Московский государственный технический университет (МФТИ)

Факультет биологической и медицинской физики

***Отчет по проекту***

по учебной дисциплине

**ИНФОРМАТИКА**

тема проекта:

**Разработка веб-сайта «Все самое интересное из мира науки о жизни»**

на основе библиотеки Django

Выполнила: Бруттан М.В.

Группа: 6114

##### Долгопрудный

##### 2018

**Содержание**

[1. Техническое задание 3](#_Toc513417205)

[*1.1 Название проекта. Назначение веб-сайта* 3](#_Toc513417206)

[*1.2 Требования к веб-сайту* 3](#_Toc513417207)

[2. Проектирование веб-сайта 4](#_Toc513417208)

[*2.1 Проектирование карты веб-сайта* 4](#_Toc513417209)

[*2.2 Проектирование базы данных веб-сайта* 6](#_Toc513417210)

[*2.4 Проектирование шаблона Мастер-страницы сайта* 8](#_Toc513417211)

[3. Разработка веб-сайта 9](#_Toc513417212)

[*3.1 Разработка классов моделей* 9](#_Toc513417213)

[*3.2 Разработка базы данных веб-сайта* 11](#_Toc513417214)

[*3.3 Разработка модулей веб-сайта* 12](#_Toc513417215)

[*3.4 Разработка шаблонов веб-страниц сайта* 17](#_Toc513417216)

[*3.5 Разработка форм* 28](#_Toc513417217)

[*3.6 Разработка дизайна страниц (стилевое оформление страниц)* 29](#_Toc513417218)

[4. Описание пользовательского интерфейса 36](#_Toc513417219)

[*4.1 Руководство пользователя* 36](#_Toc513417220)

[*4.2 Руководство для администратора* 41](#_Toc513417221)

# 1. Техническое задание

## *1.1 Название проекта. Назначение веб-сайта*

*Разработка веб-сайта «Все самое интересное из мира науки о жизни» на основе библиотеки Django (Django – библиотека языка Python, реализующая базовую функциональность веб-сайта).*

*Целью разработки является информирование обучающихся о новых достижениях в области биологии.*

## *1.2 Требования к веб-сайту*

* + 1. Функциональные требования

*Функциональные требования формируются с учетом роли пользователя.*

* *Просматривать страницы сайта могут все посетители.*
* *Гость может зарегистрироваться, чтобы получить возможность комментирования публикуемых статей.*
* *Комментировать статьи могут только зарегистрированные пользователи.*
* *Вести блог может только администратор сайта.*
* *Администратор веб-сайта должен иметь возможность редактировать/удалять статьи блога, удалять нежелательные комментарии, удалять пользователей.*
  + 1. Информационные требования

*Необходимо реализовать следующие основные информационные элементы:*

*a.* ***Главная страница*** *с общей информацией по теме и «горячими» новостями.*

*b.* ***Блог*** *– набор статей о новых достижениях в области биологии.*

*c.* ***Статья*** *– подробное содержание статьи с комментариями посетителей и формой для отправки комментария (для зарегистрированного пользователя).*

*d.* ***Контакты****. E-mail автора веб-сайта.*

*e.* ***Разделы биологии****. Сведения о разделах биологии.*

*f.* ***Регистрация****. На странице должна быть расположена форма регистрации пользователей.*

*g.* ***Вход*** *– страница с формой входа.*

*h.* ***Административный раздел*** *должен содержать следующие основные подразделы****:*** *добавление статьи, редактирование/удаление статьи, удаление комментариев, удаление пользователей.*

# 2. Проектирование веб-сайта

## *2.1 Проектирование карты веб-сайта*

*На рисунке 1 представлена карта публичного раздела веб-сайта, содержащая иерархическую структуру веб-страниц.*



*Рисунок 1 – Карта публичного раздела веб-сайта*

*Страница Регистрация доступна, пока пользователь не авторизовался.*

*На рисунке 2 представлена карта административного раздела веб-сайта.*



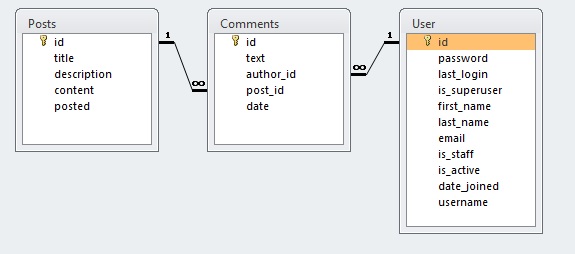
*Рисунок 2 – Карта административного раздела веб-сайта*

*Администратор может добавлять статью блога, изменять и удалять выбранную статью блога. Аналогично администратор может добавлять пользователей, либо изменять данные пользователя (исправлять ошибки ввода) и удалять уже существующего (зарегистрированного) пользователя, а так же администратор может изменять (исправлять ошибки ввода) и удалять комментарии к статьям блога, при необходимости, добавлять свой комментарий.*

## *2.2 Проектирование базы данных веб-сайта*

2.2.1 Проектирование структуры базы данных

*База данных веб-сайта содержит 3 таблицы. Схема базы данных представлена на рисунке 3.*

**

*Рисунок 3 – Схема базы данных*

2.3.2 Описание таблиц базы данных веб-сайта

*В таблице 1 представлена структура таблицы статей блога Posts.*

*Таблица 1*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название поля** | **Тип** | **Описание** |
| 1 | id | bigint | Идентификатор  (ключевое поле) |
| 2 | title | nvarchar | Заголовок статьи |
| 3 | description | text | Краткое описание статьи |
| 4 | content | text | Полный текст статьи |
| 5 | posted | datetime | Дата и время публикации |

*В таблице 2 представлена структура таблицы пользователей User.*

*Таблица 2*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название поля** | **Тип** | **Описание** |
| 1 | id | bigint | Идентификатор  (ключевое поле) |
| 2 | password | nvarchar | Пароль |
| 3 | last\_login | datetime | Дата и время последней авторизации |
| 4 | is\_superuser | bool | Является администратором |
| 5 | first\_name | nvarchar | Фамилия |
| 6 | last\_name | nvarchar | Имя |
| 7 | email | nvarchar | E-mail |
| 8 | is\_staff | bool | Имеет доступ в административный раздел сайта |
| 9 | is\_active | bool | Активный пользователь |
| 10 | date\_joined | datetime | Дата и время регистрации |
| 11 | username | nvarchar | Логин |

*В таблице 3 представлена структура таблицы комментариев к статьям блога Comments.*

*Таблица 3*

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название поля** | **Тип** | **Описание** |
| 1 | id | bigint | Идентификатор  (ключевое поле) |
| 2 | text | nvarchar | Текст комментария |
| 3 | author\_id | bigint | Автор комментария  (вторичный ключ) |
| 4 | post\_id | bigint | Статья  (вторичный ключ) |
| 5 | date | datetime | Дата и время публикации |

## *2.4 Проектирование шаблона Мастер-страницы сайта*

*На рисунке 4 представлен шаблон Мастер-страницы сайта.*

|  |  |  |
| --- | --- | --- |
|  | **2** |  |
|  | **3** |  |
|  |  |  |
| **1** | | |
| **4** | | |

*Рисунок 4 – Шаблон Мастер-страницы сайта*

*На рисунке 4 каждая область отмечена следующими цифрами:*

*1 – это область контента страницы (он у каждой страницы имеет своё содержание)*

*2 – это область «шапки» сайта*

*3 – это область горизонтального меню сайта, расположенная в области 2*

*4 – это область «подвала» сайта*

# 3. Разработка веб-сайта

*Проект разработан на основе библиотеки Django.* *Главной идеей в Django является разделение задач:*

*Файл models.py содержит описание таблицы базы данных, представленное в виде класса Python. Такой класс называется моделью. С помощью данного класса можно создавать, получать, обновлять и удалять записи в таблице базы данных веб-сайта, используя код на языке Python вместо использования повторяющихся SQL команд.*

*Файл views.py содержит логику отображения веб-страниц.*

*Файл urls.py определяет, какое именно представление будет вызвано для URL, заданного в виде шаблона.*

*Файл с расширением html является HTML шаблоном, который описывает дизайн страницы.*

*Объединённые вместе, эти компоненты приложения следуют шаблону Модель-Представление-Контроллер (Model-View-Controller, MVC). MVC определяет способ разработки программного обеспечения, при котором код для определения и доступа к данным (модель) отделён от логики приложения (управление), которая в свою очередь отделена от интерфейса пользователя (представление).*

*Основное преимущество такого подхода заключается в свободе объединения этих компонентов. Следовательно, каждая отдельная часть приложения, созданного с помощью Django, имеет одно назначение и может быть изменена независимо, т.е., без влияния на остальные компоненты. Например, разработчик может изменить URL для данной части приложения без изменения остального кода. Дизайнер может изменить HTML страницы без внесения изменений в код, который отображает страницу. Администратор базы данных может переименовать таблицу и определить эти изменения в одном месте, вместо того, чтобы искать и вносить изменения во множество файлов.*

## *3.1 Разработка классов моделей*

*Модель данных Django – это обычный класс Python, она может включать в себя не только свойства, но и методы, которые могут манипулировать хранящимися в модели данными.*

*Создаваемые модели являются потомками базового класса* ***Model*** *библиотеки Django.*

*Необходимо создать 2 модели:*

* *для представления статей блога;*
* *для представления комментариев пользователей.*

*Код моделей данных должен быть представлен в файле* ***models.py****.*

*Данные блога будут храниться в таблице* ***Posts*** *базы данных проекта* ***db.sqlite3****, созданной на основе модели* ***Blog*** *со следующими полями:*

* *Заголовок статьи (уникальное поле в пределах текущей даты).*
* *Краткое описание статьи (для вывода на странице Блог – списка статей).*
* *Содержимое статьи.*
* *Дата и время публикации статьи*

# Модель данных Блога

class Blog(models.Model):

title = models.CharField(max\_length = 100, unique\_for\_date = "posted", verbose\_name = "Заголовок")

description = models.TextField(verbose\_name = "Краткое содержание")

content = models.TextField(verbose\_name = "Полное содержание")

posted = models.DateTimeField(default = datetime.now(), db\_index = True, verbose\_name = "Опубликована")

def \_\_str\_\_(self): # название, используемое для представления отдельных записей в административном разделе

return self.title

def get\_absolute\_url(self): # строка с уникальным интернет-адресом записи

return reverse("blogpost", kwargs = {"pk": self.pk})

class Meta: # метаданные ‑ вложенный класс, который задает дополнительные параметры модели:

db\_table = "Posts" # имя таблицы для модели

ordering = ["-posted"] # порядок сортировки данных в модели ‑ по убыванию)

verbose\_name = "статья блога" # имя, под которым статья будет отображаться в административном разделе

verbose\_name\_plural = "статьи блога" # тоже для всех статей блога

*Данные комментариев будут храниться в таблице* ***Comments*** *базы данных проекта* ***db.sqlite3****, созданной на основе модели* ***Comment*** *со следующими полями:*

