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

Выполнил: Шошин Олег ПВ-42

Проверил: Картамышев С. В.

Белгород 2020г.

Лабораторная работа №4

Технологии web-программирования.

**«Разработка и проектирование базы данных»**

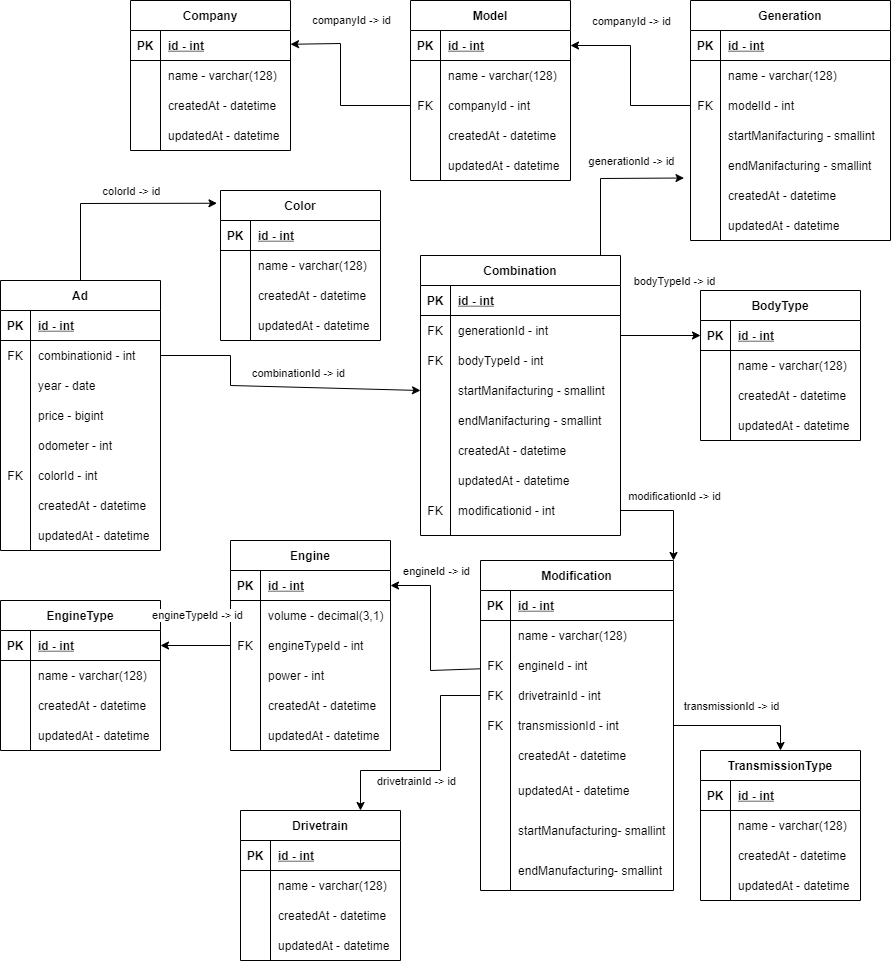
**Цель**: изучить основы взаимодействия web-приложения с базой данных. Спроектировать базу данных для хранения информации приложения (страницы, пользователи и т.п.).

**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**

Схема базы данных:



Модель Modification

<?php

namespace app\modules\v1\models;

use Yii;

/\*\*

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

 \*

 \* @property int $id

 \* @property string $name Название

 \* @property int|null $drivetrainId Привод

 \* @property int|null $engineId Двигатель

 \* @property int|null $transmissionId Коробка передач

 \* @property string $startManufacturing Начало производства

 \* @property string|null $endManufacturing Конец производства

 \* @property string|null $createdAt Дата создания

 \* @property string|null $updatedAt Дата изменения

 \*

 \* @property Combination[] $combinations

 \* @property Engine $engine

 \* @property TransmissionType $transmission

 \* @property Drivetrain $drivetrain

 \*/

class Modification extends \app\modules\v1\models\BaseModel

{

    /\*\*

     \* {@inheritdoc}

     \*/

    public static function tableName()

    {

        return 'Modification';

    }

    /\*\*

     \* {@inheritdoc}

     \*/

    public function rules()

    {

        return [

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

            [['drivetrainId', 'engineId', 'transmissionId'], 'integer'],

            [['startManufacturing', 'endManufacturing', 'createdAt', 'updatedAt'], 'safe'],

            [['name'], 'string', 'max' => 100],

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

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

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

        ];

    }

    /\*\*

     \* {@inheritdoc}

     \*/

    public function attributeLabels()

    {

        return [

            'id' => 'ID',

            'name' => 'Название',

            'drivetrainId' => 'Привод',

            'engineId' => 'Двигатель',

            'transmissionId' => 'Коробка передач',

            'startManufacturing' => 'Начало производства',

            'endManufacturing' => 'Конец производства',

            'createdAt' => 'Дата создания',

            'updatedAt' => 'Дата изменения',

        ];

    }

    /\*\*

     \* Gets query for [[Combinations]].

     \*

     \* @return \yii\db\ActiveQuery

     \*/

    public function getCombinations()

    {

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

    }

    /\*\*

     \* Gets query for [[Engine]].

     \*

     \* @return \yii\db\ActiveQuery

     \*/

    public function getEngine()

    {

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

    }

    /\*\*

     \* Gets query for [[Transmission]].

     \*

     \* @return \yii\db\ActiveQuery

     \*/

    public function getTransmission()

    {

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

    }

    /\*\*

     \* Gets query for [[Drivetrain]].

     \*

     \* @return \yii\db\ActiveQuery

     \*/

    public function getDrivetrain()

    {

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

    }

    public function toArray(array $fields = [], array $expand = [], $recursive = true) {

        return [

            'id' => $this->id,

            'name' => $this->name,

            'drivetrain' => $this->drivetrain,

            'engine' => $this->engine,

            'transmission' => $this->transmission,

            'startManufacturing' => $this->startManufacturing,

            'endManufacturing' => $this->endManufacturing,

        ];

    }

    public static function getModifications($generationId, $bodyTypeId, $year, $engineTypeId = null, $drivetrainId = null ) {

        $modifications = Modification::find()

            ->where(['<=', 'startManufacturing', $year])

            ->andWhere(['>=', 'endManufacturing', $year])

            ->joinWith('combinations')

            ->andWhere(['generationId' => $generationId])

            ->andWhere(['bodyTypeId' => $bodyTypeId]);

        if(isset($engineTypeId)) {

            $modifications

                ->joinWith('engine')

                ->andWhere(['engineTypeId' => $engineTypeId]);

        }

        if(isset($drivetrainId)) {

            $modifications

                ->andWhere(['drivetrainId' => $drivetrainId]);

        }

        return $modifications;

    }

}