**Ликино-Дулевский политехнический колледж – филиал ГГТУ**

**ОТЧЁТ**

Предметная область: «Макет сайта компании для предоставления стоматологических услуг ООО «BUYcycle»»

По дисциплине: «УП.08 Учебная практика»

Специальность 09.02.07 Информационные системы и программирование курс 3

Выполнил:

Степанов Дмитрий Даниилович

Проверил преподаватель:

Гжегожевский С.В.

**2023 год**

**Оглавление**

[1.Разработка web-приложения 3](#_Toc154406423)

[1.1. Общие сведения. 3](#_Toc154406424)

[1.1.1. Что такое Django 3](#_Toc154406425)

[Django — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. Проект поддерживается организацией Django Software Foundation. Сайт на Django строится из одного или нескольких приложений. 3](#_Toc154406426)

[1.1.2. Установка и настройка. 3](#_Toc154406427)

[1.1.3. Основные команды Django 3](#_Toc154406428)

[1.2. Руководство программиста 3](#_Toc154406429)

[1.2.1. Модели 3](#_Toc154406430)

[1.2.2. Представления 3](#_Toc154406431)

[1.2.3. Шаблоны 4](#_Toc154406432)

[1.2.4. Работа с формами 5](#_Toc154406433)

[1.3. Руководство пользователя 6](#_Toc154406434)

[2.Роутинг в CMS Bitrix 8](#_Toc154406435)

[2.1. Установить CMS Bitrix 8](#_Toc154406436)

[2.2. Настройка роутинга 11](#_Toc154406437)

[2.3. Практическая часть 12](#_Toc154406438)

[2.3.1. Получить все активные товары определенного раздела; 12](#_Toc154406439)

[2.3.2. Получить все товары по фильтру; 12](#_Toc154406440)

[2.3.3. Получить все активные товары с ценой; 13](#_Toc154406441)

[3. Быстрый старт разработчика CMS Bitrix 13](#_Toc154406442)

[3.1. Установка системы 13](#_Toc154406443)

[3.2 Интеграция верстки 19](#_Toc154406444)

[3.3 Редактируемые области 21](#_Toc154406445)

[3.4 Базовая настройка системы 21](#_Toc154406446)

[3.5 Компоненты и структура сайта 22](#_Toc154406447)

# 1.Разработка web-приложения

## 1.1. Общие сведения.

### 1.1.1. Что такое Django

Django — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. Проект поддерживается организацией Django Software Foundation. Сайт на Django строится из одного или нескольких приложений.

### 1.1.2. Установка и настройка.

Установка Django

Установка через Настройки: File-> Setting-> Python Interpreter-> “+” Django

Установка через pip: pip install django

### 1.1.3. Основные команды Django

django-admin startproject myproject - создать новый проект

python manage.py runserver - запустить тестовый сервер

django-admin startapp myapp - создать приложение в текущем проекте

python manage.py createsuperuser - создать пользователя-администратора

python manage.py [migrate](https://djangodoc.ru/3.1/ref/django-admin/#django-admin-migrate) - отвечает за выполнение и отмену миграции.

python manage.py makemigrations - отвечает за создание новых миграций на основе изменений, внесенных в модели.

## 1.2. Руководство программиста

### 1.2.1. Модели

Создана 1 модель «Position» в ней находится имя, производитель и цена.

from django.db import models  
  
  
class Position(models.Model):  
 name = models.CharField('Название', max\_length=30)  
 type = models.CharField('Производитель', max\_length=40)  
 price = models.IntegerField('Цена')  
  
 def \_\_str\_\_(self):  
 return self.name  
 class Meta:  
 verbose\_name="Товар"  
 verbose\_name\_plural="Товар магазина"

### 1.2.2. Представления

from django.shortcuts import render, redirect  
from .models import Position  
from .forms import PosForm  
from django.views.generic import DeleteView, UpdateView  
from .filters import PostFilter

**Главная страница**

def index(request):  
 positions\_all = Position.objects.all()  
 positions = PostFilter(request.GET, queryset=positions\_all)  
 return render(request, 'main/index.html', {'title': 'Главная страница сайта', 'positions': positions})

Страница О Нас

def about(request):  
 return render(request, 'main/about.html')

Страница Обслуживания

def service(request):  
 return render(request, 'main/service.html')

Страница Добавления

def add(request):  
 error = ''  
 if request.method == 'POST':  
 form = PosForm(request.POST)  
 if form.is\_valid():  
 form.save()  
 return redirect('home')  
 else:  
 error = 'Форма заполнена неправильно или не до конца'  
 form = PosForm()  
 context = {  
 'form': form  
 }  
 return render(request, 'main/add.html', context)

**Кнопка Удаления**

class Delete(DeleteView):  
 model = Position  
 success\_url = '/'  
 template\_name = 'main/delete.html'

**Кнопка Редактирования**

class Update(UpdateView):  
 model = Position  
 template\_name = 'main/add.html'  
 form\_class = PosForm  
 success\_url = '/'

### 1.2.3. Шаблоны

Шаблон «add.html» там находится форма добавления товара с полями:

Название, Производитель и Цена.

{% extends 'main/base.html' %}  
  
{% block title %}Добавить товар{% endblock %}  
{% block tab-name %}Добавить товар{% endblock %}  
{% block content %}  
 <h1>Добавить товар</h1>  
 <form method="post" class="add-form">  
 {% csrf\_token%}  
 {{form.name}} <br>  
 {{form.type}} <br>  
 {{form.price}} <br>  
 <button type="submit">Добавить</button>  
 <span>{{error}}</span>  
 </form>  
{% endblock %}

Шаблон «Index.html» там находится блок фильтрации услуг и возможность их удалить или редактировать.