* *Текст комментария.*
* *Дата и время публикации комментария.*
* *Автор (из модели пользователей User).*
* *Статья (из модели статей Blog).*

# Модель комментариев

class Comment(models.Model):

text = models.TextField(verbose\_name = "Комментарий")

date = models.DateTimeField(default = datetime.now(), db\_index = True, verbose\_name = "Дата")

author = models.ForeignKey(User, on\_delete = models.CASCADE, verbose\_name = "Автор") # из модели User (вторичный ключ), каскадное удаление записей в обоих таблицах

post = models.ForeignKey(Blog, on\_delete = models.CASCADE, verbose\_name = "Статья") # из модели Blog (вторичный ключ), каскадное удаление записей в обоих таблицах

class Meta: # метаданные ‑ вложенный класс, который задает дополнительные параметры модели:

db\_table = "Comments

verbose\_name = "Комментарий"

verbose\_name\_plural = "Комментарии статьи блога"

ordering = ["-date"]

*Django предоставляет модель User для хранения данных пользователей. Для ее использования в проекте необходимо добавить в файлы views.py и forms.py строку:*

from django.contrib.auth.models import User

## *3.2 Разработка базы данных веб-сайта*

*Django поддерживает формат баз данных SQLite. СУБД работающая с такими базами данных встроена в Python.*

*Построение базы данных проекта* ***db.sqlite3*** *реализуется на основе моделей данных, описанных в файле* ***models.py****.*

*Для создания таблиц базы данных по моделям данных в Django используются следующие команды:*

***1) Проект-Миграция сборок Django***

*manage.py makemigrations*

***2) Проект-Миграция Django...***

*manage.py migrate*

*При создании таблицы базы данных автоматически добавляется ключевое поле* ***id****.*

## *3.3 Разработка модулей веб-сайта*

*Пакет проекта Biology1 содержит модули проекта. Файл \_\_init\_\_.py, который хранится в папке проекта, обозначает, что данная папка – это пакет проекта.*

* *manage.py – файл хранит код утилиты, которая позволяет выполнять различные действия над проектом.*

*В пакете проекта содержатся следующие модули:*

* *settings.py – хранит настройки проекта в виде набора переменных;*
* *urls.py – хранит сведения о привязке приложений к интернет-адресам;*
* *wsgi.py – служебный модуль, предназначенный для публикации веб-сайта.*

*Приложение Python входит в состав проекта и реализует функциональность разделов веб-сайта. Физически приложение представляет собой пакет, папка которого находится в папке проекта. Пакет приложения формируется библиотекой Django при создании приложения. В состав пакета входят следующие модули:*

* *models.py – хранит код моделей, входящих в состав приложения;*
* *views.py – хранит код контроллеров, входящих в состав приложения;*
* *admin.py – хранит код, задающий параметры административного приложения, входящего в состав библиотеки Django;*
* *test.py - тестовый модуль.*

*Каждый подраздел сайта реализуется контроллером, который входит в состав приложения.*

*Контроллер для обработки данных и передачи данных в шаблон главной страницы* ***home****:*

def home(request):

"""Renders the home page."""

assert isinstance(request, HttpRequest)

return render(

request,

'app/index.html',

{

'title':'Главная',

'year':datetime.now().year,

}

)

*Контроллер для обработки данных и передачи данных в шаблон страницы* ***blog****:*

def blog(request):

"""Renders the blog page."""

posts = Blog.objects.order\_by('-posted') # запрос на выбор всех статей из модели, отсортированных по убыванию даты публикации (-posted)

assert isinstance(request, HttpRequest)

return render(

request,

'app/blog.html',

{

'title':'Блог о генетике',

'posts': posts, # передача в шаблон списка статей

'year':datetime.now().year,

}

)

*Контроллер для обработки данных и передачи данных в шаблон страницы* ***blogpost****:*

def blogpost(request, parametr):

"""Renders the blogpost page."""

post\_1 = Blog.objects.get(id=parametr) # запрос на выбор конкретной статьи

comments = Comment.objects.filter(post=parametr) # запрос на выбор всех комментариев конкретной статьи

if request.method == "POST": # после отправки данных формы на сервер методом POST

form = CommentForm(request.POST) # все поля формы находятся в request.POST

if form.is\_valid(): # проверка корректности формы (все ли необходимые поля заполнены и не отправлено ли некорректных значений)

comment\_f = form.save(commit=False) # сохраняем форму, commit=False означает, что мы пока не хотим сохранять модель Comment

comment\_f.author = request.user # добавляем (так как этого поля нет в форме) в модель Комментария (Comment) в поле автор авторизованного на данный момент пользователя

comment\_f.date = datetime.now() # добавляем в модель Комментария (Comment) текущую дату

comment\_f.post = Blog.objects.get(id=parametr) # добавляем в модель Комментария (Comment) статью, для которой данный комментарий

comment\_f.save() # сохраняем изменения после добавления полей

return redirect('blogpost', parametr=post\_1.id) # делаем переадресацию на ту же страницу статьи после отправки комментария

else:

form = CommentForm() # создание объекта формы для ввода комментария

assert isinstance(request, HttpRequest)

return render(

request,

'app/blogpost.html',

{

'post\_1': post\_1, # передача в шаблон веб-страницы конкретной cтатьи

'comments': comments, # передача в шаблон веб-страницы всех комментариев данной cтатьи

'form': form, # передача в шаблон веб-страницы формы

'year':datetime.now().year,

}

)

*Контроллер для обработки данных и передачи данных в шаблон страницы* ***registration****:*

def registration(request):

"""Renders the registration page."""

if request.method == "POST": # после отправки формы

regform = UserCreationForm(request.POST)

if regform.is\_valid():

reg\_f = regform.save(commit=False)

reg\_f.is\_staff = False # запрещен вход в административный раздел

reg\_f.is\_active = True # активный пользователь

reg\_f.is\_superuser = False # не является суперпользователем

reg\_f.date\_joined = datetime.now() # дата регистрации

reg\_f.last\_login = datetime.now() # дата последней авторизации

reg\_f.save() # сохраняем изменения после добавления полей

return redirect('blog') # переадресация на страницу Блог после регистрации

else:

regform = UserCreationForm() # создание объекта формы для ввода данных

assert isinstance(request, HttpRequest)

return render(

request,

'app/registration.html',

{

'regform': regform, # передача формы в шаблон веб-страницы

'year':datetime.now().year,

}

)

*Контроллер для обработки данных и передачи данных в шаблон страницы* ***sections****:*

def sections(request):

"""Renders the sections page."""

assert isinstance(request, HttpRequest)

return render(

request,

'app/sections.html',

{

'title':'Разделы биологии',

'year': datetime.now().year,

}

)

*Контроллер для обработки данных и передачи данных в шаблон страницы* ***contact****:*

def contact(request):

"""Renders the contact page."""

assert isinstance(request, HttpRequest)

return render(

request,

'app/contact.html',

{

'title':'Контакты со мной',

'year':datetime.now().year,

}

)

*В файл urls.py необходимо добавить URL для всех веб-страниц сайта:*

urlpatterns = [

url(r'^$', app.views.home, name='home'), #задает привязку интернет-адресов к приложениям

url(r'^contact$', app.views.contact, name='contact'),

url(r'^blog$', app.views.blog, name='blog'),

url(r'^(?P<parametr>\d+)/$', app.views.blogpost, name='blogpost'),

url(r'^sections$', app.views.sections, name='sections'),

url(r'^registration$', app.views.registration, name='registration'),

url(r'^login/$',

django.contrib.auth.views.login,

# функция login для входа на сайт

{

'template\_name': 'app/login.html',

'authentication\_form': app.forms.BootstrapAuthenticationForm,

'extra\_context':

{

'title': 'Вход',

'year': datetime.now().year,

}

},

name='login'),

url(r'^logout$',

django.contrib.auth.views.logout,

# функция logout для реализации выхода

{

'next\_page': '/',

},

name='logout'),

url(r'^admin/', include(admin.site.urls)),

]

*Django содержит встроенный административный раздел сайта, который позволяет просматривать содержимое созданных моделей, добавлять новые данные и редактировать существующие.*

*Для этого необходимо зарегистрировать созданные модели, добавив следующие строки в файл models.py:*

admin.site.register(Blog) # регистрация модели в административном разделе для добавления и удаления статей блога

admin.site.register(Comment) # регистрация модели в административном разделе, чтобы admin удалял ненужные комментарии

*Для доступа к административному разделу необходимо в файл urls.py добавить строчки реализующие импорт соответствующих модулей библиотеки, инициализацию административного сайта и использование соответствующего URL для входа в административный раздел:*

from django.conf.urls import include

from django.contrib import admin

admin.autodiscover() # инициализирует административный сайт

url(r'^admin/', include(admin.site.urls)),

*Предварительно необходимо создать пользователя* ***admin****.*

*После этого для входа в административный раздел необходимо добавить в строку браузера* ***/admin****. Например:*

<http://localhost:50740/admin>

*Загружается форма входа в административный раздел. В соответствующие поля формы необходимо ввести данные пользователя admin.*

*После успешного входа можно добавлять, удалять статьи блога, удалять комментарии.*

## *3.4 Разработка шаблонов веб-страниц сайта*

***Шаблон Django*** *– веб-страница, которая включает как HTML-код и разметку (статическая веб-страница), так и особые команды, помещающие в нужное место шаблона какие-либо значения или управляющие формированием кода. Таким образом, реализуется динамика на веб-странице (динамическая веб-страница).*

*Переменные шаблона хранят выводимые в этом шаблоне данные.*

*Пример: <h2> {{ переменная }} </h2>*

*Теги шаблона выполняют какие-либо действия над фрагментами кода шаблона. Они записываются в формате: {% <тег> %}*

*Наследование шаблонов*

*Необходимо вынести повторяющийся код в родительский шаблон – шаблон Мастер-страницы* ***layout.html****, а уникальное для каждой страницы содержимое реализовать в соответствующих им дочерних шаблонах (для этого будем определять блоки шаблона).*

*{% block имя %} {% endblock %}*

*В папке templates/app находятся файлы шаблонов веб-страниц сайта:*

* *layout.html – мастер-страница – используется для создания единой компоновки веб-страниц для сайта в целом.*
* *index.html – главная страница сайта*
* *login.html – страница входа на сайт*
* *loginpartial.html – страница для отображения формы с приветствием для авторизованного пользователя или ссылки на страницу Вход для гостя.*
* *sections.html – страница со сведениями о разделах биологии.*
* *blog.html – страница блога.*
* *blogpost.html – страница статьи блога.*
* *contact.html – страница Контакты.*
* *registration.html – страница регистрации пользователей*

