

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

Факультет программной инженерии и компьютерной техники

КУРСОВАЯ РАБОТА

Тема: Проектирование, реализация и тестирование блога о фильмах с функцией агрегации новостей из сторонних источников.

Работу выполнила Ян Цзяфэн **группы** P33212
(фамилия, имя) (номер группы)

Руководитель Штенников Дмитрий Геннадьевич
(фамилия, имя, отчество)

Работа защищена " " 2021 г.

с оценкой

Подписи членов комиссии:

СОДЕРЖАНИЕ

Введение	3
Диаграмма вариантов использования	3
Диаграммы состояний модели	9
Даталогическая модель сущностей	10
Код программы	13
Визуальный результат работы	16
Тестирование	23
Заключение	24

ВВЕДЕНИЕ

Проект представляет собой блог о фильмах, основанный на языке Python и фреймворке Django. Он использует базу данных Mysql для хранения информации, такой как пользователи, статьи и комментарии, и базу данных Redis для хранения информации о сеансах. Это веб-приложение реализует функции регистрации, входа в систему, изменения паролей, изменения личной информации пользователя, публикации блога, публикации комментариев, агрегации новостей со сторонних сайтов (<https://thefilm.blog/>), рекомендации статей и пейджинга.

Django - это фреймворк для веб-приложений с открытым исходным кодом, написанный на Python. Используя Django, вы можете легко заполнить большую часть контента, необходимого для формального веб-сайта, с помощью очень небольшого количества кода и в дальнейшем разработать полнофункциональную веб-службу. Сам Django основан на модели MVC, а именно Model + View + Controller режим разработки. Режим MVC упрощает последующую модификацию и расширение программы и дает возможность повторно использовать определенную часть программы.

MySQL - самая популярная система управления реляционными базами данных. С точки зрения веб-приложений MySQL - одна из лучших прикладных программ СУБД (система управления реляционными базами данных).

Redis является полностью открытым исходным кодом, соответствует протоколу BSD и представляет собой высокопроизводительную базу данных «ключ-значение». Redis поддерживает сохранение данных. Данные в памяти могут быть сохранены на диске и могут быть загружены снова для использования при перезапуске. Redis не только поддерживает простые данные типа «ключ-значение», но также предоставляет хранилище для списков, наборов, zset, хешей и других структур данных.

В части моделирования системы веб-приложений используется унифицированный язык моделирования UML. UML состоит из нескольких частей, таких как представления, диаграммы, элементы модели и общие механизмы. Он может объяснять, визуализировать и документировать продукты объектно-ориентированной системы разработки.

ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ

Разработка информационной системы начинается с анализа функциональных требований. В этом процессе устанавливается модель прецедентов использования, которая описывает динамическое поведение системы моделирования с точки зрения взаимодействия между внешними агентами и системой и описывает отношения между пользователями, требованиями и функциональными единицами системы. Диаграмма прецедентов использования состоит из субъектов, прецедентов использования и отношений между ними. Ниже приведена диаграмма прецедентов использования для разработки этой системы блогов о фильмах.