{% extends 'main/base.html' %}  
  
{% block title %}  
{{ title }}  
{% endblock %}  
{% block tab-name %}Главная{% endblock %}  
{% block content %}  
<h1>Каталог</h1>  
{% if positions %}  
 <form method="get" class="form-filter">  
 <div>  
 Название: {{ positions.form.name }}  
 Производитель: {{ positions.form.type }}  
 Цена: {{ positions.form.price }}  
 <button type="submit" class="btn btn-primary">Фильтровать</button>  
 </div>  
 </form>  
<table>  
 <tr>  
 <th>Название</th>  
 <th>Производитель</th>  
 <th>Цена</th>  
 <th>Управление записями</th>  
 </tr>  
 {% for el in positions.qs %}  
 <tr>  
 <td>{{el.name}}</td>  
 <td>{{el.type}}</td>  
 <td>{{el.price}} р.</td>  
 <td>  
 <div class="produtcs\_\_btns">  
 <a href="{% url 'delete' el.id %}">  
 <button class="btn btn-danger">Удалить</button>  
 </a>  
 <a href="{% url 'update' el.id %}">  
 <button class="btn btn-success">Редактировать</button>  
 </a>  
 </div>  
 </td>  
 </tr>  
 {% endfor %}  
</table>  
{% else %}  
<p>У нас нет позиций в каталоге</p>  
{%endif%}  
{% endblock %}

### 1.2.4. Работа с формами

Создана форма в которой находятся: Название, Производитель и цена.

from .models import Position  
from django.forms import ModelForm, TextInput, Textarea  
  
  
class PosForm(ModelForm):  
 class Meta:  
 model = Position  
 fields = ["name", "type", "price"]  
 widgets = {  
 "name": TextInput(attrs={'placeholder': 'Название', 'class': 'form-input'}),  
 "type": TextInput(attrs={'placeholder': 'Производитль', 'class': 'form-input'}),  
 "price": TextInput(attrs={'placeholder': 'Цена в руб.', 'class': 'form-input'}),  
 }

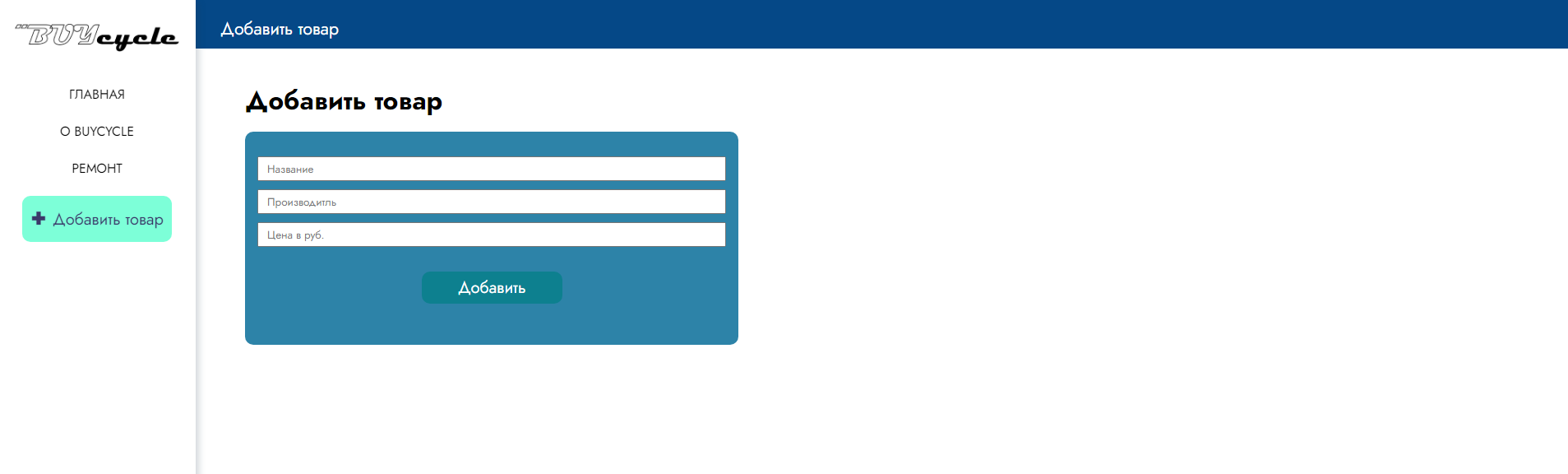
## 1.3. Руководство пользователя

На странице «Главная» есть фильтрация и таблица с велосипедами, велосипеды можно редактировать и удалять.