*Код Мастер-страницы веб-сайта layout.html.*

<!DOCTYPE html>

<html>

**<head>**

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"> # ширина области просмотра равняется ширине устройства, запрещено любое масштабирование

<title>**{{** title }}</title>

{% load staticfiles %}

<link href="http://allfont.ru/allfont.css?fonts=pt-sans-bold-italic" rel="stylesheet" type="text/css" />

<link href="http://allfont.ru/allfont.css?fonts=heinrichscript" rel="stylesheet" type="text/css" />

<link href="http://allfont.ru/allfont.css?fonts=a\_campusgrav" rel="stylesheet" type="text/css" />

<link href="http://allfont.ru/allfont.css?fonts=a\_campusotlsh-bold" rel="stylesheet" type="text/css" />

<link href="http://allfont.ru/allfont.css?fonts=1isadora-m-bold" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="{% static 'app/content/bootstrap.css' %}" />

<link rel="stylesheet" type="text/css" href="{% static 'app/content/site.css' %}" />

<script src="{% static 'app/scripts/modernizr-2.6.2.js' %}"></script>

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

<a href="/" class="navbar-brand">Все самое интересное из мира науки о жизни</a>

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

{#<li><a href="{% url 'home' %}">Главная</a></li>#}

<li><a href="{% url 'sections' %}">Разделы биологии</a></li>

<li><a href="{% url 'contact' %}">Контакты</a></li>

<li><a href="{% url 'blog' %}">Блог</a></li>

{% if not user.is\_authenticated %}

<li><a href="{% url 'registration' %}">Регистрация</a></li>

{% endif %}

</ul>

{% include 'app/loginpartial.html' %}

</div>

</div>

</div>

<div class="container body-content">

{% block content %}{% endblock %}

<hr/>

<footer>

<p>&copy; **{{** year }} - MIPT - Dolgoprudny</p>

</footer>

</div>

<script src="{% static 'app/scripts/jquery-1.10.2.js' %}"></script>

<script src="{% static 'app/scripts/bootstrap.js' %}"></script>

<script src="{% static 'app/scripts/respond.js' %}"></script>

{% block scripts %}{% endblock %}

</body>

</html>

*Код главной страницы веб-сайта index.html.*

{% extends "app/layout.html" %}

{% block content %}

<div class="jumbotron">

<h1 class="black">Генетика</h1>

<p><a href="https://hi-news.ru/tag/genetika" target ="\_blank" class="btn btn-primary btn-large">Узнать больше</a></p>

</div>

<div class="row">

<div class="col-md-6">

<h2 class="black">"Тупайи против!", или коротко о том, как можно жить не по правилам</h2>

<p>

В 1964 году зоолог Дж. Фостер опубликовал статью,

в которой утверждалось, что мелкие зверушки на островах крупнее,

чем на материке (островной гигантизм).

Также с 1847 года известно правило немецкого биолога К. Бергмана:

в холодных климатах обитают виды с более крупными размерами тела, чем в жарких.

Но милые маленькие зверьки тупайи отказались подчиняться этим законам.

</p>

<p><a class="btn btn-default" href="http://batrachospermum.ru/2018/01/tupaia-vs-rules/" target="\_blank">Подробнее &raquo;</a></p>

</div>

<div class="col-md-6">

<h2 class="black">12 паразитов, способных управлять поведением своих хозяев</h2>

<p>

Огромное количество паразитов населяет нашу планету.

Некоторые просто живут за счет других организмов, но

есть те, кто нашел способ жить с максимальным комфортом.

Предлагаем вашему вниманию ссылку на статью сообщества

"Наука и Технологии. Астрономия и Космонавтика", в которой описываются самые

удивительные паразиты, имеющие уникальную способность

манипулировать своими "хозяевами".</p>

<p><a class="btn btn-default" href="https://vk.com/@science\_newworld-12-parazitov-sposobnyh-upravlyat-povedeniem-svoih-hozyaev" target="\_blank">Подробнее &raquo;</a></p>

</div>

</div>

{% endblock %}

*Код страницы Вход login.html.*

{% extends "app/layout.html" %}

{% block content %}

<h2 class="white">**{{** title }}</h2>

<hr />

<h4 class="white">Используйте ваш логин для входа</h4>

<hr />

<div class="row">

<div class="col-md-8">

<section id="loginForm">

<form action="." method="post" class="form-horizontal">

{% csrf\_token %}

<div class="form-group">

<label for="id\_username" class="col-md-2 control-label white">Имя пользователя</label>

<div class="col-md-10">

**{{** form.username }}

</div>

</div>

<div class="form-group">

<label for="id\_password" class="col-md-2 control-label white">Ваш пароль</label>

<div class="col-md-10">

**{{** form.password }}

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="hidden" name="next" value="/" />

<input type="submit" value="Выполнить вход" class="btn btn-default" />

</div>

</div>

{% if form.errors %}

<p class="validation-summary-errors">Пожалуйста</p>

{% endif %}

</form>

</section>

</div>

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

<section id="socialLoginForm"></section>

</div>

</div>

{% endblock %}

{% block scripts %}

{% load staticfiles %}

<script src="{% static 'app/scripts/jquery.validate.min.js' %}"></script>

{% endblock %}

*Код страницы Разделы биологии sections.html.*

{% extends "app/layout.html" %}

{% block content %}

{% load staticfiles %}

<h2 class="white">**{{** title }}.</h2>

<h3>**{{** message }}</h3>

<div>

<table>

<tr>

<td class="red bolding"><h2>Генетика</h2></td>

<td><img src="{% static "app/content/chromosoma.png" %}" alt="Генетика" /></td>

<td><p class="sections">ГЕНЕТИКА (греч. genetikos относящийся к происхождению) — наука о наследственности и изменчивости организмов.

<br />

Предметом изучения генетики являются два свойства организмов — наследственность

и изменчивость. Наследственность — свойство организмов передавать следующему поколению

присущие данному организму особенности становления в ходе онтогенеза определенных

черт строения и типа обмена веществ. Передача особенностей организма следующим поколениям

возможна только в процессе размножения или самовоспроизведения.

<br />

Другим свойством организмов, входящим в предмет исследования генетики,

является изменчивость. Изменчивость — свойство живых организмов,

заключающееся в изменении генов и их проявления в процессе развития организма,

т. е. изменчивость является свойством, противоположным наследственности.</p></td>

</tr>

<tr>

<td class="red bolding"><h2>Цитология</h2></td>

<td><img src="{% static "app/content/cell.png" %}" alt="Цитология" /></td>

<td><p class="sections">ЦИТОЛОГИЯ (греческий kytos вместилище, здесь — клетка + logos учение) — наука о строении, функциях

и развитии клеток животных и растений, а также одноклеточных организмов и бактерий.

Цитологические исследования имеют существенное значение для диагностики заболеваний человека и животных.

<br />

Развитие цитологии исторически связано с созданием и усовершенствованием микроскопа

и гистологических методов исследования. Термин «клетка» впервые был применен Гуком,

описавшим клеточное строение (точнее — целлюлозные оболочки клеток) ряда растительных тканей.

В 17 веке наблюдения Гука были подтверждены и развиты М. Мальпиги, Грю, А. Левенгуком.

В 1781 году Фонтана опубликовал рисунки животных клеток с ядрами.</p></td>

</tr>

<tr>

<td class="red bolding"><h2>Ботаника</h2></td>

<td><img src="{% static "app/content/flower.png" %}" alt="Ботаника" /></td>

<td><p class="sections">БОТАНИКА (греч, botanikos относящийся к растениям) — наука о растениях:

их строении, жизнедеятельности, классификации, распространении, историческом развитии,

значении в жизни природы и человека; один из основных разделов биологии.

<br />

Морфология растений изучает закономерности строения и формообразования растений

в процессе их индивидуального и исторического развития.

<br />

Физиология растений изучает жизненные процессы в растениях.

<br />

Систематика растений изучает разнообразие растений, разделяет растительный мир на

соподчиненные единицы — таксоны, располагает отдельные таксоны

(виды, роды, семейства и др.) в единую систему в соответствии с их родственными

отношениями и эволюционным развитием (филогения), устанавливает рациональные их

названия (номенклатура).

</p></td>

</tr>

<tr>

<td class="red bolding"><h2>Зоология</h2></td>

<td><img src="{% static "app/content/enot.png" %}" alt="Зоология" /></td>

<td><p class="sections">ЗООЛОГИЯ (греч, zoon животное + logos учение) — раздел биологии,

изучающий животных. 3оология тесно связана с практической деятельностью человека,

освоением и охраной животного мира как естественной составной части биосферы Земли.

<br />

Строение и функции организма, отдельных его органов и тканей изучаются

методами морфологии, физиологии, гистологии, цитологии, физиологии и других наук.

Исследование процесса индивидуального развития животного (его онтогенеза) — предмет

изучения эмбриологии. Закономерности наследования признаков и их изменчивости

изучает генетика, поведение животных — этология, взаимосвязи животных со средой

их обитания — экология, взаимоотношение паразитов и их хозяев — паразитология

и т. д.

<br />

Становление зоологии как науки связывают с именем древнегреческого

ученого Аристотеля (4 в. до н. э.), обобщившего имеющиеся достижения

в изучении животных.</p></td>

</tr>

<tr>

<td class="red bolding"><h2>Анатомия</h2></td>

<td><img src="{% static "app/content/heart.png" %}" alt="Анатомия" /></td>

<td><p class="sections">АНАТОМИЯ (греческий anatomē — рассечение, расчленение) — наука

о строении и формах организма, органов и тканей человека и животных.

<br />

Данная наука исследует тело человека и животных, системы органов и

образующих их тканей в их функциональном взаимодействии, в процессе

становления и развития. В анатомии, как в любой науке, анализ сочетается

с синтезом, за подробным описанием строения следует раскрытие его

причинной обусловленности и исследуются объективные законы организации

живых систем.