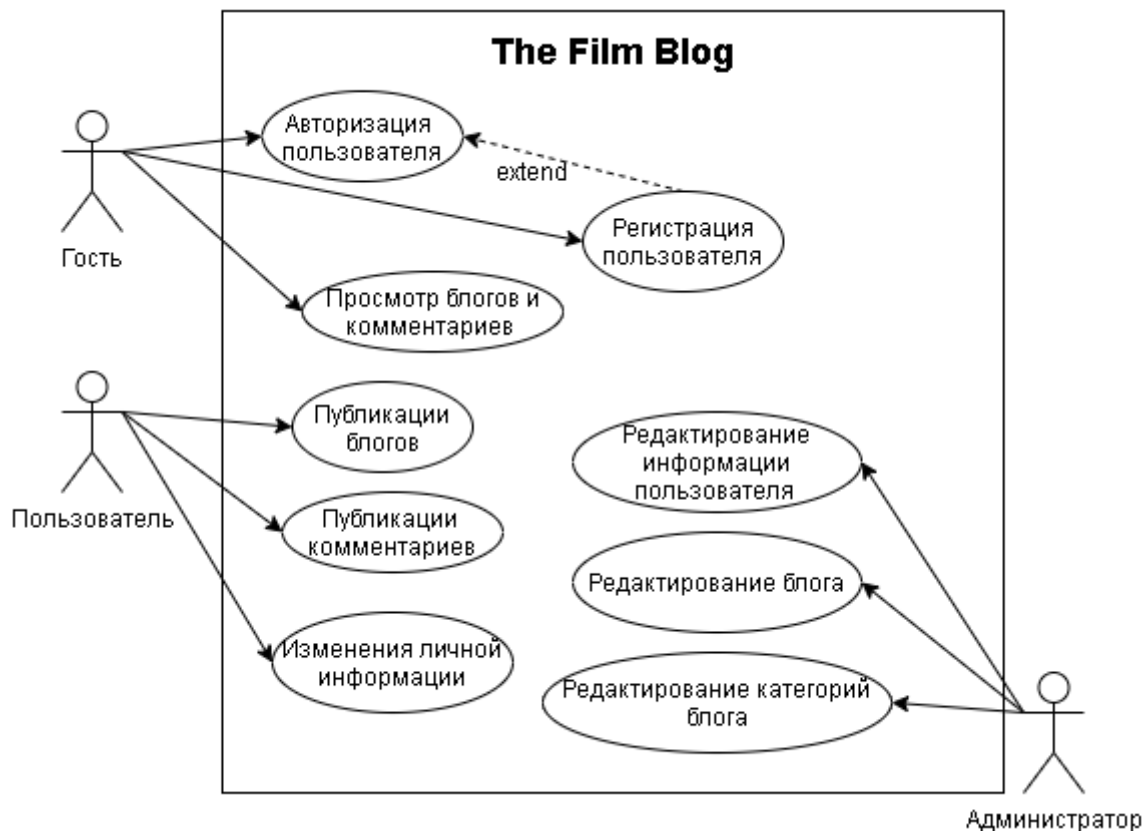


Рисунок 1.1 – Диаграмма прецедентов использования

Участники этой модели прецедентов использования делятся на гостей, авторизованных пользователей и администраторов.

- Гости - это неавторизованные пользователи, которые имеют право регистрироваться и входить в систему, а также просматривать блоги и комментарии.
- Авторизованные пользователи имеют право редактировать личную информацию, публиковать блоги и комментарии.
- Администратор является авторизованным пользователем и имеет право создавать и редактировать пользователей, редактировать категории блогов и редактировать сообщения блога.

Следует отметить, что вариант использования Регистрации расширяет вариант использования Авторизации. Основная цель варианта использования Редактирования информации пользователя и статьи - помочь администратору просматривать информацию о пользователях и статьях. Администратор обычно изменяет информацию о категории блога.

Далее в таблицах 1.1-1.4 будут описаны данные прецеденты более детально.

