ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Ордена Трудового Красного Знамени

федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский Технический Университет Связи и Информатики» (МТУСИ)

Лабораторная работа №3

Выполнил:

Студент 2 курса группы БСТ1703

Кузьмин В.А.

**Москва 2019**

Создаем проект с названием blog и в нем создаем приложение с названием article. После этого в файле article/models создаем модель которая будет использоваться для хранения наших значений в БД.

**from** django.db **import** models  
  
*# Create your models here.***from** django.contrib.auth.models **import** User  
**class** Article(models.Model):  
 **class** Meta:  
 db\_table = **"articles\_db"** title = models.CharField(max\_length=200)  
 author = models.ForeignKey(User, on\_delete = models.DO\_NOTHING)  
 text = models.TextField()  
 created\_date = models.DateField(auto\_now\_add=True)  
   
 **def** \_\_unicode\_\_(self):  
 **return "%s: %s"** % (self.author.username, self.title)  
  
 **def** get\_excerpt(self):  
 **return** self.text[:140] + **"..." if** len(self.text) > 140 **else** self.text

Класс article Описывает типы данных, которые будут сохранятся в БД, а так-же функцию get\_excerpt для отображения не полной записи а ее части и функции \_\_unicode\_\_ для более понятного описания статей.

Далее регистрируем эту модель в файле admin.py

**from** django.contrib **import** admin  
**from** .models **import** Article  
*# Register your models here.***class** ArticleAdmin(admin.ModelAdmin):  
 list\_display = (**'title'**, **'author'**, **'get\_excerpt'**, **'created\_date'**)  
  
  
admin.site.register(Article, ArticleAdmin)

После этого запускаем сервер и переходим на страицу статей и создаем несколько, после чего с помощью SQLite редактируем некоторые записи и добавляем еще несколько(в данном примере все данные заносимые через интерфес были занесены на англиском языке, а все изменения и добавления посредством sql были сделаны на русском)

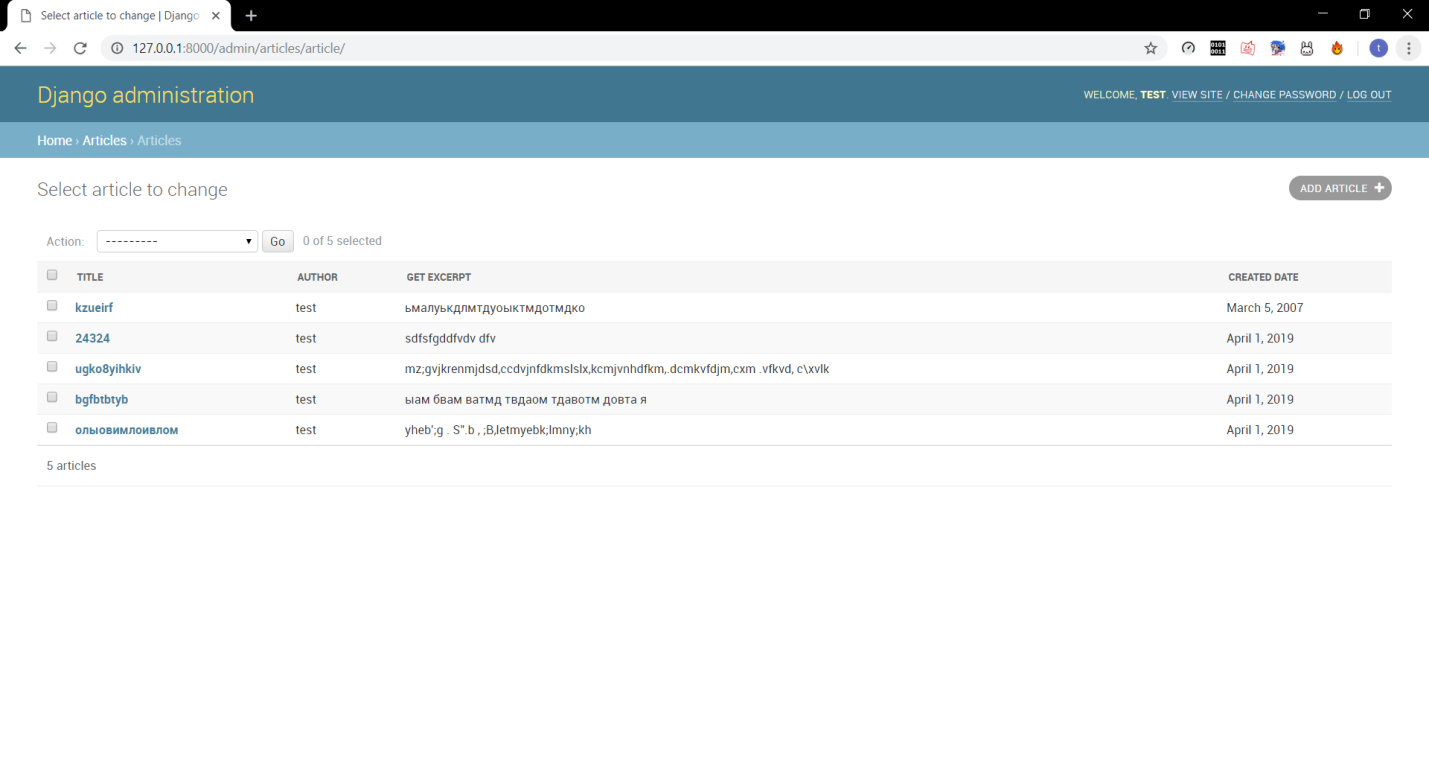


Рисунок – Список статей

Далее добавим страницу с отображением всех статей, для этого создадим html файл со следующим содержанием