<br />

Анатомия человека дает систематизированное описание формы, строения,

положения и топографических взаимоотношений частей тела и органов с

учетом их возрастных изменений, половых различий и индивидуальных

особенностей. </p></td>

</tr>

</table>

</div>

{% endblock %}

*Код страницы Блог blog.html.*

{% extends "app/layout.html" %}

{% block content %}

{% load staticfiles %}

<h2 class="white">**{{** title }}.</h2>

<br/>

{% for post in posts %}

<hr/>

<div class="">

<h2 class="white"> **{{** post.title }} </h2>

<p> **{{** post.posted }} </p>

</div>

<p><a class="blog" href="{% url 'blogpost' parametr=post.id %}">**{{** post.description }}</a> </p>

{% endfor %}

{% endblock %}

*Код страницы Статья blogpost.html.*

{% extends "app/layout.html" %}

{% block content %}

{% load staticfiles %}

<h2 class="white"> **{{** post\_1.title }} </h2>

<p> **{{** post\_1.content|linebreaksbr }} </p>

<hr/>

{% if user.is\_authenticated %}

<h3 class="white bolding"><i>Добавить комментарий</i></h3>

<form method="POST" >{% csrf\_token %}

<div>

**{{** form.as\_p }}

<button type="submit" class="save btn btn-default" > Отправить комментарий </button>

</div>

</form>

{% else %}

<h3 class="red"> <i>Для добавления своего комментария, пожалуйста, авторизуйтесь</i></h3>

{% endif %}

{% for comment in comments %}

<hr/>

<h5 class="white"> **{{** comment.date }} </h5>

<h5 class="red bolding"><u> **{{** comment.author }} </u></h5>

<p class="comm"> **{{** comment.text|linebreaksbr }} </p>

{% endfor %}

{% endblock %}

*Код страницы Контакты contact.html.*

{% extends "app/layout.html" %}

{% block content %}

<h2 class="white bolding">**{{** title }}.</h2>

<address>

<strong class="white">Со всеми вопросами и предложениями можно обратиться по данному адресу:</strong> <a class="contact" href="mailto:katarina\_2503@mail.ru">nevomasha@phystech.edu</a><br />

</address>

{% endblock %}

*Код страницы Регистрация registration.html.*

{% extends "app/layout.html" %}

{% block content %}

<h2 class="white">Регистрация пользователя</h2>

<hr />

<form method="POST" > {% csrf\_token %}

<div>

<table>

<tr>

<td class="white"> Введите Ваш логин</td>

<td> **{{** regform.username }}</td>

</tr>

<tr>

<td class="white"> Введите пароль (не менее 8 символов)</td>

<td> **{{** regform.password1 }}</td>

</tr>

<tr>

<td class="white"> Подтверждение пароля</td>

<td> **{{** regform.password2 }}</td>

</tr>

<tr>

<td> </td>

<td> <button type="submit" class="save btn btn-default" > Зарегистрировать </button></td>

</tr>

</table>

</div>

</form>

{% endblock %}

## *3.5 Разработка форм*

*Код разрабатываемых форм должен содержаться в файле* ***forms.py****.*

*Код класса формы для добавления комментария к статье авторизованным пользователем:*

class CommentForm(forms.ModelForm):

class Meta:

model = Comment # данная модель будет использоваться для создания формы

fields = ('text',) # требуется ввести только поле text

labels = {'text': "Комментарий"} # метка к полю формы text

# author будет автоматически выбран в зависимости от авторизованного пользователя

# date должна автоматически проставляться в момент создания записи

*Код класса формы для авторизации пользователя:*

class BootstrapAuthenticationForm(AuthenticationForm):

username = forms.CharField(max\_length=254,

widget=forms.TextInput({

'class': 'form-control',

'placeholder': 'Логин'}))

password = forms.CharField(label=\_("Password"),

widget=forms.PasswordInput({

'class': 'form-control',

'placeholder':'Пароль'}))

*Django предоставляет встроенную форму – класс* ***UserCreationForm –*** *для регистрации пользователя на сайте. Для ее использования в проекте необходимо в файл views.py добавить строку:*

from django.contrib.auth.forms import UserCreationForm

*У формы UserCreationForm 3 поля:*

*• username – для ввода логина пользователя;*

*• password1 – для ввода пароля пользователя;*

*• password2 – для подтверждения пароля.*

*Создание объекта формы и передача формы в шаблон веб-страницы реализуется в соответствующем контроллере.*

*После отправки формы происходит возврат к тому же представлению, но в этот раз с новыми данными в request.POST*

*Получается, что в соответствующем контроллере в файле view.py необходимо обработать две разные ситуации:*

1. *когда вошлина страницу и хотим получить пустую форму;*
2. *когда реализуется возврат к представлению со всей информацией, которая была введена в форму.*

*Таким образом, необходимо ввести условие (использовать условный оператор if):*

*if request.method == "POST":*

*[...]*

*else:*

*form = CommentForm()*

*Форма встраивается в шаблон соответствующей веб-страницы:*

* *форма авторизации – в шаблон страницы Вход,*
* *форма регистрации пользователей – в шаблон страницы Регистрация,*
* *форма добавления комментария – в шаблон страницы Статья.*

## *3.6 Разработка дизайна страниц (стилевое оформление страниц)*

*Для уникального оформления веб-страниц сайта была разработана фоновая картинка (файл backgrouder.jpg), представленная на рисунке 5, настроена требуемая прозрачность фона, загружены дополнительные шрифты.*

*Разработанные классы стилей находятся в файле site.css, также были добавлены свои стили и изменены существующие в файле bootstrap.css.*

**

*Рисунок 5 – Фоновая картинка страниц сайта*

*Содержание файла site.css*

body {

background-image: url('backgrouder.jpg');

background-position: center center;

background-repeat: no-repeat;

background-attachment: fixed;

background-size: cover;

background-color: #00e7cf;

padding-top: 50px;

padding-bottom: 20px;

}

/\* Set padding to keep content from hitting the edges \*/

.body-content {

background-image: url(http://vfl.ru/fotos/82603c4720937513.html);

background-position: center center;

background-repeat: no-repeat;

background-attachment: fixed;

background-size: cover;

background-color: rgba(0, 239, 220, 0.3);

/\*padding-left: 15px;\*/

/\*padding-right: 15px;\*/

}

/\* Set width on the form input elements since they're 100% wide by default \*/

input,

select,

textarea {

max-width: 280px;

}

/\* styles for validation helpers \*/

.field-validation-error {

color: #00ff21;

}

.field-validation-valid {

display: none;

}

input.input-validation-error {

border: 1px solid #b94a48;

}

input[type="checkbox"].input-validation-error {

border: 0 none;

}

.validation-summary-errors {

color: #b94a48;

}

.validation-summary-valid {

display: none;

}

th, td {

padding: 20px;

}

.white {

font-family: arial;

font-size: 16px;

color: white;

font-style: oblique;

}

.red {

font-family: arial;

font-size: 16px;

color: orangered;

font-style: oblique;

}

.bolding {

font-weight: 700;

}

h1.black {

font-family: '1Isadora M Bold', arial;

font-size: 48px;

text-shadow: 4px 4px 4px #00e7cf;

color: black

}

h1.white {

font-family: '1Isadora M Bold', arial;

font-size: 48px;

text-shadow: 4px 4px 4px #00e7cf;

color: white

}

h2.black {

font-family: 'PT Sans Bold Italic', arial;

font-size: 32px;

color: black

}

h2.white {

font-family: 'PT Sans Bold Italic', arial;

font-size: 32px;

color: white;

}

p {

font-family: arial;

font-size: 18px;

color: white;

font-style: italic;

text-align: justify

}

p.sections {

font-family: 'HeinrichScript', arial;

font-size: 20px;

color: orangered;

font-style: italic;

text-align: justify

}

p.comm {

font-family: 'HeinrichScript', arial;

font-size: 24px;

color: orangered;

font-style: italic;

text-align: justify

}

a.blog {

font-family: 'HeinrichScript', arial;

font-size: 26px;

font-style: italic;

color: white;

}

a.contact {

font-family: 'HeinrichScript', arial;

font-size: 26px;

font-style: italic;

color: orangered;

}

a:hover {

color: white;

text-decoration: underline;

}

*Измененные стили файла bootstrap.css*

*А) для меню*

.navbar-inverse {

background-color: #181c2b;

border-color: #080808;

}

.navbar-inverse .navbar-brand {

color: #3b98ff;

}

.navbar-inverse .navbar-brand:hover,

.navbar-inverse .navbar-brand:focus {

color: #ffffff;

background-color: transparent;

}

.navbar-inverse .navbar-nav > li > a {

color: #3b98ff;

}

.navbar-inverse .navbar-nav > li > a:hover,

.navbar-inverse .navbar-nav > li > a:focus {

color: #ffffff;

background-color: transparent;

}

.navbar-inverse .navbar-nav > .active > a,

.navbar-inverse .navbar-nav > .active > a:hover,

.navbar-inverse .navbar-nav > .active > a:focus {

color: #ffffff;

background-color: #080808;

}

.navbar-inverse .navbar-nav > .disabled > a,

.navbar-inverse .navbar-nav > .disabled > a:hover,

.navbar-inverse .navbar-nav > .disabled > a:focus {

color: #444444;

background-color: transparent;

}

.navbar-inverse .navbar-toggle {

border-color: #333333;

}

.navbar-inverse .navbar-toggle:hover,

.navbar-inverse .navbar-toggle:focus {

background-color: #333333;

}

.navbar-inverse .navbar-toggle .icon-bar {

background-color: #ffffff;

}

.navbar-inverse .navbar-collapse,

.navbar-inverse .navbar-form {

border-color: #101010;

}

.navbar-inverse .navbar-nav > .open > a,

.navbar-inverse .navbar-nav > .open > a:hover,

.navbar-inverse .navbar-nav > .open > a:focus {

color: #ffffff;

background-color: #080808;

}

.navbar-inverse .navbar-nav > .dropdown > a:hover .caret {

border-top-color: #ffffff;

border-bottom-color: #ffffff;

}

*Б) для кнопки на главной странице*

btn-primary {

color: #ffffff;

background-color: #4ca1ea;

border-color: #357ebd;

font-family: 'HeinrichScript', arial;

font-size: 24px;

font-style: italic

}

.btn-primary:hover,

.btn-primary:focus,

.btn-primary:active,

.btn-primary.active,

.open .dropdown-toggle.btn-primary {

color: #ffffff;

background-color: #3276b1;

border-color: #285e8e;

}

*В) для кнопок*

.btn-default {

color: #050a2b;

background-color: #ffffff;

border-color: #050a2b;

border-width: 2px;

font-family: 'a\_CampusOtlSh Bold', arial;

font-size: 20px;

font-style: normal

}

.btn-default:hover,

.btn-default:focus,

.btn-default:active,

.btn-default.active,

.open .dropdown-toggle.btn-default {

color: #333333;

background-color: #ebebeb;

border-color: #adadad;

}

# 4. Описание пользовательского интерфейса

## *4.1 Руководство пользователя*

*При загрузке сайта открывается главная страница веб-сайта, представленная на рисунке 6.*