Прецедент использования	Регистрация пользователя	Авторизация пользователя
Краткое описание	Этот вариант использования позволяет гостю зарегистрировать новую учетную запись в системе.	Этот вариант использования позволяет гостю войти в свою учетную запись в системе.
Действующие лица	Гость	Гость
Предусловия	Действующее лицо Гость желает зарегистрировать новую учетную запись.	Действующее лицо Гость желает войти в свою учетную запись.
Основной поток	Прецедент использования начинается с того, что Гость решает зарегистрировать новую учетную запись и заполняет соответствующую регистрационную форму, обязательно указав имя учетной записи и пароль, а также код подтверждения, после чего отправляет заявку на регистрацию. Прецедент использования завершается.	Прецедент использования начинается с того, что Гость решает войти в свою учетную запись и заполнить соответствующую форму входа, убедившись, что учетная запись и пароль верны, а затем отправляет заявку на вход. Прецедент использования заканчивается.
Альтернативные потоки	Действующее лицо Гость неверно заполнило регистрационную форму или учетная запись с указанным именем уже существует. Заявка на регистрацию не создается, действующее лицо Гость получает сообщение об ошибке и должно либо повторно заполнить форму, продолжая прецедент использования, либо отменить регистрацию, завершая прецедент использования.	Действующее лицо Гость неверно заполнило свою учетную запись или пароль. Заявка на вход не создается, действующее лицо Гость получает сообщение об ошибке и должно либо повторно заполнить форму, продолжая прецедент использования, либо отменить вход, завершая прецедент использования.
Постусловия	Если прецедент использования завершился успешно, в системе регистрируется новая учетная запись, в противном случае состояние системы остается неизменным	Если прецедент использования завершился успешно, Гость войдет в систему как учетная запись, в противном случае состояние системы остается неизменным

Таблица 1.1 Описание прецедента Регистрация и Авторизация.

Прецедент использования	Просмотр блогов и комментариев	Публикации комментариев
Краткое описание	Этот вариант использования позволяет Актёру просматривать блоги и комментарии в системе.	Этот вариант использования позволяет Пользователю оставить свои комментарии в системе.
Действующие лица	Гость, Пользователь	Пользователь
Предусловия	Актер желает читать блоги или комментарии.	Действующее лицо Пользователь желает оставить свои комментарии.
Основной поток	Прецедент использования начинается с того, что Актер нажимает на одно из названий на странице. Система перенаправляет пользователя на страницу данной новости. Прецедент использования завершается.	Прецедент использования начинается с того, что Пользователь решает оставить свои комментарии и заполнить свои комментарии в области комментариев, а затем отправляет заявку. Прецедент использования заканчивается.
Постусловия	Если прецедент использования завершился успешно, Актер перенаправлен на страницу новости.	Если прецедент использования завершился успешно, обновится страница и отобразят новые комментарии

Таблица 1.2 Описание прецедента Просмотра блогов и комментариев и Публикации комментариев.

Прецедент использования	Публикации блогов	Изменения личной информации пользователя
Краткое описание	Этот вариант использования позволяет Пользователю публиковать блоги в системе.	Этот вариант использования позволяет Пользователю редактировать личную информацию в системе.
Действующие лица	Пользователь	Пользователь
Предусловия	Действующее лицо Пользователь желает публиковать блоги.	Действующее лицо Пользователь желает редактировать личную информацию.
Основной поток	Прецедент использования начинается с того, что Пользователь решает публиковать блоги и заполняет соответствующую форму, обязательно указав заголовок и заглавное изображение блога, а также контент, после чего отправляет заявку. Прецедент использования завершается.	Прецедент использования начинается с того, что Пользователь решает редактировать личную информацию и заполнить соответствующую форму, а затем отправляет заявку. Прецедент использования заканчивается.
Альтернативные потоки	Действующее лицо Пользователь неверно заполнило форму или блог с указанным именем уже существует. Заявка на публикацию блога не создается, действующее лицо Пользователь получает сообщение об ошибке и должно либо повторно заполнить форму, продолжая прецедент использования, либо отменить публикацию, завершая прецедент использования.	Действующее лицо Пользователь неверно заполнило форму. Заявка на редактирование не создается, Пользователь получает сообщение об ошибке и должно либо повторно заполнить форму, продолжая прецедент использования, либо отменить редактирование, завершая прецедент использования.
Постусловия	Если прецедент использования завершился успешно, в системе публикуется новый блог, в противном случае состояние системы остается неизменным	Если прецедент использования завершился успешно, обновится страница и отобразит новая информация, в противном случае состояние системы остается неизменным

Таблица 1.3 Описание прецедента Публикации блогов и Изменения личной информации пользователя.