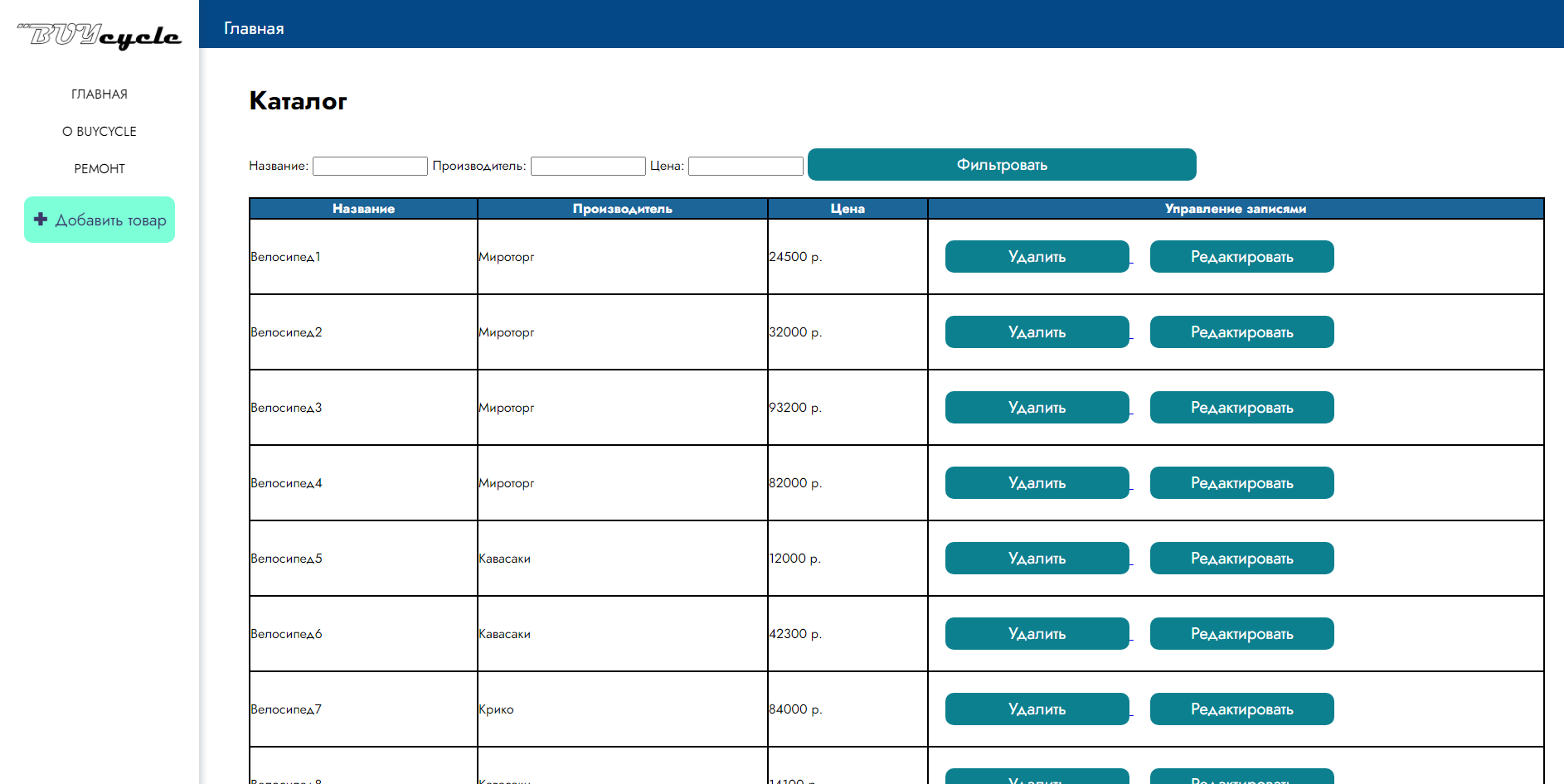
На странице «О нас» располагается информация о фирме.

На странице «Обслуживание» находятся информация о том, как произвести ремонт в магазине компании.

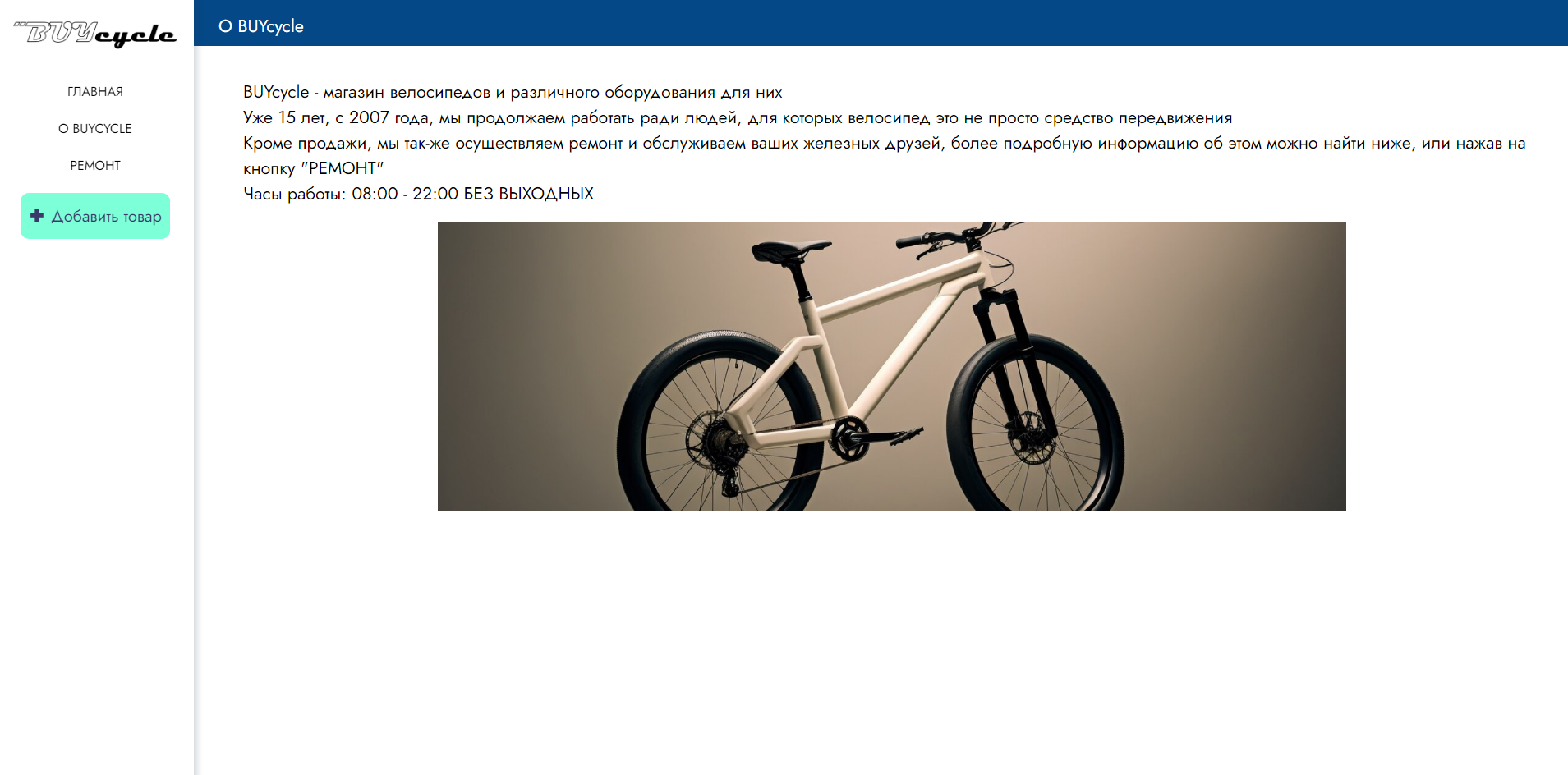
На странице «Добавить услугу» находится форма с полями, где можно добавить услугу.



