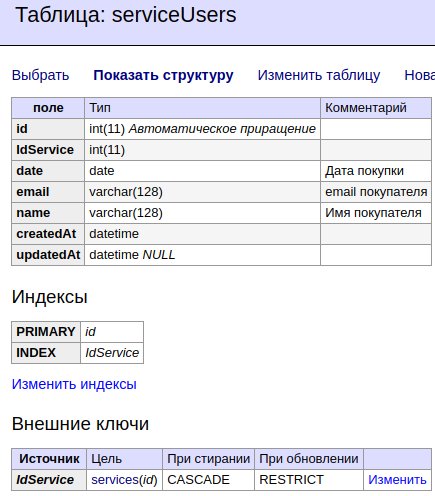
UML диаграмма:



Создание БД:







Разработка соответствующих моделей в приложении:

**Модель Category**

<?php

namespace app\models;

use Yii;

/

\* This is the model class for table "categories".

\*

\* @property int $id

\* @property string $name Наименование

\*

\* @property Service[] $services

\*/

class Category extends BaseModel

{

/

\* {@inheritdoc}

\*/

public static function tableName()

{

return 'categories';

}

/

\* {@inheritdoc}

\*/

public function rules()

{

return [

[['name'], 'required'],

[['name'], 'string', 'max' => 128],

];

}

/

\* {@inheritdoc}

\*/

public function attributeLabels()

{

return [

'id' => 'ID',

'name' => 'Наименование',

];

}

/\*\*

\* @return \yii\db\ActiveQuery

\*/

public function getServices()

{

return $this->hasMany(Service::className(), ['idCategories' => 'id']);

}

}

**Модель Service**

<?php

namespace app\models;

use Yii;

/

\* This is the model class for table "services".

\*

\* @property int $id

\* @property string $name Наименование

\* @property int $price Стоимость

\* @property string $description Описание

\* @property string $createdAt

\* @property string|null $updatedAt

\*

\* @property ServiceTeamVirtul[] $serviceTeamVirtuls

\* @property ServiceUser[] $serviceUsers

\*/

class Service extends BaseModel

{

/

\* {@inheritdoc}

\*/

public static function tableName()

{

return 'services';

}

/

\* {@inheritdoc}

\*/

public function rules()

{

return [

[['name', 'price', 'description', 'createdAt'], 'required'],

[['price'], 'integer'],

[['description'], 'string'],

[['createdAt', 'updatedAt'], 'safe'],

[['name'], 'string', 'max' => 128],

];

}

/

\* {@inheritdoc}

\*/

public function attributeLabels()

{

return [

'id' => 'ID',

'name' => 'Наименование',

'price' => 'Стоимость',

'description' => 'Описание',

'createdAt' => 'Created At',

'updatedAt' => 'Updated At',

];

}

/

\* @return \yii\db\ActiveQuery

\*/

public function getServiceTeamVirtuls()

{

return $this->hasMany(ServiceTeamVirtul::className(), ['idService' => 'id']);

}

/

\* @return \yii\db\ActiveQuery

\*/

public function getServiceUsers()

{

return $this->hasMany(ServiceUser::className(), ['IdService' => 'id']);

}

}

**Модель ServiceUser**

<?php

namespace app\models;

use Yii;

/

\* This is the model class for table "serviceUsers".

\*

\* @property int $id

\* @property int $IdService

//\* @property int $idUserSute

\* @property string $date Дата покупки

\* @property string $email email покупателя

\* @property string $name Имя покупателя

\* @property string $createdAt

\* @property string|null $updatedAt

\*

\* @property Service $idService

//\* @property UserSute $idUserSute0

\*/

class ServiceUser extends BaseModel

{

/

\* {@inheritdoc}

\*/

public static function tableName()

{

return 'serviceUsers';

}

/

\* {@inheritdoc}

\*/

public function rules()

{

return [

[['IdService', 'date', 'email', 'name', 'createdAt'], 'required'],

[['IdService'], 'integer'],

[['date', 'createdAt', 'updatedAt'], 'safe'],

[['email', 'name'], 'string', 'max' => 128],

[['IdService'], 'exist', 'skipOnError' => true, 'targetClass' => Service::className(), 'targetAttribute' => ['IdService' => 'id']],

//[['idUserSute'], 'exist', 'skipOnError' => true, 'targetClass' => UserSute::className(), 'targetAttribute' => ['idUserSute' => 'id']],

];

}

/

\* {@inheritdoc}

\*/

public function attributeLabels()

{

return [

'id' => 'ID',

'IdService' => 'Id Service',

//'idUserSute' => 'Id User Sute',

'date' => 'Дата покупки',

'email' => 'email покупателя',

'name' => 'Имя покупателя',

'createdAt' => 'Created At',

'updatedAt' => 'Updated At',

];

}

/

\* @return \yii\db\ActiveQuery

\*/

public function getIdService()

{

return $this->hasOne(Service::className(), ['id' => 'IdService']);

}

///

// \* @return \yii\db\ActiveQuery

// \*/

//public function getIdUserSute0()

//{

// return $this->hasOne(UserSute::className(), ['id' => 'idUserSute']);

//}

}