Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Московский государственный технический
университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Курс «Разработка интернет-приложений»

Отчет по лабораторной работе №8

Выполнил:
студент группы РТ5-51Б
Коваль И.А.
Преподаватель:
Гапанюк Ю.Е

2020

Описание задания:

1. Создайте прототип веб-приложения с использованием фреймворка Django:
   - Создайте виртуальное окружение.
   - Установите в него Django.
   - Создайте проект и приложение Django.
2. 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).

Текст программы:

Manage.py

```
import os
import sys


def main():
 """Run administrative tasks."""
 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'mysite.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()
```

Settings.py

```
from pathlib import Path

# 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/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 's#z7y!rp)s7y=%7-e=l&rkk%8qo5ck-q6wu=2m#x^ju9zc-r$4'

# 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',
 'shop'
]

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 = 'mysite.urls'

TEMPLATES = [
 {
 'BACKEND': 'django.template.backends.django.DjangoTemplates',
'DIRS': [],
 '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 = 'mysite.wsgi.application'


# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases

DATABASES = {
 'default': {
 'ENGINE': 'django.db.backends.sqlite3',
 'NAME': BASE_DIR / 'db.sqlite3',
 }
}


# Password validation
# https://docs.djangoproject.com/en/3.1/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/3.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
```

```
# https://docs.djangoproject.com/en/3.1/howto/static-files/

STATIC_URL = '/static/'
```

Urls.py

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
 path('admin/', admin.site.urls),
 path('', include('shop.urls')),
]
```

Admin.py

```
from django.contrib import admin
from .models import Cinema, Description


admin.site.register(Cinema)
admin.site.register(Description)
```

Apps.py

```
from django.apps import AppConfig


class ShopConfig(AppConfig):
 name = 'shop'
```

Urls.py

```
from django.urls import path
from . import views

app_name = 'shop'
urlpatterns = [
 path('', views.index, name='home'),
 path('<int:cinema_id>/', views.detail, name='detail'), ]
```

Views.py

```
from django.shortcuts import get_object_or_404, render

from .models import Cinema
from .models import Description


def index(request):
 cinema_list = Cinema.objects.all()
 context = {'cinema_list': cinema_list}
 return render(request, 'shop/index.html', context)


def detail(request, cinema_id):
 cinema_list = Cinema.objects.all()
 for i in cinema_list:
 if i.id == cinema_id:
 cinema = i
```

```
 description = get object or 404(Description, pk=cinema_id)  return
render(request, 'shop/detail.html', {'cinema': cinema,
'description': description})
```

Models.py

```
from django.db import models


class Cinema(models.Model):
 name = models.CharField(max_length=20)


class Description(models.Model):
 cinema id = models.ForeignKey(Cinema, on delete=models.CASCADE)
description = models.CharField(max_length=1500)
```

Экранные формы с примерами выполнения программы