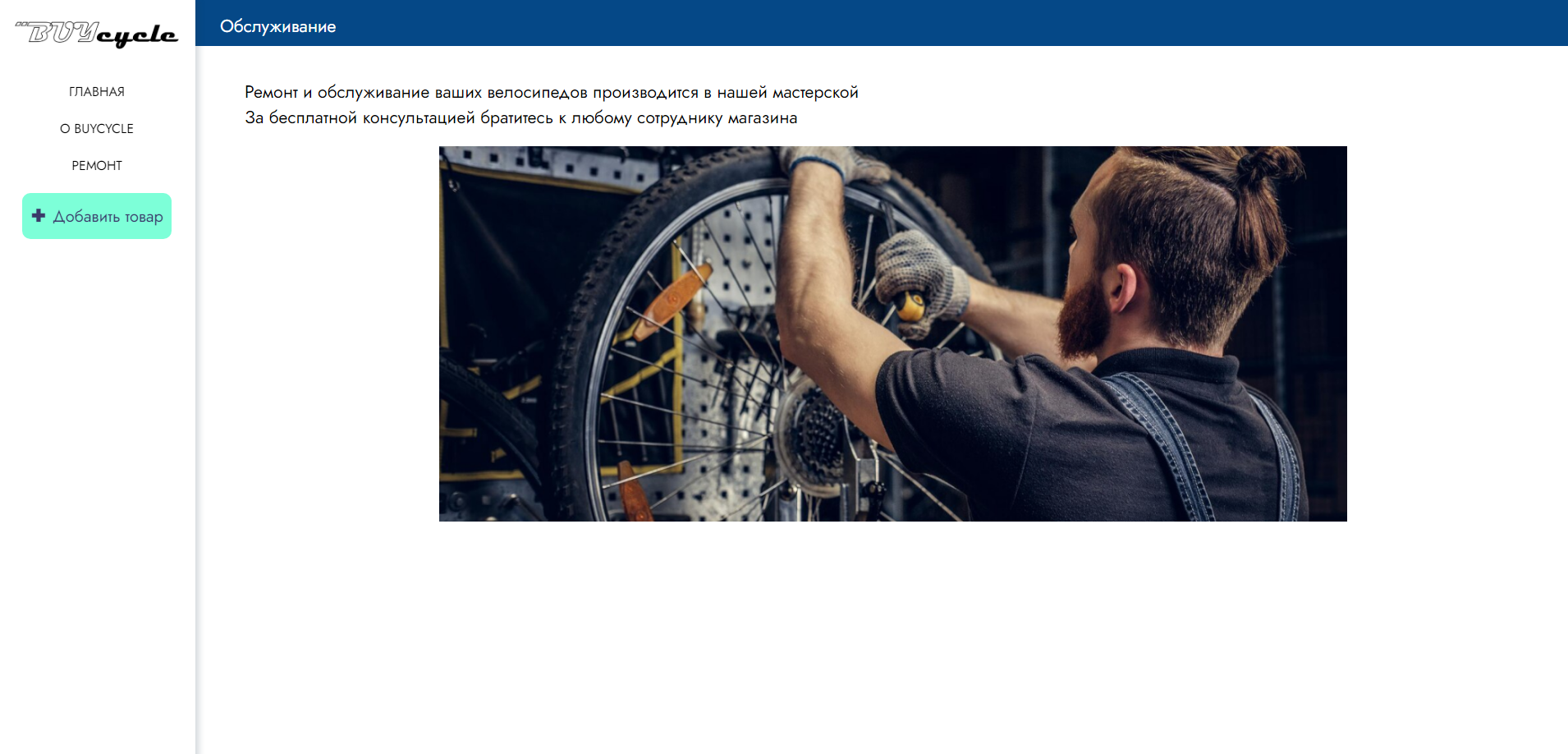
*Рис. 1. – Добавление услуги.*



*Рис. 2. – Главная страница.*



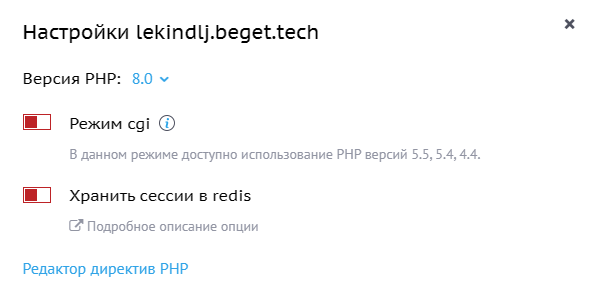
*Рис. 3. – Страница О нас.*



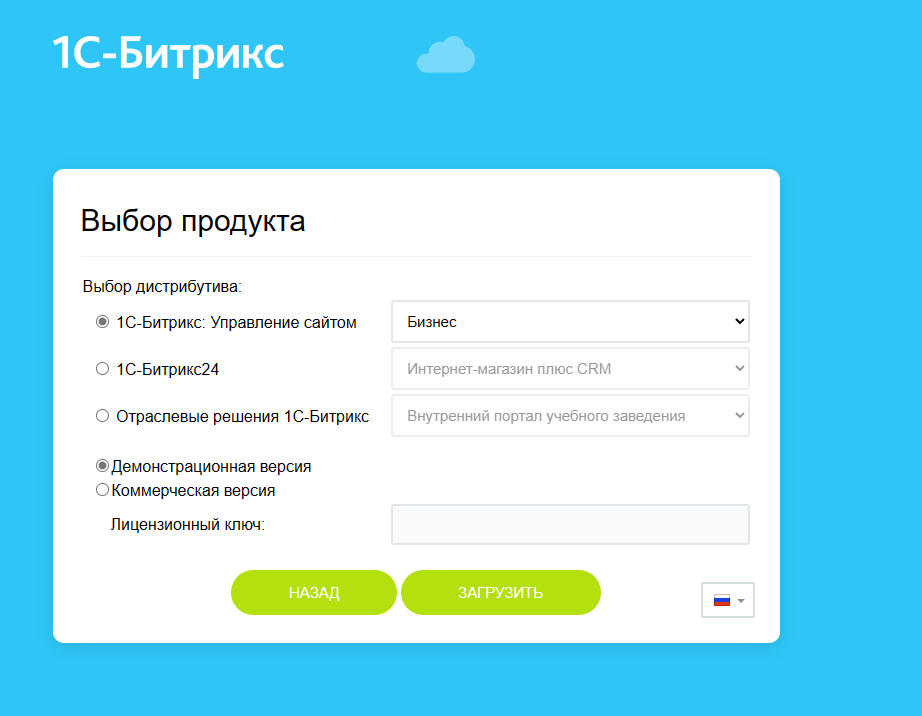
*Рис. 4. – Страница обслуживания.*

