1. Для комментариев пользователей **добавим новую модель Message в riddles/models.py**:

*# встроенная модель пользователя  
# нужна для авторов сообщений***from** django.contrib.auth.models **import** User  
*# тип "временнАя зона" для получения текущего времени***from** django.utils **import** timezone  
...  
**class** Message(models.Model):  
 chat = models.ForeignKey(  
 Riddle,  
 verbose\_name=**'Чат под загадкой'**,  
 on\_delete=models.CASCADE)  
 author = models.ForeignKey(  
 User,  
 verbose\_name=**'Пользователь'**)  
 message = models.TextField(**'Сообщение'**)  
 pub\_date = models.DateTimeField(  
 **'Дата сообщения'**,  
 default=timezone.now)

1. Создадим миграцию – код на языке SQL для создания в БД таблицы, соответствующей нашей модели – напишем **в терминале**:

**python manage.py makemigrations riddles**

1. Выполним сгенерированный код на языке SQL – напишем **в терминале**:

**python manage.py migrate**

1. Дадим администраторам возможность подчищать созданную в БД таблицу – **допишем в существующий файл riddles/admin.py:**

**from** .models **import** Message  
...  
**admin.site.register(Message)**

1. Для добавления объектов созданного класса в БД **допишем новую функцию в существующем файле riddles/views.py:**

*# сообщения***from** .models **import** Message  
**from** datetime **import** datetime

**...**

**def** post(request, riddle\_id):  
 msg = Message()  
 msg.author = request.user  
 msg.chat = get\_object\_or\_404(Riddle, pk=riddle\_id)  
 msg.message = request.POST[**'message'**]  
 msg.pub\_date = datetime.now()  
 msg.save()  
 **return** HttpResponseRedirect(app\_url+str(riddle\_id))

1. **Привяжем созданную функцию к соответствующему URL, на который будем присылать сообщения, в riddles/urls.py:**

urlpatterns = [  
 url(**r'^$'**, views.index, name=**'index'**),  
 url(**r'^([0-9]+)/$'**, views.detail, name=**'detail'**),  
 url(**r'^([0-9]+)/answer/$'**, views.answer, name=**'answer'**),

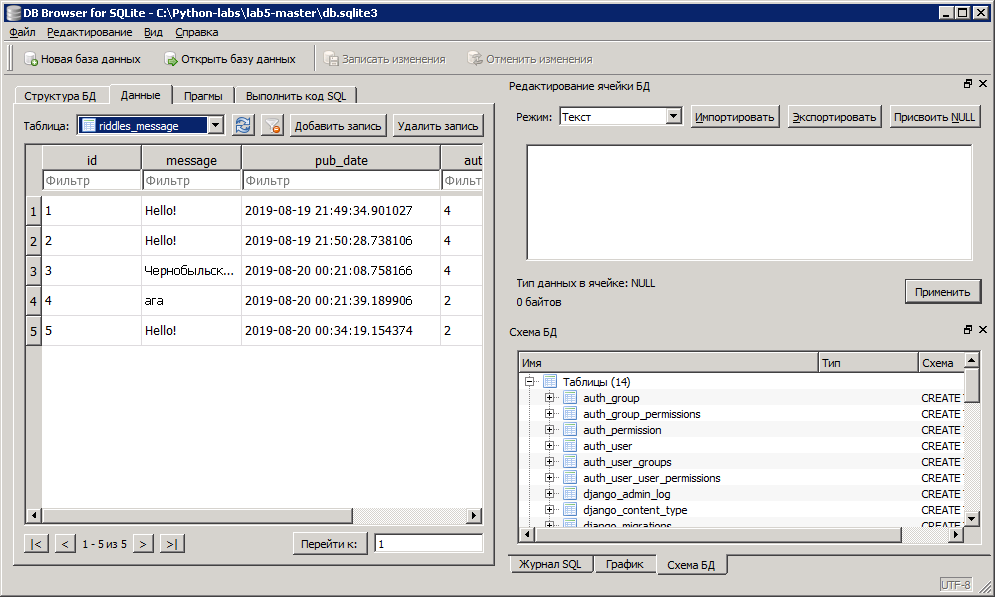
...  
 ***# отправка сообщения* url(r'^([0-9]+)/post/$', views.post, name='post'),**  
]

1. Чтобы авторизованные пользователи могли вводить сообщения под загадками, **допишем новый фрагмент в шаблон** **riddles/templates/answer.html (строчка {% endblock %} не копируется, это старая все время спускается в самый низ):**

<**hr**>  
<**h4**>Chat</**h4**>  
{% **if user**.**is\_authenticated** %}  
 <**form action="post/" method="post"**>  
 {% **csrf\_token** %}  
 <**div class="input-group mb-3"**>  
 <**input type="text"  
 class="form-control"  
 placeholder="Your message"  
 name="message"**/>  
 <**div class="input-group-append"**>  
 <**input class="btn btn-success"  
 type="submit"  
 value="Send"**/>  
 </**div**>  
 </**div**>  
</**form**>  
{% **endif** %}

{% **endblock** %}

1. **Запустим приложение, авторизуемся, зайдем на страницу загадки, отправим сообщение, откроем БД в DB Browser, зайдем на вкладку данные, выберем таблицу riddles\_message, убедимся, что наши сообщения находятся в БД:**