Прецедент использования	Редактирование информации пользователя	Редактирование информации блогов	Редактирование категорий блогов
Краткое описание	Этот вариант использования позволяет Администратору редактировать информацию пользователя в системе.	Этот вариант использования позволяет Администратору редактировать информацию блогов в системе.	Этот вариант использования позволяет Администратору редактировать категории блогов в системе.
Действующие лица	Администратор	Администратор	Администратор
Предусловия	Действующее лицо Администратор желает редактировать информацию пользователя.	Действующее лицо Администратор желает редактировать информацию блогов.	Действующее лицо Администратор желает редактировать категории блогов.
Основной поток	Прецедент использования начинается с того, что Администратор решает редактировать информацию пользователя и входит в раздел пользователи на странице администратора, вносит изменения в поля данных пользователя, после чего нажимает кнопку сохранить. Прецедент использования завершается.	Прецедент использования начинается с того, что Администратор решает редактировать информацию блогов и входит в раздел пользователи на странице администратора, вносит изменения в поля данных статей, после чего нажимает кнопку сохранить. Прецедент использования завершается.	Прецедент использования начинается с того, что Администратор решает редактировать категории блогов и входит в раздел пользователи на странице администратора, вносит изменения в поля данных категорий блогов, после чего нажимает кнопку сохранить. Прецедент использования завершается.
Постусловия	Информация пользователя изменена.	Информация данного блога изменена.	Информация категорий блогов изменена.

Таблица 1.4 Описание прецедента Редактирования информации пользователя, длогов и категорий.

ДИАГРАММЫ СОСТОЯНИЙ МОДЕЛИ

Диаграммы состояний модели в основном используются для описания различных состояний объекта, процесса перехода между состояниями, а также различных событий и условий, запускающих переходы между состояниями. Веб-страница изображается в виде прямоугольника с закругленными углами, внутри которого указывается название. Переходы на диаграмме деятельности снабжаются текстовыми метками следующего формата: eventName [guardCondition] / actions, где eventName – имя события, которое может вызвать переход; guardCondition – сторожевое условие, соблюдение которого требуется для запуска перехода; actions – список действий, которые выполняются при запуске перехода.

Диаграмма состояний, показанная на рисунке ниже, описывает возможные действия Гостя в этой системе. Его конечным состоянием является авторизация пользователя. Эта диаграмма состояний в основном рассматривает состояние регистрации и состояние авторизации, оба из которых должны проверять достоверность данных.