# 2.Роутинг в CMS Bitrix

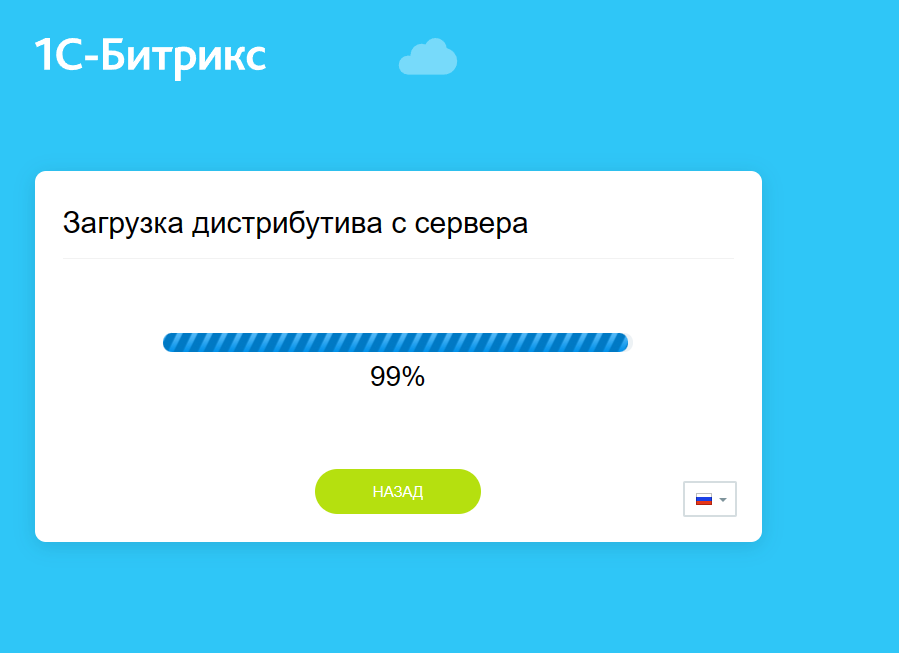
## 2.1. Установить CMS Bitrix



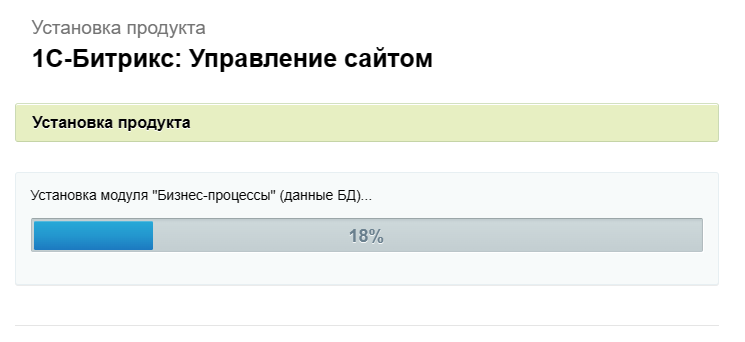
*Рис. 5. – Настройка хоста.*



*Рис. 6. – Выбор направления сайта.*

****

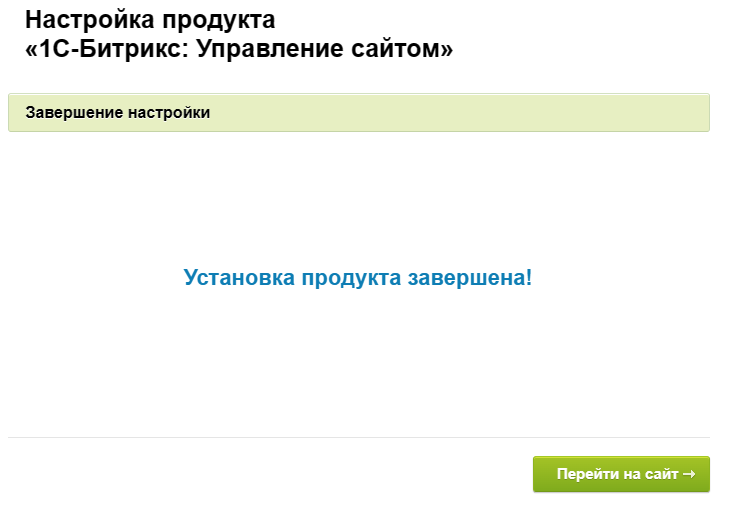
*Рис. 7. – Загрузка дистрибутива сайта.*