*Рисунок 6 – Главная страница сайта*

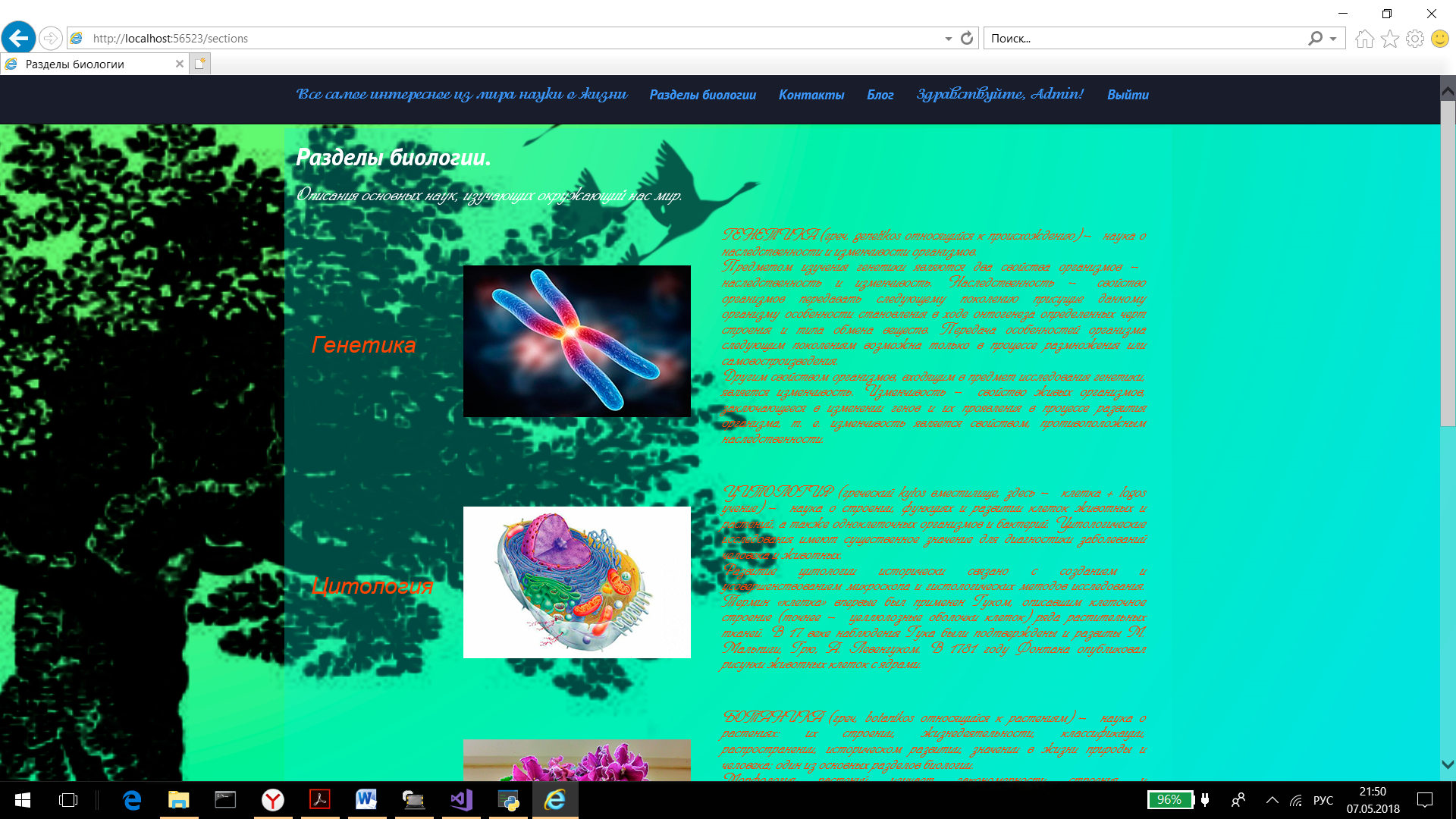
*1) На главной странице веб-сайта при нажатии кнопки «Узнать больше» реализован переход на сайт про генетику.*

*При нажатии кнопки «Подробнее» под анонсом статьи «Тупайи против!, или коротко о том, как жить не по правилам» реализован переход на сайт с полным содержанием статьи. Аналогично, при нажатии кнопки «Подробнее» под анонсом статьи «12 паразитов, способных управлять поведением своих хозяев» осуществляется переход на сайт с полным содержанием данной статьи.*

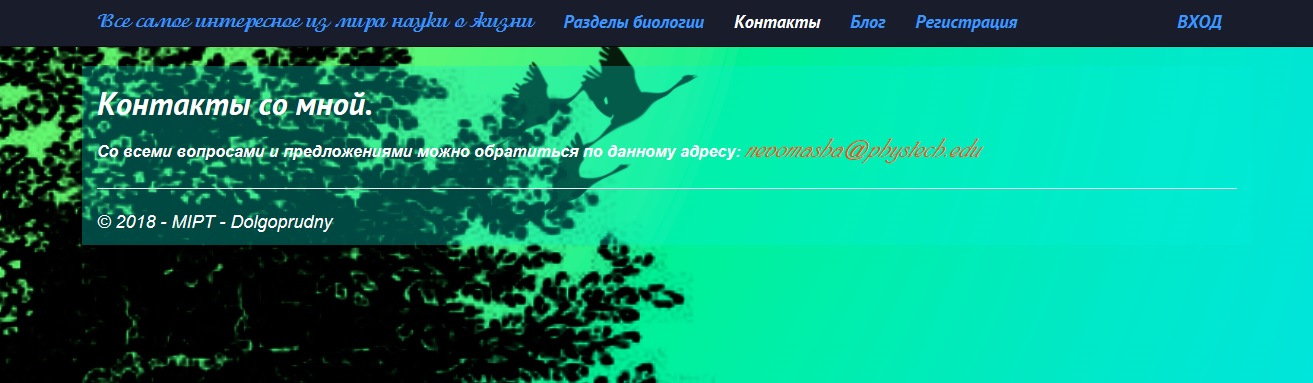
*2) Гость может посмотреть следующие страницы сайта, нажав левой клавишей мыши на соответствующем пункте меню:*

* *Страница сайта «Разделы биологии», представленная на рисунке 7.*
* *Страница сайта «Контакты», представленная на рисунке 8.*
* *Страница сайта «Блог», представленная на рисунке 9. Дальше Гость может перейти на страницу конкретной статьи блога, которая представлена на рисунке 10. Ниже статьи расположены комментарии зарегистрированных пользователей к данной статье (рисунок 11).*

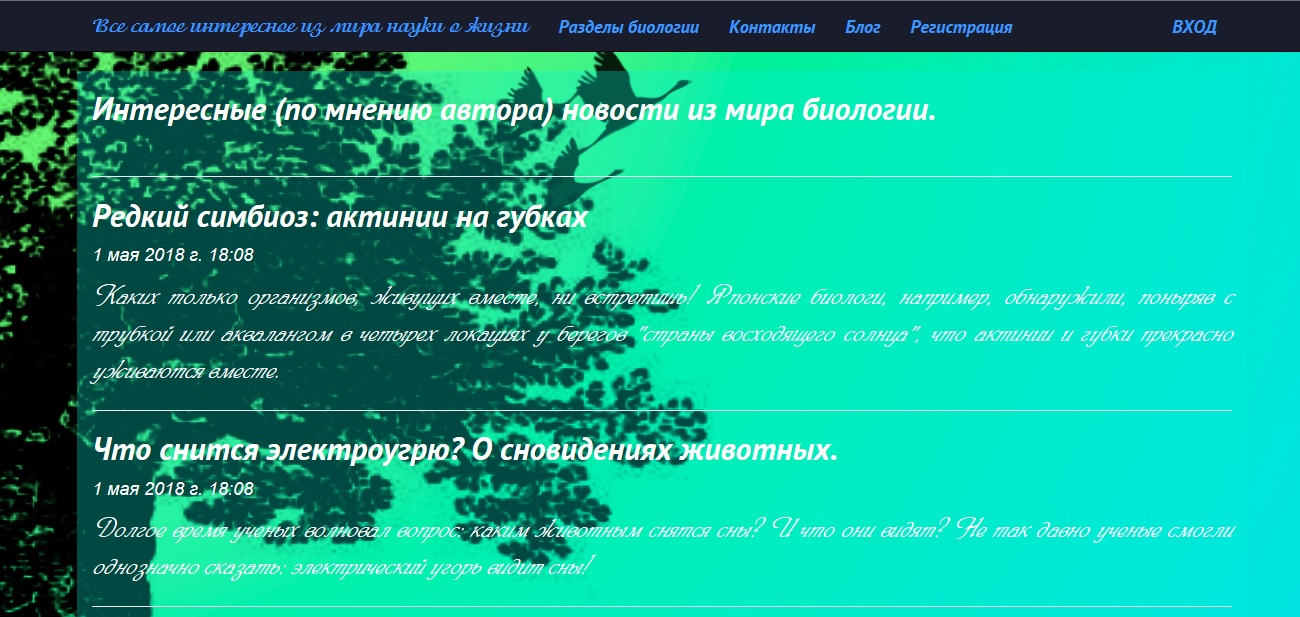
*При необходимости посетитель может перейти с любой страницы веб-сайта на главную страницу, выбрав в меню пункт «Все самое интересное из мира науки о жизни».*



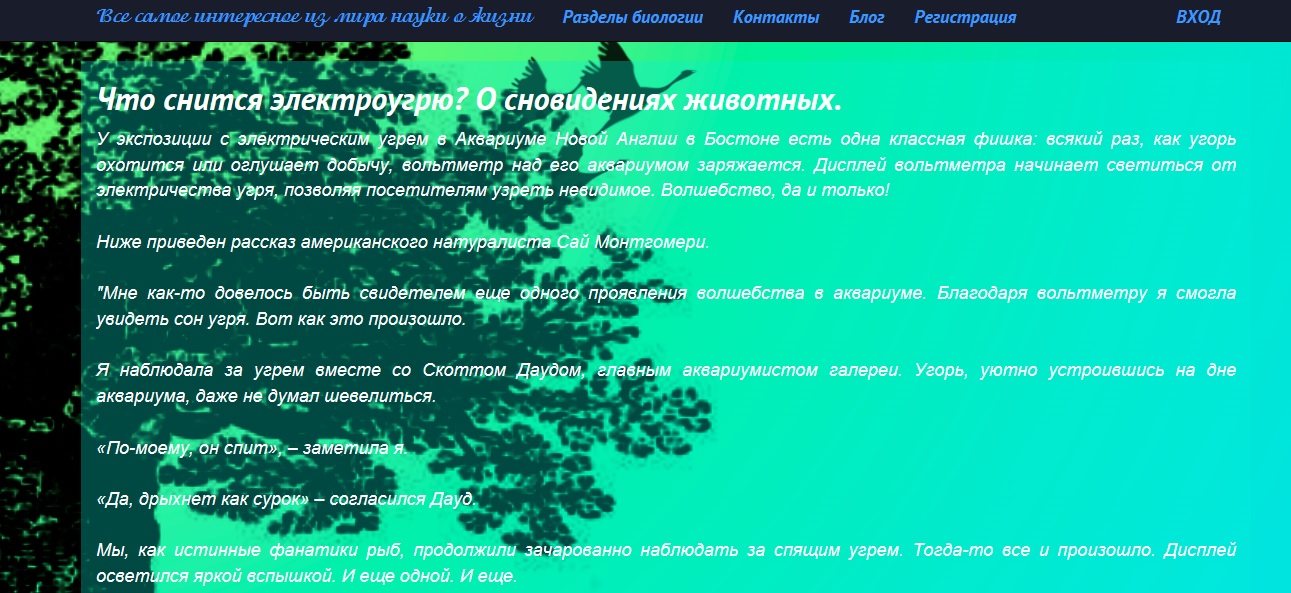
*Рисунок 7 – Страница сайта «Разделы биологии»*

