Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»



**Отчет**

**Лабораторная работа № 4**

**По курсу «Разработка интернет приложений»**

**ИСПОЛНИТЕЛЬ:**

Группа ИУ5Ц-72Б

Пылаев Б.А.

**ПРЕПОДАВАТЕЛЬ:**

Гапанюк Ю.Е.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_"\_\_\_\_\_\_\_\_\_\_\_2021 г.

Москва 2021

1. **Общее задание**
2. Создайте прототип веб-приложения с использованием фреймворка Django:
   * Создайте виртуальное окружение.
   * Установите в него Django.
   * Создайте проект и приложение Django.
3. Создайте представления и шаблоны (по желанию можно использовать модели), реализующие концепцию master/detail со следующей функциональностью:
   * На странице master в виде списка HTML выводится информация о трех объектах (например, о трех сортах мороженого). Каждая строка списка представляет собой гиперссылку, при нажатии на которую происходит переход к странице detail.
   * Страница detail содержит детальное описание объекта (сорта мороженого), фотографию, гиперссылку на master-страницу.
   * Фотография относится к статическому содержимому сайта.
   * Страница detail должна выводить данные с использованием таблицы HTML.
   * Шаблон страницы detail получает от представления данные о детальном объекте с использованием контекста.
   * НЕОБЯЗАТЕЛЬНЫЙ ПУНКТ. По желанию можно использовать верстку с применением Bootstrap (или аналогичного фреймворка), а также представления на основе классов (class-based views).
4. **Текст программы**

manage.py

#!/usr/bin/env python

"""Django's command-line utility for administrative tasks."""

import os

import sys

def main():

    """Run administrative tasks."""

    os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'LAB\_4.settings')

    try:

        from django.core.management import execute\_from\_command\_line

    except ImportError as exc:

        raise ImportError(

            "Couldn't import Django. Are you sure it's installed and "

            "available on your PYTHONPATH environment variable? Did you "

            "forget to activate a virtual environment?"

        ) from exc

    execute\_from\_command\_line(sys.argv)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

asgi.py

For more information on this file, see

https://docs.djangoproject.com/en/3.2/howto/deployment/asgi/

"""

import os

from django.core.asgi import get\_asgi\_application

os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'LAB\_4.settings')

application = get\_asgi\_application()

settings.py

"""

Django settings for LAB\_4 project.

Generated by 'django-admin startproject' using Django 3.2.8.

For more information on this file, see

https://docs.djangoproject.com/en/3.2/topics/settings/

For the full list of settings and their values, see

https://docs.djangoproject.com/en/3.2/ref/settings/

"""

from pathlib import Path

import os

# Build paths inside the project like this: BASE\_DIR / 'subdir'.

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'django-insecure-jf4pbm$v0bz&\_!un9j4a3zvzgplu%6845txtdvhtq9^av4y6e5'

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'master.apps.LabAppConfig',

]

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'django.middleware.common.CommonMiddleware',

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'LAB\_4.urls'

TEMPLATES = [

    {

        'BACKEND': 'django.template.backends.django.DjangoTemplates',

        'DIRS': [BASE\_DIR / 'templates']

        ,

        'APP\_DIRS': True,

        'OPTIONS': {

            'context\_processors': [

                'django.template.context\_processors.debug',

                'django.template.context\_processors.request',

                'django.contrib.auth.context\_processors.auth',

                'django.contrib.messages.context\_processors.messages',

            ],

        },

    },

]

WSGI\_APPLICATION = 'LAB\_4.wsgi.application'

# Database

# https://docs.djangoproject.com/en/3.2/ref/settings/#databases

DATABASES = {

    'default': {

        'ENGINE': 'django.db.backends.sqlite3',

        'NAME': BASE\_DIR / 'db.sqlite3',

    }

}

# Password validation

# https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators

AUTH\_PASSWORD\_VALIDATORS = [

    {

        'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

    },

]

# Internationalization

# https://docs.djangoproject.com/en/3.2/topics/i18n/

LANGUAGE\_CODE = 'ru-ru'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC\_URL = '/static/'

STATICFILES\_DIRS = [

   os.path.join(BASE\_DIR, "static"),

]

# Default primary key field type

# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

urls.py

"""LAB\_4 URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:

    https://docs.djangoproject.com/en/3.2/topics/http/urls/

Examples:

Function views

    1. Add an import:  from my\_app import views

    2. Add a URL to urlpatterns:  path('', views.home, name='home')

Class-based views

    1. Add an import:  from other\_app.views import Home

    2. Add a URL to urlpatterns:  path('', Home.as\_view(), name='home')

Including another URLconf

    1. Import the include() function: from django.urls import include, path

    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))

"""

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('master/', include('master.urls')),

    path('admin/', admin.site.urls)

]

wsgi.py

"""

WSGI config for LAB\_4 project.

It exposes the WSGI callable as a module-level variable named ``application``.

For more information on this file, see

https://docs.djangoproject.com/en/3.2/howto/deployment/wsgi/

"""

import os

from django.core.wsgi import get\_wsgi\_application

os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'LAB\_4.settings')

application = get\_wsgi\_application()

admin.py

from django.contrib import admin

from .models import Animal

# Register your models here.

admin.site.register(Animal)

apps.py

from django.apps import AppConfig

class LabAppConfig(AppConfig):

    default\_auto\_field = 'django.db.models.BigAutoField'

    name = 'master'

    verbose\_name = 'ЛР-4'

models.py

from django.db import models

# Create your models here.

class Animal(models.Model):

    animal\_type = models.CharField('Вид животного', max\_length=50)

    animal\_name = models.CharField('Кличка животного', max\_length=50)

    animal\_photo = models.URLField('Фото жифотного', max\_length=200)

    def \_\_str\_\_(self):

        return f'Тип: {self.animal\_type}; Имя: {self.animal\_name}; URL фото: {self.animal\_photo}'

    class Meta:

        verbose\_name = 'Животное'

        verbose\_name\_plural = 'Животные'

tests.py

from django.test import TestCase

# Create your tests here.

urls.py

from django.urls import path

from . import views

app\_name = 'master'

urlpatterns = [

    path('', views.index, name='index'),

    path('<int:animal\_id>', views.detail, name='detail')

]

views.py

from django.http import Http404, HttpResponseRedirect

from django.shortcuts import render

from .models import Animal

def index(request):

    a\_list = Animal.objects.all()

    return render(request, 'master/list.html', dict([('a\_list', a\_list)]))

def detail(request, animal\_id):

    try:

        a = Animal.objects.get(id=animal\_id)

    except:

        raise Http404('Животное не найдено!')

    return render(request, 'master/detail.html', dict([('animal', a)]))

1. **Экранные формы с примерами выполнения программы**