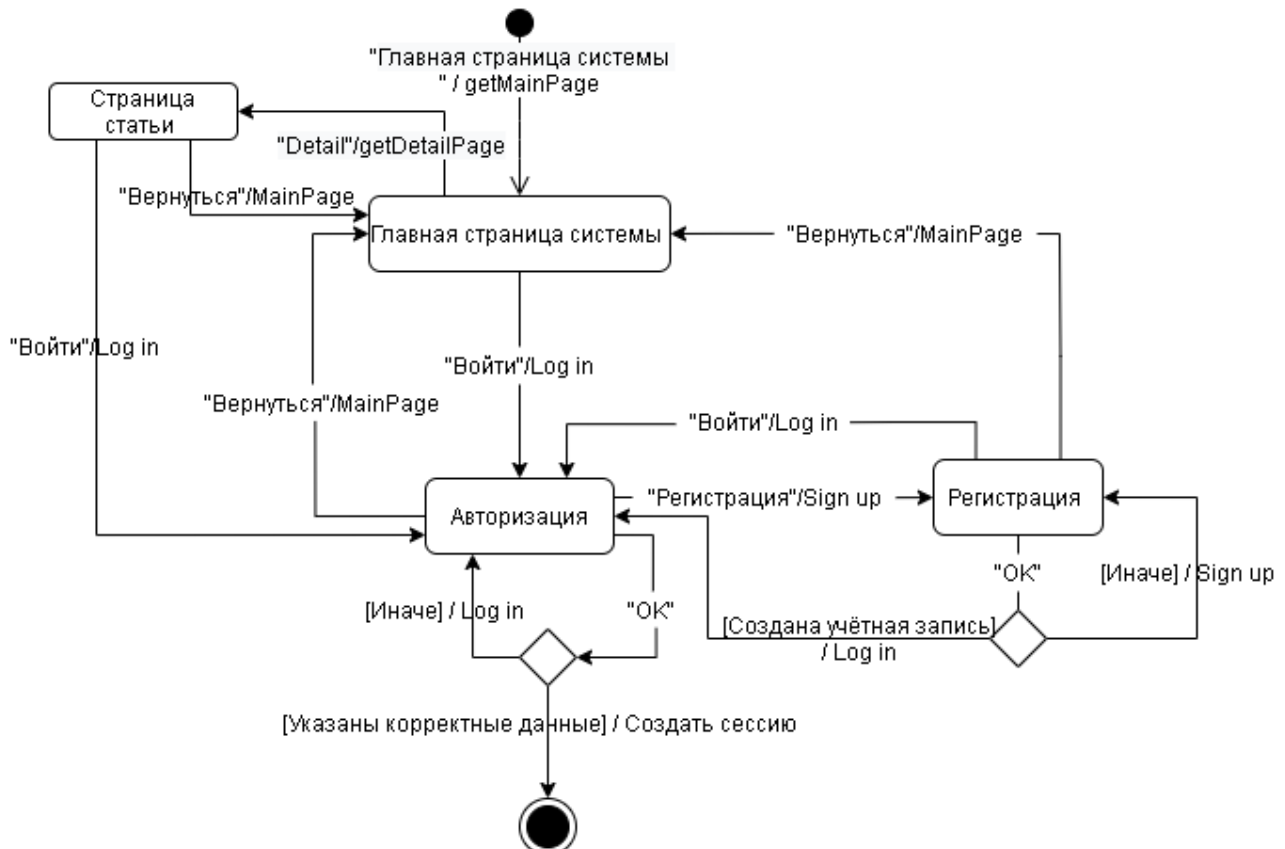


Рисунок 2.1 – Диаграмма состояний модели. (Гость)

На рисунке 2.2 изображена диаграмма состояний авторизованных пользователей, показывающая переходы различных страниц, и конечным состоянием является выход из системы.