**

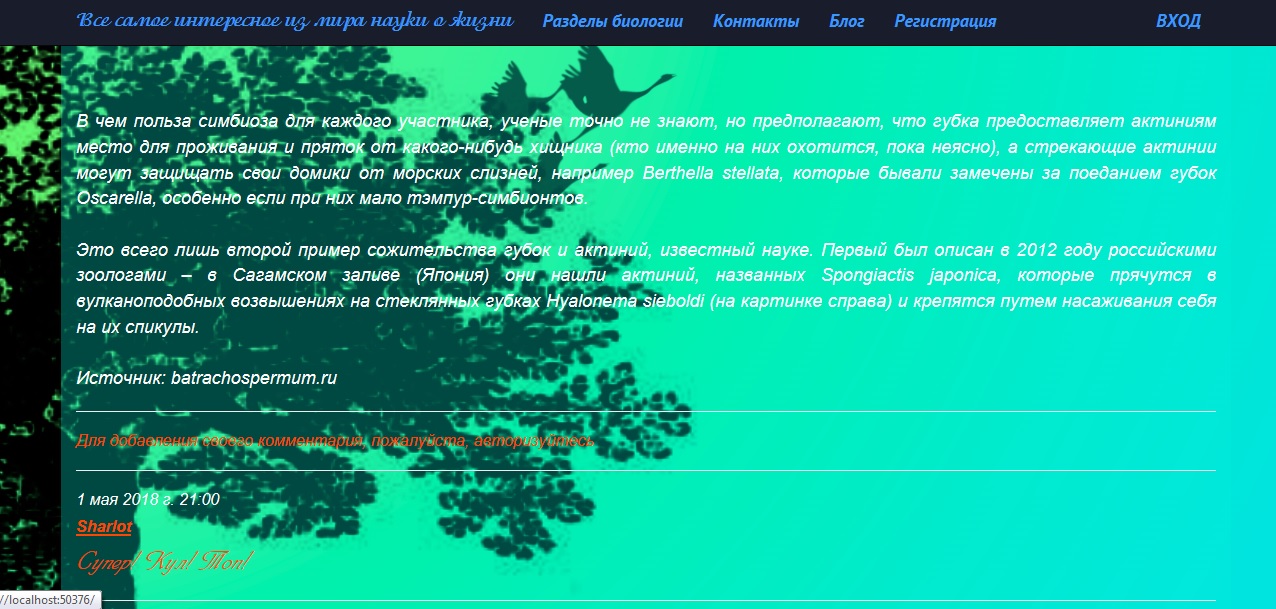
*Рисунок 8 – Страница сайта «Контакты»*

**

*Рисунок 9 – Страница сайта «Блог»*

**

*Рисунок 10 – Страница сайта «Статья блога»*

**

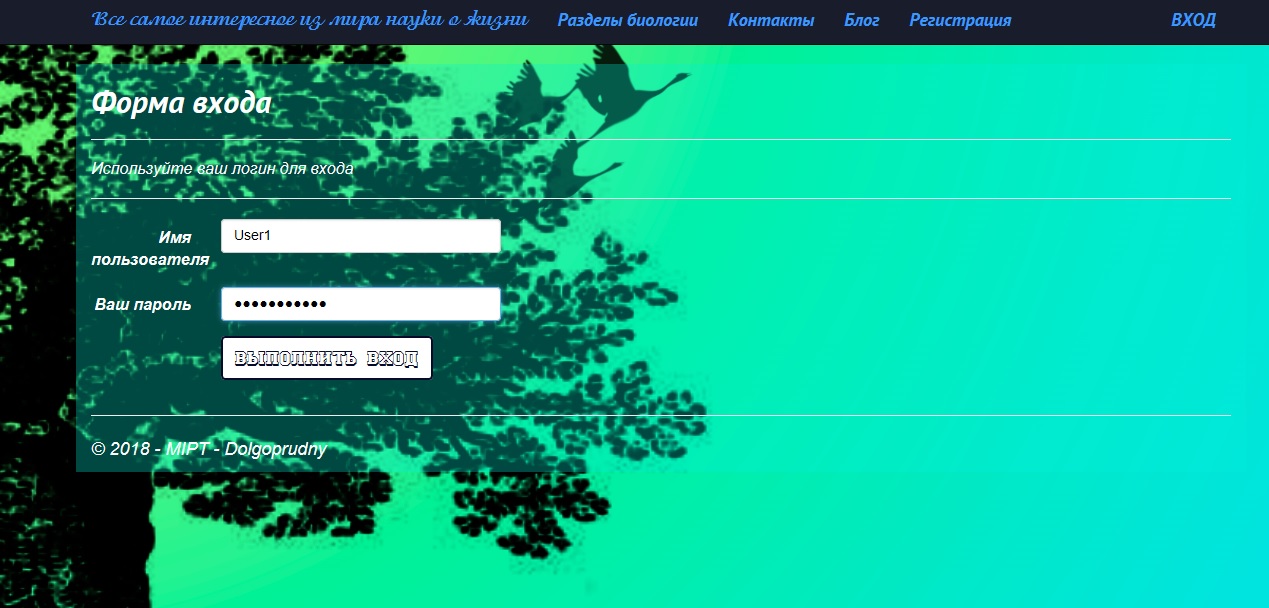
*Рисунок 11 – Страница сайта «Статья блога» с комментариями зарегистрированных пользователей*

*3) Пользователь может зарегистрироваться на данном сайте. Для этого он должен перейти на страницу «Регистрация» и заполнить форму регистрации, которая представлена на рисунке 12. Пароль должен содержать не менее 8 символов.*

**

*Рисунок 12 – Страница сайта «Регистрация»*

*4) Затем посетитель может авторизоваться на сайте, нажав левой клавишей мыши на ссылке «Вход», и ввести в поля формы соответственно свои Логин и Пароль (рисунок 13), нажать кнопку «Выполнить вход». Авторизованный пользователь попадает на главную страницу сайта, на которой отображается приветствие: «Здравствуйте, <Имя пользователя>!» (рисунок 14). Страница регистрация становится недоступной для авторизованного пользователя. Вместо ссылки «Вход» отображается ссылка «Выйти».*

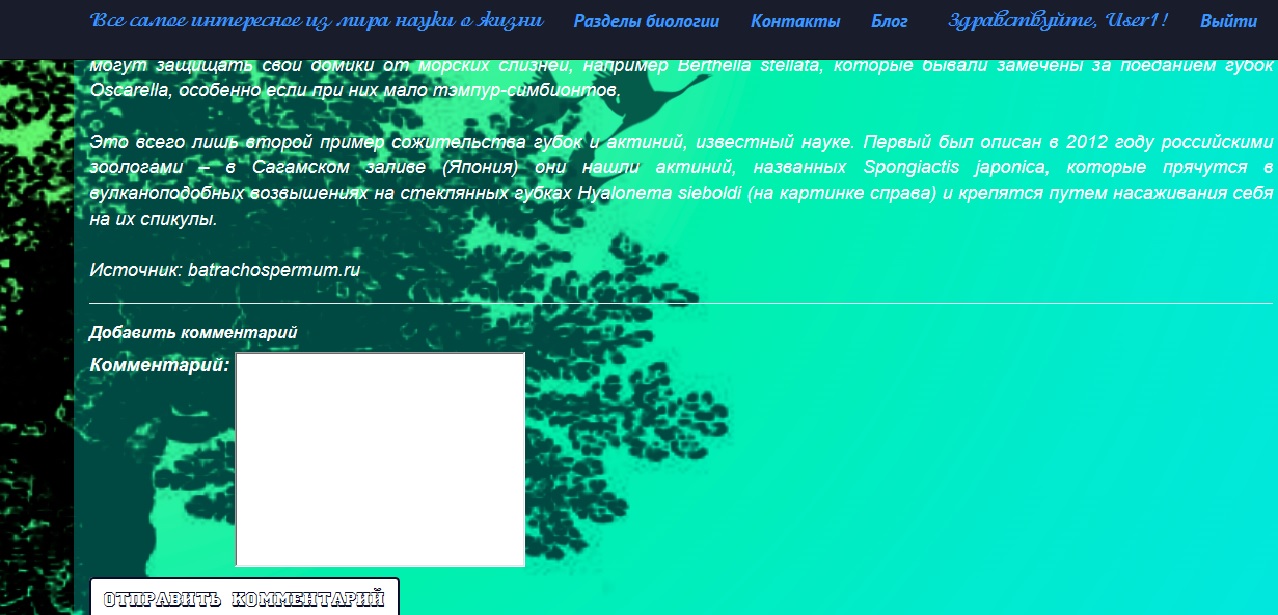
**

*Рисунок 13 – Страница сайта «Вход»*

**

*Рисунок 14 – Вид главной страницы сайта, когда вошел пользователь User1*

*5) Зарегистрированный пользователь может оставлять комментарии к статьям блога (рисунок 15) после осуществления входа на сайт. Поэтому страница «Статья блога» для авторизованного пользователя изменяется по сравнению с видом страницы для Гостя – отображается форма для ввода и отправки комментария к статье. Пользователь может ввести текст в поле «Комментарий» и нажать кнопку «Отправить комментарий». Текст комментария с именем пользователя, датой и временем отправки комментария выводится в списке комментариев к статье внизу страницы (см. рисунок 11). Комментарии выводятся в отсортированном по убыванию даты виде.*

**

*Рисунок 15 – Добавление комментария к статье блога пользователем User1*

*Вид страниц «Разделы биологии», «Блог» и «Контакты» одинаковый и для авторизованного пользователя, и для Гостя.*