<!DOCTYPE **html**>  
<**html lang="en"**>  
<**head**>  
 <**meta charset="UTF-8"**>  
 <**title**>Архив статей</**title**>  
</**head**>  
<**body**>  
<**div class="header"**>  
</**div**>  
<**div class="archive"**>  
 {% for post in posts %}  
 <**div class="one-post"**>  
 <**a class="post-title", href="{% url 'article:get\_article' post.id"**>{{ post.title }}</**a**>  
 <**div class="article-info"**>  
 <**div class="article-author"**>{{ post.author.username }}</**div**>  
 <**div class="article-created-date"**>{{ post.created\_date }}</**div**>  
 </**div**>  
 <**p class="article-text"**>{{ post.get\_excerpt }}</**p**>  
 </**div**>  
 {% endfor %}  
</**div**>  
</**body**>  
</**html**>

Данная страница итеративно отображает все статьи занесенные в БД, далее добавляем url и view для этой страницы в article и открываем эту страницу в браузере.

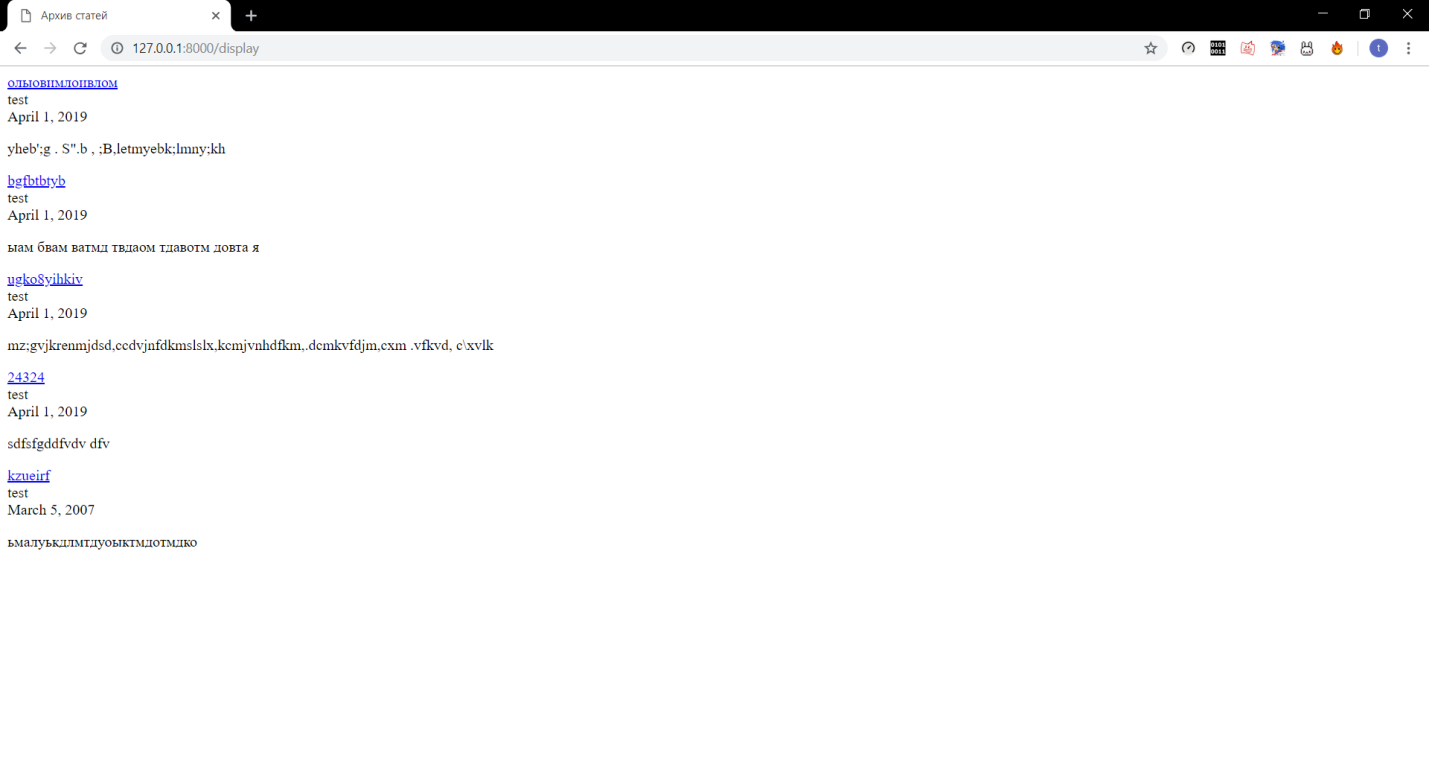


Рисунок – вывод всех статей