****

*Рис. 8. – Загрузка продукта.*

****

*Рис. 9. – Настройка продукта.*

****

*Рис. 10. – Настройка и загрузка завершены.*

## 2.2. Настройка роутинга

Находится в папке routes/web.php

use Bitrix\Main\HttpRequest;

use Bitrix\Main\Routing\Route;

use Bitrix\Main\Routing\RoutingConfigurator;

## 2.3. Практическая часть

### 2.3.1. Получить все активные товары определенного раздела;

Находится в папке routes/web.php

$elements = \Bitrix\Iblock\Elements\ElementCatalogTable::getList([

'select' => ['ID', 'NAME', 'SORT'],

'filter' => [

'=ACTIVE' => 'Y',

'=IBLOCK\_SECTION\_ID' => $\_REQUEST['section\_id'],

],

])->fetch();

if(!$elements){

return json\_encode([

'status' => 'error',

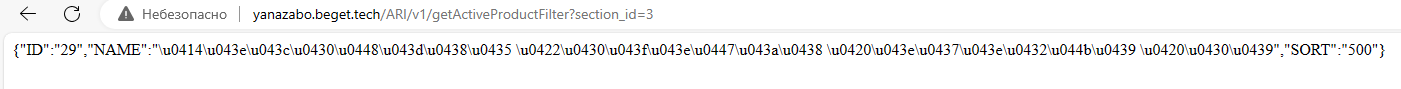
'error' => 'В данном разделе нет товаров!'

]);

}

return json\_encode($elements);

});



*Рис. 11. – Вывод всех активных товаров определенного раздела.*

### 2.3.2. Получить все товары по фильтру;

Находится в папке routes/web.php

$routes->any('/ARI/v1/getActiveProductSection', function () {

\Bitrix\Main\Loader::includeModule('iblock');

if(!$\_REQUEST['section\_id']){

return json\_encode([

'status' => 'error',

'error' => 'Нет ключа section\_id'

]);

}

$elements = \Bitrix\Iblock\Elements\ElementCatalogTable::getList([

'select' => ['ID', 'NAME'],

'filter' => [

'=ACTIVE' => 'Y',

'=IBLOCK\_SECTION\_ID' => $\_REQUEST['section\_id'],

],

])->fetchAll();

if(!$elements){

return json\_encode([

'status' => 'error',

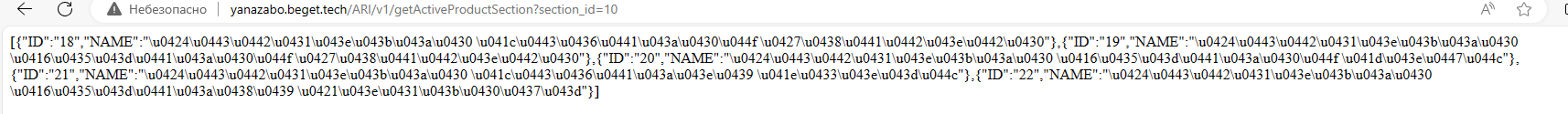
'error' => 'В данном разделе нет товаров!'

]);

}

return json\_encode($elements);

});



*Рис. 12. – Вывод всех товаров по фильтру.*

### 2.3.3. Получить все активные товары с ценой;

Находится в папке routes/web.php

$routes->any('/ARI/v1/getActiveProductPrice', function () {

\Bitrix\Main\Loader::includeModule('iblock');

\Bitrix\Main\Loader::includeModule('sale');

$element = \Bitrix\Iblock\Elements\ElementCatalogTable::getList([

'select' => ['ID', 'NAME'],

'filter' => [

'=ACTIVE' => 'Y',

'=ID' => 4

],

])->fetch();

$price = \Bitrix\Catalog\PriceTable::getList([

'select' => ['PRICE', 'CURRENCY', 'ID', 'PRODUCT\_ID'],

'filter' => ['=PRODUCT\_ID' => $element['ID']]

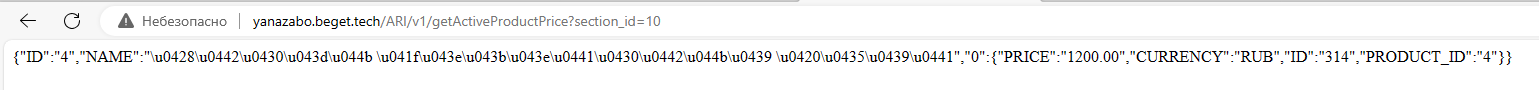
])->fetch();

array\_push($element, $price);

return json\_encode($element);

d($element);

});