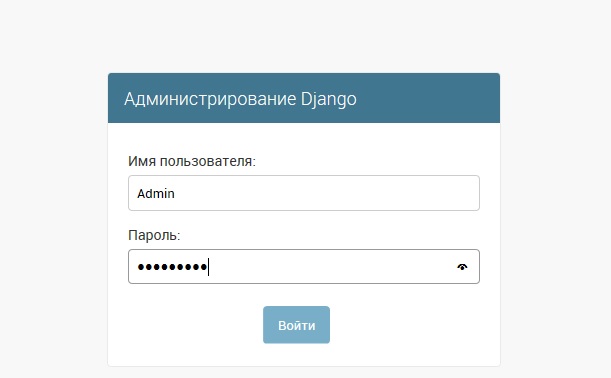
*Ссылка «Выйти» позволяет завершить авторизованный сеанс пользователя.*

## *4.2 Руководство для администратора*

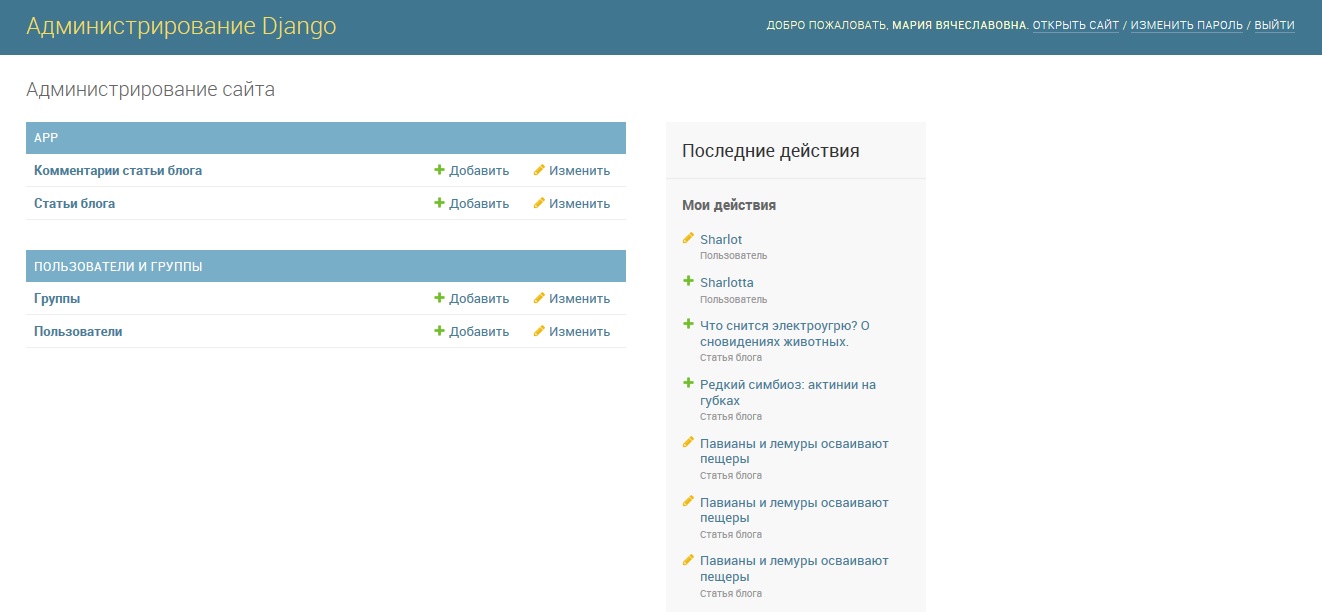
*Для того чтобы войти в административный раздел, необходимо набрать в строке браузера «admin» (рисунок 16) – открывается форма входа в административный раздел (рисунок 17), в поля которой соответственно необходимо ввести Логин и пароль администратора. После нажатия кнопки «Войти» загружается административный раздел веб-сайта (рисунок 18).*

**

*Рисунок 16 – Переход в административный раздел сайта*

**

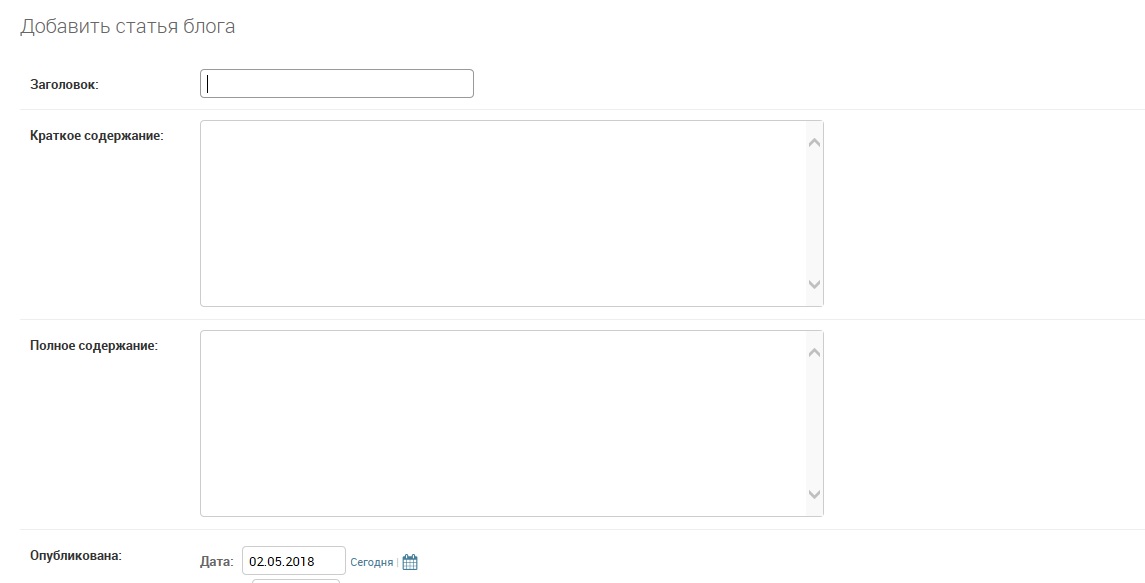
*Рисунок 17 – Форма входа в административный раздел*

**

*Рисунок 18 – Административный раздел сайта*

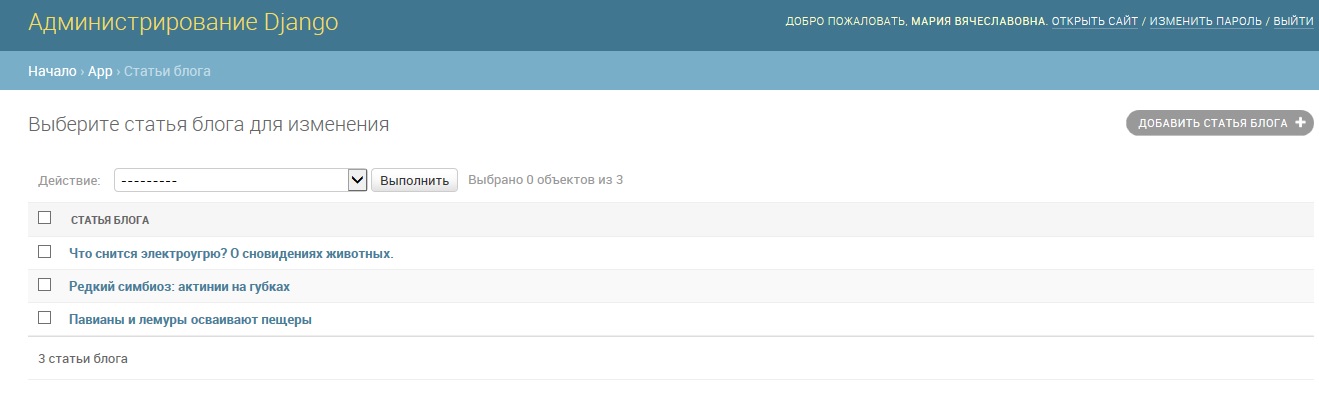
*Изменение, удаление, добавление всех объектов, зарегистрированных в административном разделе – статей блога, комментариев, пользователей – реализуется единообразно.*

*1) Для добавления новой статьи блога необходимо в административном разделе рядом со ссылкой на подраздел «Статьи блога» нажать кнопку «+Добавить», заполнить поля формы (рисунок 19) и нажать кнопку «Сохранить».*

**

*Рисунок 19 – Добавление администратором статьи блога*

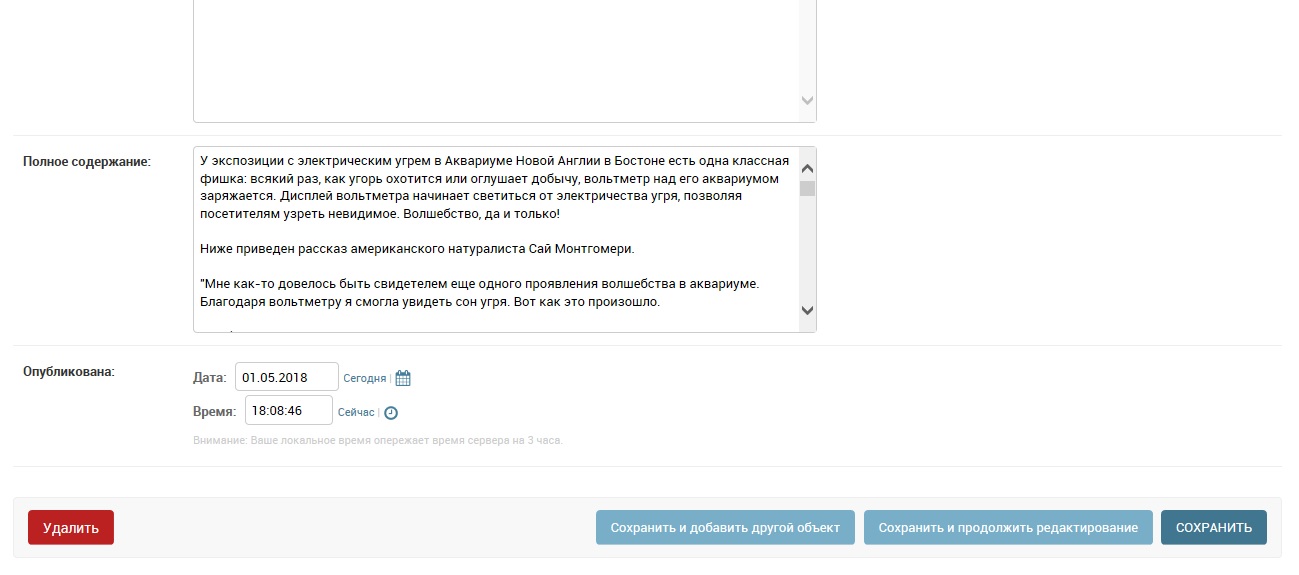
*2) Для изменения добавленных ранее статей блога необходимо войти в подраздел «Статьи блога» (рисунок 20). Перейти по ссылке названия статьи блога и изменить данные статьи в форме (рисунок 21) и нажать кнопку «Сохранить» (рисунок 22).*

