Міністерство освіти і науки України

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

ЗВІТ

З лабораторної роботи №4

з дисципліни «Проєктування та моделювання програмного забезпечення вбудованих систем» на тему:

«ПРИНЦИПИ ТЕСТУВАННЯ У DJANGO»

Виконала:

ст. гр. КНТ-119 м М.В. Саман

Прийняв:

професор, к.т.н. Г. В. Табунщик

2019

**Мета роботи:**

Ознайомитися із принципами тестування у Djangо, навчитися створювати власні тести та аналізувати результати.

**Завдання на роботу:**

Протестувати роботу веб-застосування та виявити чи має застосування порушення у функціональності або містить помилки, а також виявити та виправити помилки, якщо вони були допущені при написанні тестів.

**Виконання завдання.**

Під час створення застосування, важливим етапом є його тестування, що дозволяє впевнитися в правильності та коректності його роботи. Юніт-тести – це невеликі функції, що дозволяють тестувати окремі частини створеного застосування. Під час написання тестів було виявлено помилки (рис. 1) та усунено їх. Тестування веб-застосування завершено успішно (рис. 2). Код тестів наведено нижче.

Код файлу «tests.py»

from django.urls import reverse\_lazy

from django.test import TestCase

from django.http import HttpResponseNotAllowed

from .models import Measurement

class TestCalls(TestCase):

def setUp(self):

self.create\_url = 'create\_measurement'

self.list\_url = 'measurement\_list'

def test\_call\_view\_loads(self):

for url in [self.create\_url, self.list\_url]:

path = str(reverse\_lazy(url))

response = self.client.get(path)

self.assertEqual(response.status\_code, 200)

self.assertTemplateUsed(response, '{}.html'.format(url))

def test\_call\_view\_fails\_blank(self):

path = str(reverse\_lazy(self.create\_url))

response = self.client.post(path, {})

self.assertFormError(response, 'form', 'value', u'This field is required.')

def test\_call\_view\_fails\_incorrect(self):

path = str(reverse\_lazy(self.create\_url))

value = '1'\*51

measurement\_params = {'value': value,}

response = self.client.post(path, measurement\_params)

self.assertFormError(response, 'form', 'value', u'Ensure this value has at most 50 characters (it has 51).')

def test\_call\_view\_succeed(self):

path = str(reverse\_lazy(self.create\_url))

value = '1'\*10

description = 'measurement description'

measurement\_params = {'value': value, 'description': description}

response = self.client.post(path, measurement\_params)

self.assertEqual(response.status\_code, 302)

measurement = Measurement.objects.get(id=1)

for field in measurement\_params.keys():

measurement\_field = getattr(measurement, field)

self.assertEqual(measurement\_field, measurement\_params[field])

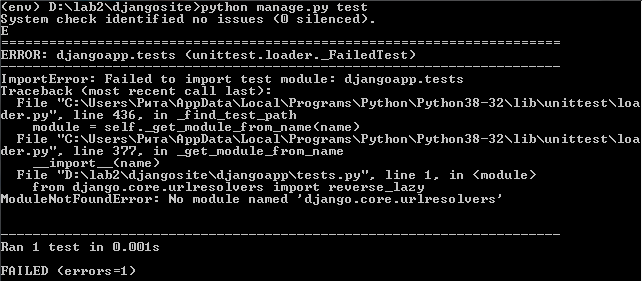


Рисунок 1 – Помилки, що допущені при написанні тестів

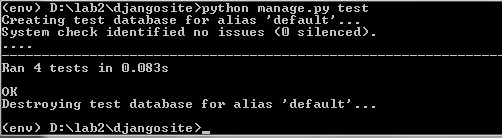


Рисунок 2 – Успішне тестування веб-застосування

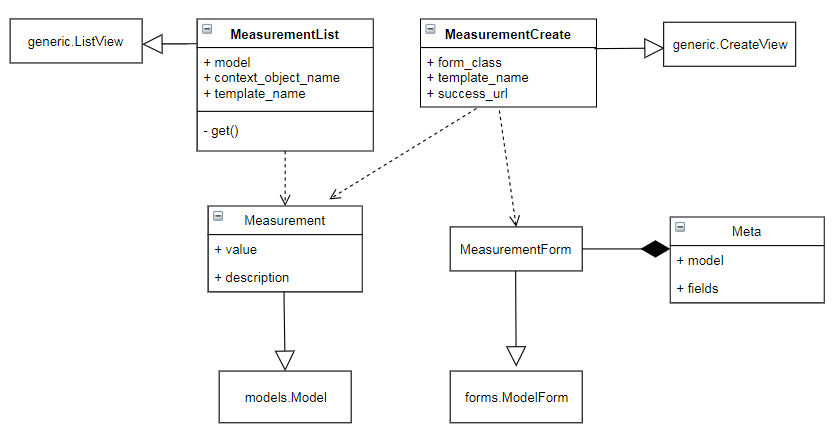


Рисунок 2 – Діаграма класів

**Висновок:** під час виконання лабораторної роботи я ознайомилася принципами застосування в Django. Виконано тестування створеного веб-застосування.