*Рисунок 13 – Вывод всех активных товаров с ценой.*

# 3. Быстрый старт разработчика CMS Bitrix

## 3.1. Установка системы

Таблица №1 «Процесс установки Bitrix»

|  |  |
| --- | --- |
| Шаг 1  Установка BitrixSetup |  |
| Шаг 2  Загружаем файл BitrixSetup в файловый менеджер |  |
| Шаг 3  Выбираем направление сайта |  |
| Шаг 4  Ждем окончания загрузки |  |
| Шаг 5  Начало установки нажимаем «Далее» |  |
| Шаг 6  Принимаем лицензионное соглашение и нажимаем «Далее» |  |
| Шаг 7  Вводим свои данные, выбираем установить UTF-8 и нажимаем «Далее» |  |
| Шаг 8  Пролистываем в самый конец и нажимаем «Далее» |  |
| Шаг 9  Вводим логин и пароль от базы данных и нажимаем «Далее» |  |
| Шаг 10  Ждем окончания загрузки продукта и нажимаем «Далее» |  |
| Шаг 11  Создаем администратора и нажимаем «Далее» |  |
| Шаг 12  Выбираем решение для установки «Загрузить из Marketplace» и нажимаем «Далее» |  |
| Шаг 13  Выбираем решение «Чистая установка» и нажимаем «Далее» |  |
| Шаг 14  Ждем окончания загрузки решения |  |
| Шаг 15  Даем название шаблону и нажимаем «Настройка решения» |  |
| Шаг 16  Даем название сайту и нажимаем «Настройка решения» |  |
| Шаг 17  Выбираем все кроме «Перевод» и «Социальные сервисы» и нажимаем «Настройка решения» |  |
| Шаг 18  Ждем окончания установки |  |
| Шаг 19  Конец установки |  |

## 3.2 Интеграция верстки

Интегрируем изначальный код header и footer

Header

$blsMainPage = $APPLICATION->GetCurPage(false) == SITE\_DIR;

?>

<!DOCTYPE html>

<!--[if lt IE 8]> <html class="no-js lt-ie9 lt-ie8"><![endif]-->

<!--[if IE 8]> <html class="no-js lt-ie9"><![endif]-->

<!--[if gt IE 8]> <html class="no-js"><![endif]-->

<head>

<!--[if IE]>

<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"><![endif]-->

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title><?$APPLICATION->ShowTitle();?></title>

<?$APPLICATION->ShowHead();?>

<?$APPLICATION->SetAdditionalCSS(SITE\_TEMPLATE\_PATH.'/css/common-styles.css');?>

<link rel="icon" href="<?=SITE\_TEMPLATE\_PATH?>/ico/favicon\_bx.png">

<!--[if lt IE 9]>

<script src="<?=CUtil::GetAdditionalFileURL(SITE\_TEMPLATE\_PATH.'/js/vendor/modernizr-html5shiv-respond.min.js')?>"></script>

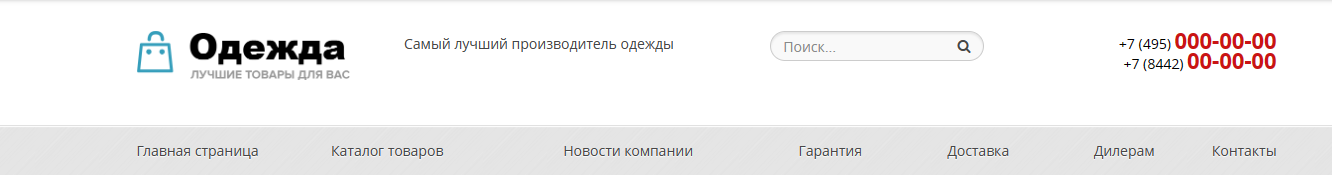
<![endif]-->

<!--[if gte IE 9]><!-->

<script src="<?=CUtil::GetAdditionalFileURL(SITE\_TEMPLATE\_PATH.'/js/vendor/modernizr.min.js')?>"></script>

<!--<![endif]-->

</head>



*Рис. 14. – Header.*

Footer

<footer>

<div class="sticky-footer">

<div class="container">

<div class="row">

<div class="col-md-4">

<address>

<?$APPLICATION->IncludeFile(

SITE\_DIR."include/address.php",

array(),

array(

"MODE" => "html"

)

);?>

</address>

</div>



*Рис. 15. – Footer.*

## 3.3 Редактируемые области

Даём возможность администратору сайта менять текст в шаблоне сайта, без знаний HTML верстки и программирования.

<div class="col-sm-6 col-xs-12">   
            <div class="row">   
                <div class="col-lg-7 col-xs-12 hidden-xs">   
                          <ul class="btn-list-inline">   
                               <?$APPLICATION->IncludeFile(   
                                            SITE\_DIR."include/slogan.php",   
                             array(),   
                             array(   
                              "MODE" => "text"   
                                            )   
                                        );?>   
   
                                    </ul>   
                                </div>

Создаём папку «include» в корневой папке и закидываем туда slogan.php и вставляем туда «Самый лучший производитель одежды»   
Также делаем с social-links.php, phone1, phone2, address.