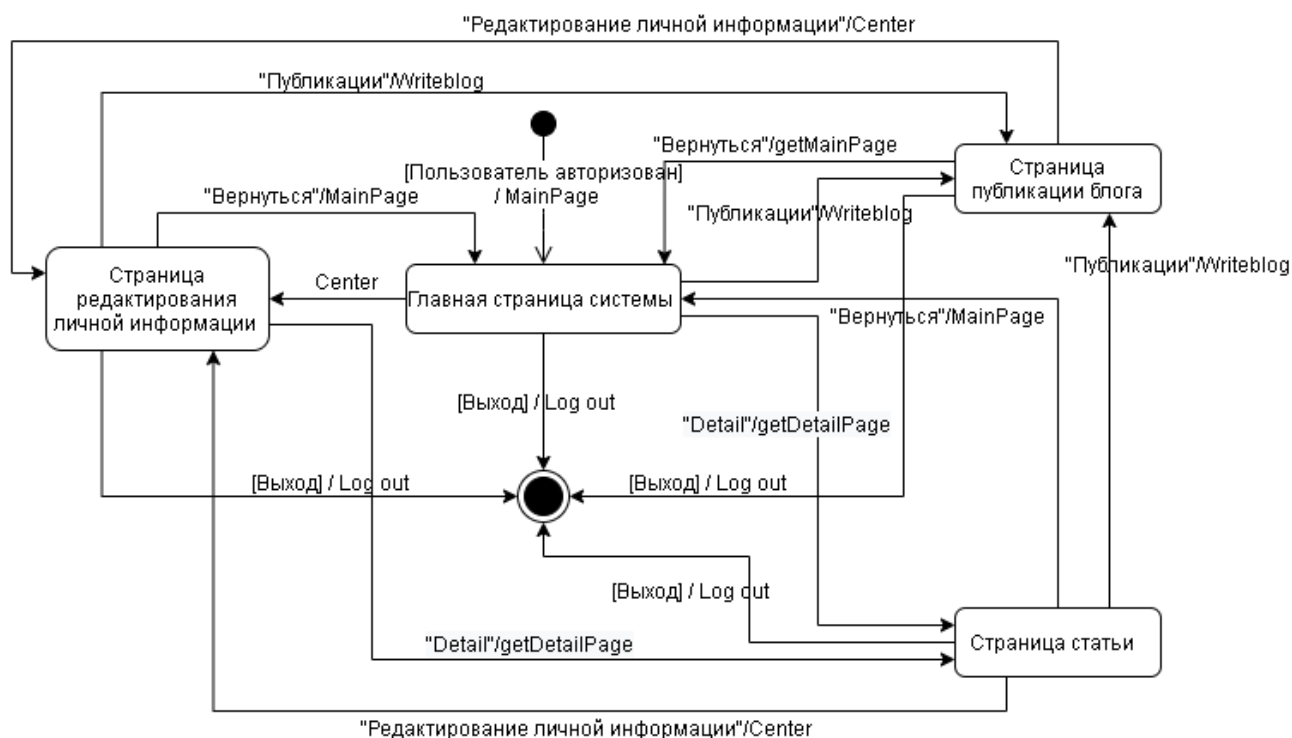


Рисунок 2.2 – Диаграмма состояний модели. (Пользователь)

ДАТАЛОГИЧЕСКАЯ МОДЕЛЬ СУЩНОСТЕЙ

Django предоставляет унифицированный API вызовов для различных баз данных, и в этом проекте используется база данных MySQL. Модель Django использует собственный ORM для преобразования кода Python в операторы SQL. Операторы SQL отправляются на сервер базы данных через `rumysql`, где операторы SQL выполняются в базе данных и возвращаются результаты.

На следующем рисунке представлена даталогическая модель сущностей, которые необходимо создать в этой системе. Вы можете понять, какие атрибуты должны иметь созданные сущности (Пользователь, Статьи, Комментарии и Категории статей), а также связи между ними.

