Laboratorium 1

1. Pobranie Django ze strony https://www.djangoproject.com/download/

2. Instalacja Django
   python setup.py install - -user

3. Utworzenie katalogu na projekt

4. Eksport ścieżek
   export PATH=”$PATH:~/.local/bin”

5. Generowanie projektu
   django-admin.py startproject nazwa_projektu

6. Uruchomienie serwera
   python manage.py runserver

7. Generowanie aplikacji
   python manage.py startapp nazwa_aplikacji

8. Tworzenie widoku

   nazwa_aplikacji/views.py

```
from django.http import HttpResponse

def index(request):
    return HttpResponse("Hello, world. You're at the polls index.")
```

nazwa_aplikacji/urls.py

```
from django.conf.urls import url

from . import views

urlpatterns = [
    url(r'^$', views.index, name='index'),
]
```

nazwa_projektu/urls.py

```
from django.conf.urls import include, url

from django.contrib import admin

urlpatterns = [
    url(r'^polls/', include('polls.urls')),
    url(r'^admin/', admin.site.urls),
```

```
]
```

9. Uruchomienie serwera
10. Synchronizacja bazy danych

    python manage.py migrate

11. nazwa_aplikacji/models.py

```
from django.db import models

class Question(models.Model):

    question_text = models.CharField(max_length=200)

    pub_date = models.DateTimeField('date published')

class Choice(models.Model):

    question = models.ForeignKey(Question, on_delete=models.CASCADE)

    choice_text = models.CharField(max_length=200)

    votes = models.IntegerField(default=0)
```

12. **nazwa_projektu/settings.py**

```
INSTALLED_APPS = [

    'nazwa_aplikacji.apps.PollsConfig',

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

]
```

13. **python manage.py makemigrations nazwa_aplikacji**
14. **python manage.py sqlmigrate nazwa_aplikacji 0001**
15. **python manage.py migrate**
16. Wbudowany **shell**
    Interpreter Pythona posiada także tryb interaktywny, w którym wyrażenia można wprowadzać z terminala, otrzymując natychmiast wyniki. Zgodnie z założeniem twórców Pythona ułatwiać ma to naukę programowania, gdyż pozwala wypróbowywać fragmenty kodu ze skutkiem natychmiastowym.

python manage.py shell

```
>>> from nazwa_aplikacji.models import Question, Choice
```

- **Sprawdzenie co jest w systemie**

```
>>> Question.objects.all()
<QuerySet []>
```

- **Tworzenie nowego pola**

```
>>> from django.utils import timezone
>>> q = Question(question_text="What's new?", pub_date=timezone.now())
```

- **Zapisywanie obiektu w bazie. Wywołanie metody save() jawnie.**

```
>>> q.save()
```

- **Nadanie obiektowi identyfikatora**

```
>>> q.id
1
```

- **Model dostępu wartości pól za pośrednictwem atrybutów Pythona.**

```
>>> q.question_text
"What's new?"
>>> q.pub_date
datetime.datetime(2012, 2, 26, 13, 0, 0, 775217, tzinfo=<UTC>)
```

- **Zmiana wartości poprzez zmianę atrybutów, a następnie wywołanie metody save().**

```
>>> q.question_text = "What's up?"
>>> q.save()
```

- **objects.all () wyświetla wszystkie zapytania w bazie danych.**

```
>>> Question.objects.all()
<QuerySet [<Question: Question object>]>
```

- **Dodanie metod do klas Poll I Choise:**

nazwa_aplikacji/models.py

```
from django.db import models
```

```
from django.utils.encoding import python_2_unicode_compatible

class Question(models.Model):

    # ...

    def __str__(self):

        return self.question_text


class Choice(models.Model):

    # ...

    def __str__(self):

        return self.choice_text
```

nazwa_aplikacji/models.py

```
import datetime

from django.db import models

from django.utils import timezone

class Question(models.Model):

    # ...

    def was_published_recently(self):

        return self.pub_date >= timezone.now() -
datetime.timedelta(days=1)
```

ponowne uruchomienie shell'a

- **Sprawdzenie, czy __str __() dodał się prawidłowo.**

```
>>> Question.objects.all()

<QuerySet [<Question: What's up?>]>
```

- **Django zapewnia bogate bazy danych, wyszukiwanie API, które jest całkowicie prowadzone przez argumenty kluczowe.**

```
>>> Question.objects.filter(id=1)

<QuerySet [<Question: What's up?>]>

>>> Question.objects.filter(question_text__startswith='What')

<QuerySet [<Question: What's up?>]>
```

- **Sprawdzenie zapytanie, które zostały opublikowane w tym roku.**

```
>>> from django.utils import timezone
>>> current_year = timezone.now().year
>>> Question.objects.get(pub_date__year=current_year)
<Question: What's up?>
```

- **Poproś o identyfikator, który nie istnieje.**

```
>>> Question.objects.get(id=2)
```

- **Skrót do klucza podstawowego dokładnego wyszukiwania**

```
>>> Question.objects.get(pk=1)
<Question: What's up?>
```

- **Upewnij się, że metoda niestandardowa działa.**

```
>>> q = Question.objects.get(pk=1)
>>> q.was_published_recently()
```

- **Wyszukiwanie danych**

```
>>> q = Question.objects.get(pk=1)
```

- **Wyświetlenie możliwości wyboru z odpowiedniego zestawu obiektów.**

```
>>> q.choice_set.all()
<QuerySet []>
```

- **Tworzenie trzech możliwości.**

```
>>> q.choice_set.create(choice_text='Not much', votes=0)
<Choice: Not much>
>>> q.choice_set.create(choice_text='The sky', votes=0)
<Choice: The sky>
>>> c = q.choice_set.create(choice_text='Just hacking again', votes=0)
```

- **Obiekty Choice mają API dostępu do obiektów Question.**

```
>>> c.question
<Question: What's up?>
```

- **I odwrotnie.**

```
>>> q.choice_set.all()

<QuerySet [<Choice: Not much>, <Choice: The sky>, <Choice: Just hacking
again>]>

>>> q.choice_set.count()

3
```

- **Wyszukiwanie wszystkich Choice odnośnie pytania - pub_date is in this year**

```
>>> Choice.objects.filter(question__pub_date__year=current_year)

<QuerySet [<Choice: Not much>, <Choice: The sky>, <Choice: Just hacking
again>]>
```

- **Zastosowanie funkcji delate**

```
>>> c = q.choice_set.filter(choice_text__startswith='Just hacking')

>>> c.delete()
```

17. Aktywacja panelu administracyjnego

    python manage.py createsuperuser

    Username:

    Password:

    Email:

18. Uruchomienie ponownie serwera

19. Modyfikacja panelu administracyjnego

    nazwa_aplikacji/admin.py

```
from django.contrib import admin

from .models import Question

admin.site.register(Question)
```