1. **Через панель администратора добавьте новую загадку и комментарий к ней.**
2. Для выбора из таблицы БД сообщений, соответствующих определенной загадке, **модифицируем функцию detail в файле riddles/views.py:**

**def** detail(request, riddle\_id):  
 error\_message = **None  
 if "error\_message" in** request.GET:  
 error\_message = request.GET[**"error\_message"**]

**return** render(  
 request,  
 **"answer.html"**,  
 {  
 **"riddle"**: get\_object\_or\_404(  
 Riddle, pk=riddle\_id),  
 **"error\_message"**: error\_message,  
 **"latest\_messages":  
 Message.objects  
 .filter(chat\_id=riddle\_id)  
 .order\_by('-pub\_date')[:5]**  
 }  
 )

1. Для вывода отобранных сообщений **допишем шаблон riddles/templates/answer.py (строчка {% endblock %} не копируется, это старая все время спускается в самый низ):**

<**div** **id="history"**>  
 {% **for message in latest\_messages** %}  
 <**div class="card"**>  
 <**div class="card-body"**>  
 <**strong**>  
 {{ **message**.**author**.**username** }}  
 </**strong**>  
 <**br**/>  
 <**em**>  
 {{ **message**.**pub\_date**|**date**:**'Y-m-d H:i:s'** }}  
 </**em**>  
 <**br**/>  
 {{ **message**.**message** }}  
 </**div**>  
 </**div**>  
 <**br**/>  
 {% **endfor** %}  
</**div**>  
{% **endblock** %}

1. **Запускаем проект, проверяем отображение отправленных ранее сообщений, добавляем парочку новых от имени различных пользователей.**
2. Плохо, что список сообщений обновляется только при открытии страницы с загадкой и при отправке сообщения. Будем делать обновление списка сообщений по таймеру каждые 5 сек. Но для начала нужно где-то брать сообщения для текущей загадки, поэтому **допишем функцию, возвращающую список сообщений к загадке в формате JSON, в файл riddles/views.py:**

*# для ответа на асинхронный запрос в формате JSON***from** django.http **import** JsonResponse   
**import** json

**...**

**def** msg\_list(request, riddle\_id):  
 *# выбираем список сообщений* res = list(  
 Message.objects  
 *# фильтруем по id загадки* .filter(chat\_id=riddle\_id)  
 *# отбираем 5 самых свежих* .order\_by(**'-pub\_date'**)[:5]

*# выбираем необходимые поля* .values(**'author\_\_username'**,  
 **'pub\_date'**,  
 **'message'** )  
 )  
 *# конвертируем даты в строки - сами они не умеют* **for** r **in** res:  
 r[**'pub\_date'**] = \  
 r[**'pub\_date'**].strftime(  
 **'%d.%m.%Y %H:%M:%S'** )  
 **return** JsonResponse(json.dumps(res), safe=**False**)

1. **Допишем URL, запросы к которому будут обрабатываться этой функцией, в urls.py:**

urlpatterns = [  
 url(**r'^$'**, views.index, name=**'index'**),  
 url(**r'^([0-9]+)/$'**, views.detail, name=**'detail'**),  
 url(**r'^([0-9]+)/answer/$'**, views.answer, name=**'answer'**),  
 *...*  
 *# отправка сообщения* url(**r'^([0-9]+)/post/$'**, views.post, name=**'post'**),  
 *# отправка списка сообщений***url(r'^([0-9]+)/msg\_list/$',  
 views.msg\_list,  
 name='msg\_list'),**  
]

1. Допишем подключение библиотеки jQuery, отправку асинхронных запросов каждые 5 сек и вывод ответов на них на страницу в шаблон **riddles/templates/answer.py (строчка {% endblock %} не копируется, это старая все время спускается в самый низ):**

<**script src="https://code.jquery.com/jquery-3.4.1.min.js"**>  
</**script**>  
<**script**>  
**function** *send\_msg\_update\_request*(){  
 **$**.get(**"msg\_list/"**,{},*parse\_msg\_update\_response*);  
}  
**function** *parse\_msg\_update\_response*(data, success){  
 *//alert(data);* **res** = ***JSON***.parse(data);  
 **res\_str** = **""**;  
 **for**(**i**=0;**i**<**res**.**length**;**i**++){

*// HTML-код вставляемых блоков с сообщениями* *// скопирован выше - тот, что выводится* *// при загрузке страницы*

**res\_str** += **'<div class="card">\n'** +  
 **' <div class="card-body">\n'** +  
 **' <strong>\n'** +  
 **' '**+**res**[**i**].author\_\_username+**'\n'** +  
 **' </strong>\n'** +  
 **' <br/>\n'** +  
 **' <em>\n'** +  
 **' '**+**res**[**i**].pub\_date+**'\n'** +  
 **' </em>\n'** +  
 **' <br/>\n'** +  
 **' '**+**res**[**i**].**message**+**'\n'** +  
 **' </div>\n'** +  
 **' </div>\n'** +  
 **' <br/>'** }  
 **$**(**"#history"**).html(**res\_str**);  
}  
*setInterval*(*send\_msg\_update\_request*, 5000);  
</**script**>  
{% **endblock** %}

1. **Заходим в двух вкладках браузера на страницу одной и той же загадки, добавляем сообщение на одной вкладке и смотрим, появится ли на другой. Если не появляется – открываем консоль разработчика на F12 и смотрим ошибки.**