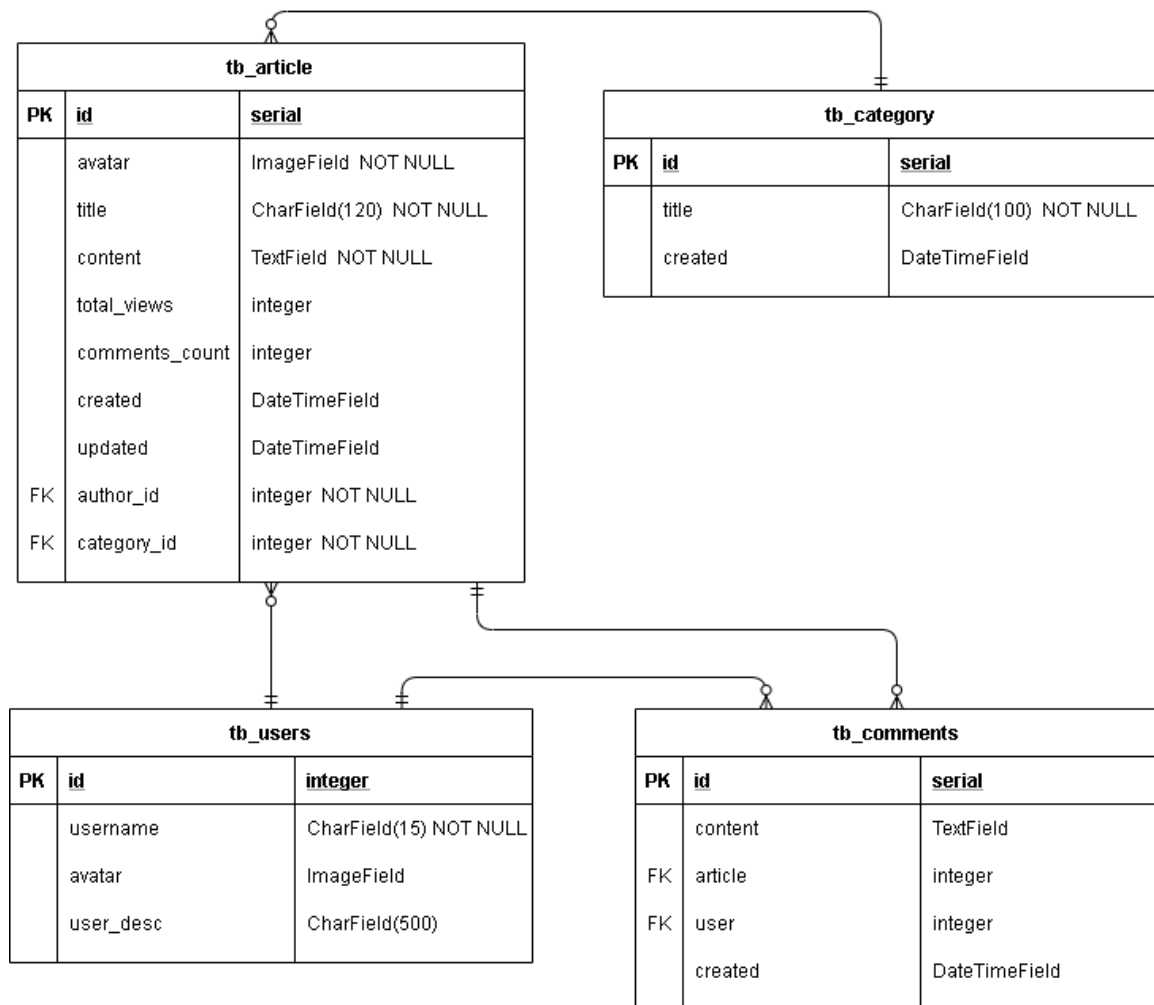


Рисунок 3.1 – даталогическая модель сущностей

```
class User(AbstractUser):
    #account
    username=models.CharField(max_length=15,unique=True, blank=False)
    #Avatar
    avatar=models.ImageField(upload_to='avatar/%Y%M%D', blank=True)
    #Introduction
    user_desc=models.CharField(max_length=500, blank=True)
    USERNAME_FIELD = 'username'
    REQUIRED_FIELDS = ['email']
    class Meta:
        db_table='tb_users'
        verbose_name='User_Management' #admin background display
        verbose_name_plural=verbose_name #admin background display
    def __str__(self):
        return self.username
```

Рисунок 3.2 – Код для создания модели User

```

class ArticleCategory(models.Model):
    title = models.CharField(max_length=100, blank=True)
    created = models.DateTimeField(default=timezone.now)

    def __str__(self):
        return self.title

    #admin site display, easy to debug and view objects
    class Meta:
        db_table = 'tb_category' #modify table name
        verbose_name = 'Category management' #admin site display
        verbose_name_plural = verbose_name

```

Рисунок 3.3 – Код для создания модели Category

```

class Article(models.Model):
    #on_delete: When the data in the user table is deleted, the article information is also deleted synchronously
    author = models.ForeignKey(User, on_delete=models.CASCADE)
    #author = models.CharField(max_length=50, blank=True)
    avatar = models.ImageField(upload_to='article/%Y%m%d/', blank=True)
    title = models.CharField(max_length=120, blank=True)
    category = models.ForeignKey(ArticleCategory, null=True, blank=True, on_delete=models.CASCADE, related_name='article')
    #tags = models.CharField(max_length=50, blank=True)
    #summary = models.CharField(max_length=600, null=False, blank=False)
    content = models.TextField()
    total_views = models.PositiveIntegerField(default=0)
    comments_count = models.PositiveIntegerField(default=0)
    created = models.DateTimeField(default=timezone.now)
    updated = models.DateTimeField(auto_now=True)
    #Modify the table name and configuration information.
    class Meta:
        db_table = 'tb_article'
        ordering = ('-created',)
        verbose_name = 'Article management'
        verbose_name_plural = verbose_name
        unique_together = (('avatar', 'title', 'category'),)

    def __str__(self):
        return self.title