## 3.4 Базовая настройка системы

*Таблица №2 «Базовая настройка системы»*

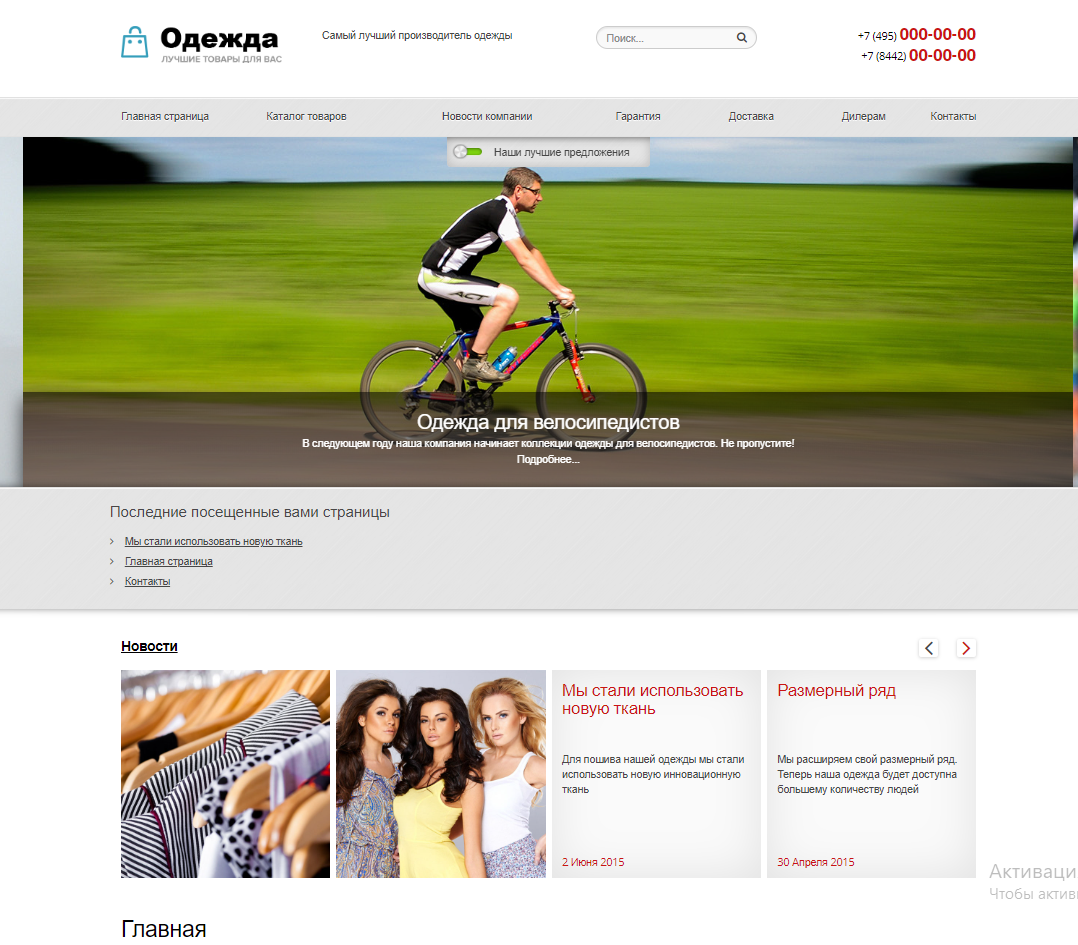
|  |  |
| --- | --- |
| Заходим в админку, следом в настройки и в Главный модуль, меняем название сайта под своё, переходим по ссылке «формы подключения AIP-ключа». |  |
| Нужно авторизоваться в Яндексе, чтобы продолжить. |  |
| Введите описание ключа |  |
| Вставьте полученный ключ и нажимаем «Применить» |  |
| Теперь заходим в Информационные блоки. Отмечаем первую и третью галочку и нажимаем «Сохранить» |  |

## 3.5 Компоненты и структура сайта

*Таблица №3 «Компоненты и структура сайта»*

|  |  |
| --- | --- |
| В админке заходим в контент и переходим в файлы и папки. |  |
| Создаем новую папку и вписываем данные, после чего нажимаем «Сохранить» |  |
| Меняем название и вставляем нужный текст, после чего нажимаем «Сохранить» |  |
| Создаем меню и нажимаем «Сохранить» |  |
| Меняем название и вставляем нужный текст, после чего нажимаем «Сохранить» |  |
| Заходим на страницу test.php, нажимаем на редактирование страницы. Открываем меню справа, выбираем навигацию и заходим в меню. После два раза подтверждаем сохранение. |  |
| Включаем режим правки |  |
| Заходим в настройки |  |
| Выбираем тип для первого уровня «Главное меню» и нажимаем сохранить. |  |
| Вставили пункты главного меню |  |
| Заходим в настройки, меню и копировать шаблон компонента. |  |
| Меняем данные и нажимаем «Сохранить» |  |
| Нажимаем на изменить страницу и заходим в «В режиме PHP-кода» |  |
| Копируем эту часть кода |  |
| Вставляем в наш код в header.php |  |
| Вот что мы получаем |  |
| Нажимаем на создать раздел |  |
| Задаем заголовок, имя папки, подтверждаем вторую стрелочку и нажимаем «Далее». |  |
| Выбираем тип меню «Главное меню», вставить перед пунктом «Гарантия» и нажимаем «Далее». |  |
| Нажимаем «Готово» |  |
| Добавляем так же остальные пункты меню. |  |
| Создаем еще один новый раздел. Называем «Поиск» и даем имя нужной папки. Убираем все галочки если они есть и нажимаем «Готово». |  |
| Вот как это должно выглядеть. |  |
| Возвращаемся на страницу test.php, но теперь выбираем не меню, а навигационная цепочка. |  |
| Открывшееся окно просто сохраняем |  |
| Вот что получается |  |
| Заходим в настройки, навигационная цепочка и выбираем «Копировать шаблон компонента» |  |
| В название шаблона вставляем имя, что написано сверху в ссылке, выбираем копировать в текущий, применить новый шаблон компонента и нажимаем «Сохранить». |  |
| Вот что у нас получается в итоге. |  |

**Итог главной страницы**



*Рис. 14. – Итоговая Главная страница.*

1. Разработка API для бэкэнда
   1. Разработка сервера
   2. Выполнение запросов к серверу
2. Проектирование предметной области
3. Разработка технической документации
   1. Техническое задание
   2. Руководство пользователя
   3. Руководство программиста
   4. Методика тестирования и испытания программы
   5. Тест-план программы
4. Обеспечения качества программного обеспечения
5. Расчет надежности работы сайта