**

*Рисунок 20 – Список статей блога для внесения изменений в содержание статьи*

**

*Рисунок 21 – Изменение статьи блога*

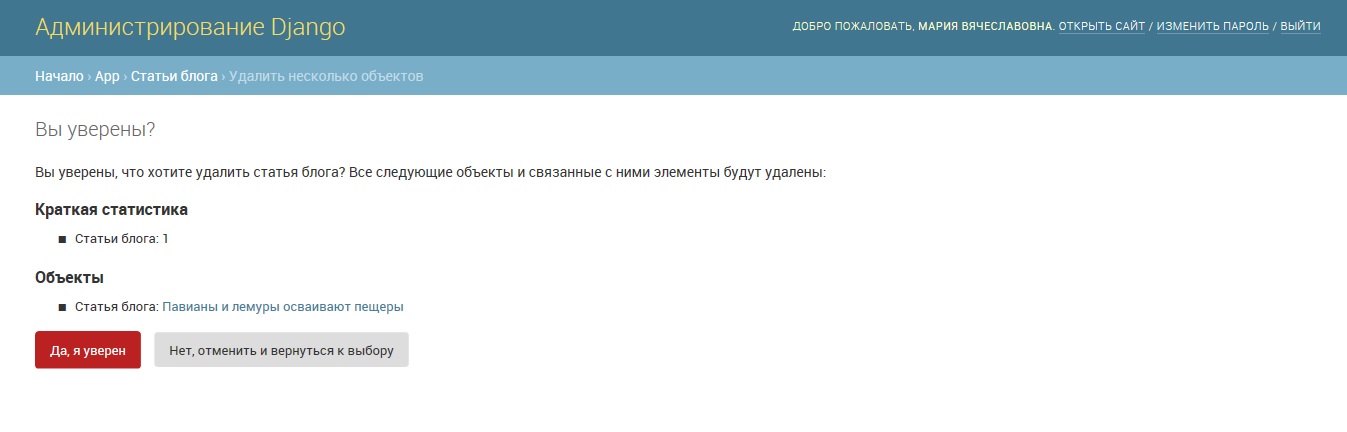
**

*Рисунок 22 – Сохранение изменений статьи блога*

*3) Для удаления статьи (статей) блога необходимо в подразделе «Статьи блога» выбрать статью (статьи) – установить галочку в элементе управления CheckBox (флажок), в списке «Действие» выбрать пункт «Удалить выбранные статьи блога» и нажать кнопку «Выполнить» (рисунок 23). Затем необходимо подтвердить удаление статьи (рисунок 24). Если нажать кнопку «Да, я уверен», то статья будет удалена вместе с её комментариями. Если нажать кнопку «Нет, отменить и вернуться к выбору», то будет отменено удаление статьи.*

**

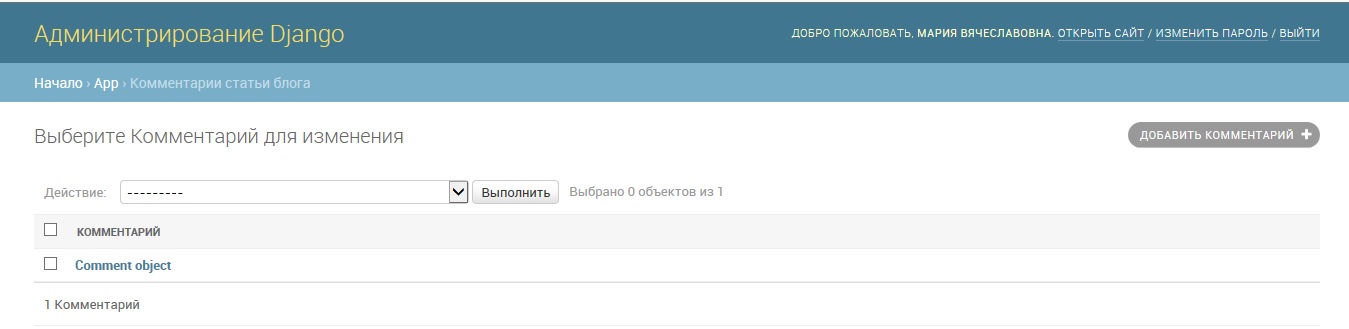
*Рисунок 23 – Удаление выбранной статьи блога*

**

*Рисунок 24 – Подтверждение удаления выбранной статьи блога*

*4) Администратор может добавлять комментарии в административном разделе сайта аналогично добавлению статьи блога. Для этого, рядом со ссылкой на подраздел «Комментарии статьи блога» нажать кнопку «+Добавить». Затем заполнить форму добавления комментария и нажать кнопку «Сохранить».*

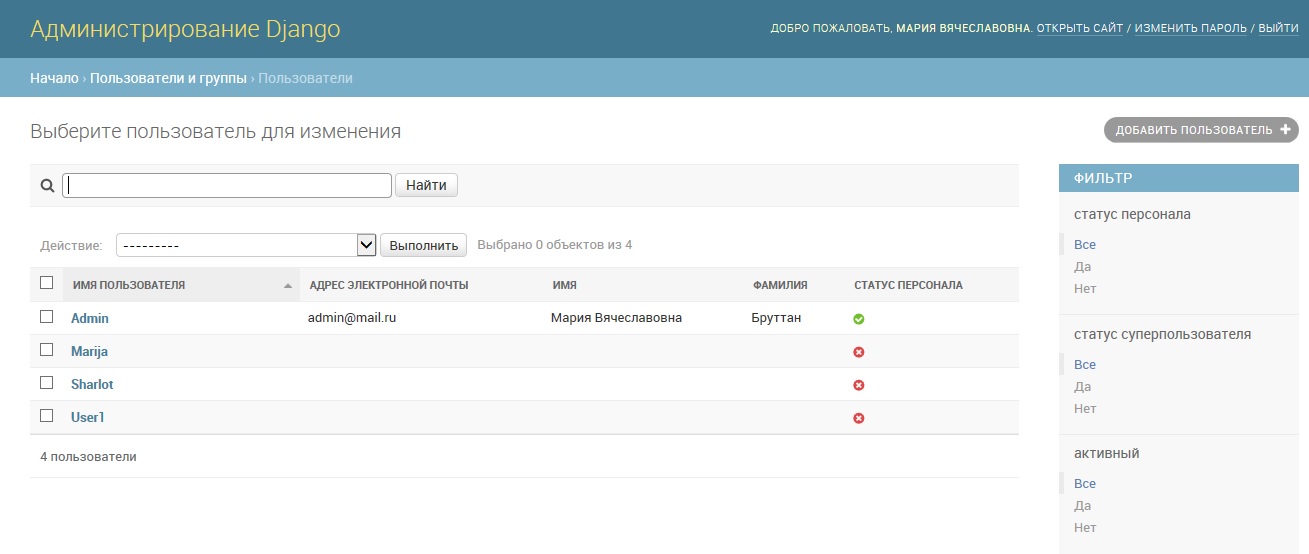
*5) Изменение комментариев (с целью исправления орфографических, синтаксических ошибок) реализовано аналогично изменению статей блога (см. пункт 2 данного раздела отчета). Необходимо перейти по ссылке названия комментария в подразделе «Комментарии статьи блога» (рисунок 25), изменить данные в форме и нажать кнопку «Сохранить».*

**

*Рисунок 25 – Комментарии статьи блога*

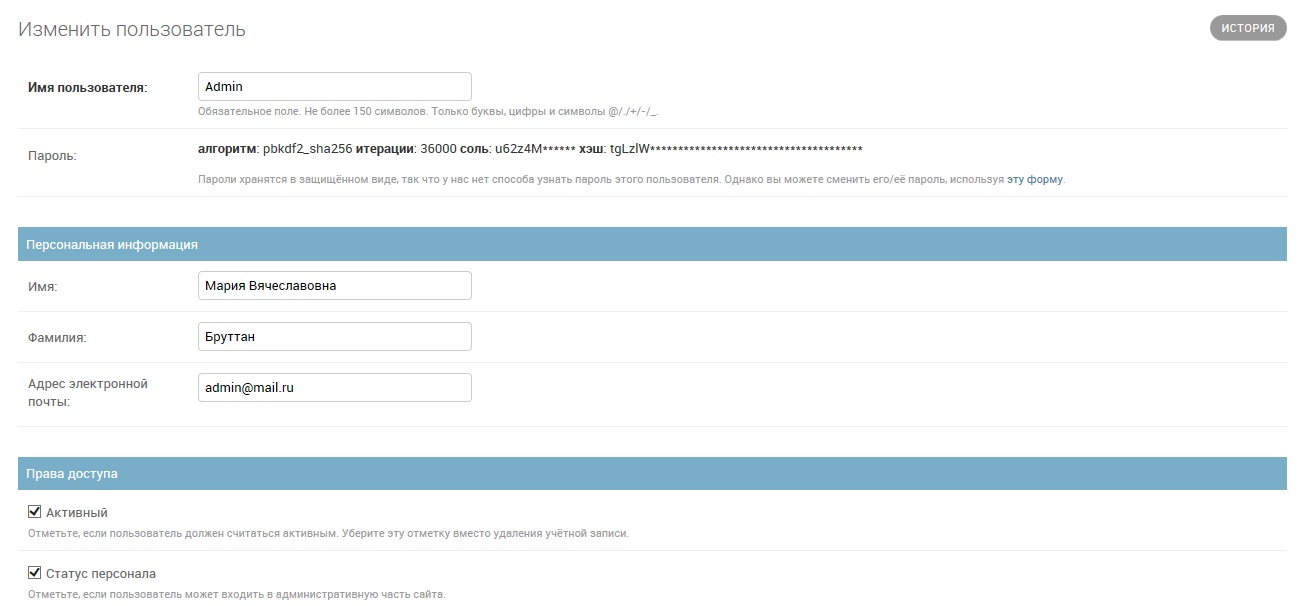
*6) Удаление комментариев к статье блога реализовано аналогично удалению статей (см. пункт 3 данного раздела отчета). Для этого необходимо перейти в подраздел «Комментарии статьи блога» (рисунок 25).*

*7) Удаление пользователей реализовано аналогично удалению статей и комментариев. Для этого необходимо перейти в подраздел «Пользователи» (рисунок 26), выбрать удаляемого пользователя – установить галочку в элементе управления CheckBox и нажать кнопку «Удалить». Затем подтвердить удаление.*

**

*Рисунок 26 – Список пользователей*

*8) Администратор может изменить свои данные. Для этого в подразделе «Пользователи» необходимо перейти по ссылке Admin. Изменить данные в форме (рисунок 27) и нажать кнопку «Сохранить».*

**

*Рисунок 27 – Изменение данных администратора*

*9) После завершения всех изменений данных в административном разделе необходимо нажать левой клавишей мыши по ссылке «Выйти», которая позволяет выйти из административного раздела веб-сайта.*