```

Рисунок 3.4 – Код для создания модели Article

```

class Comment(models.Model):
    content = models.TextField()
    article = models.ForeignKey(Article, on_delete=models.SET_NULL, null=True)
    user = models.ForeignKey('users.User', on_delete=models.SET_NULL, null=True)
    created = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.article.title

    class Meta:
        db_table = 'tb_comment'
        verbose_name = 'Comment management'
        verbose_name_plural = verbose_name

```

Рисунок 3.5 – Код для создания модели Comment

На рисунках 3.2, 3.3, 3.4 и 3.5 показаны коды, используемые для создания моделей Пользователь, Категория, Статья, Комментарий соответственно. Стоит отметить, что в связанных моделях, таких как Статья и Пользователь, если пользователь удален, статьи, опубликованные этим пользователем, также будут удалены.

КОД ПРОГРАММЫ

На рисунке 4.1 показан код setting.py. Он указывает, что аутентифицированный пользователь системы использует определенную нами модель пользователя вместо пользовательской модели аутентификации по умолчанию системы. Также изменен путь входа в систему и путь сохранения и доступа к изображениям.

```
#Replace the User of the system to use the User defined by ourselves  
#The configuration information is "Sub-application name. Model type"  
AUTH_USER_MODEL='users.User'  
  
#modify the default redirect link when user is not logged in  
LOGIN_URL='/login/'  
  
#setting for the uploaded pictures, save them to the directory 'media'  
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')  
  
#setting for the unified route for image access.  
MEDIA_URL = '/media/'
```

Рисунок 4.1 - Фрагмент кода программы setting.py.

На рисунке 4.2 показан код /home/urls.py. Этот код связывает функцию view с путем ссылки.

```
urlpatterns = [  
    #The first parameter of path: routing  
    #The second parameter of path: view function name  
    path('register/', RegisterView.as_view(), name='register'),  
  
    #Image verification code routing  
    path('imagecode/', ImageCodeView.as_view(), name='imagecode'),  
  
    path('login/', LoginView.as_view(), name='login'),  
  
    path('logout/', LogoutView.as_view(), name='logout'),  
  
    path('forgetpassword/', ForgetPasswordView.as_view(), name='forgetpassword'),  
  
    path('center/', UserCenterView.as_view(), name='center'),  
  
    path('writeblog/', WriteBlogView.as_view(), name='writeblog'),  
]
```

Рисунок 4.3 - Фрагмент кода программы /home/urls.py.

На рисунке 4.3 показан фрагмент кода thefilmblog.py. Использовала BeautifulSoup, чтобы разместить на сайте контент блогов о фильмах из сайта <https://thefilm.blog/>.

```
import requests
from bs4 import BeautifulSoup

URL = ["https://thefilm.blog/", "https://thefilm.blog/category/news/",
      "https://thefilm.blog/category/previews/", "https://thefilm.blog/category/reviews/"]

def get_article(url):
    url_ = url
    article = []
    for p in range(1,3):
        url = url_ + "page/" + str(p) + "/"
        html = requests.get(url).text
        soup = BeautifulSoup(html, "html.parser")
        for id, articles in enumerate(soup.find_all("article")):
            category = articles.find("span", {"class": "cat-links"}).get_text()
            title = articles.find("h1").get_text()
            img = articles.find("img").get("data-orig-file")
            detail_url = articles.find("h1").find("a").get("href")
            article.append({"category": category, "title": title, "avatar": img, "url": detail_url})
    return article
```

Рисунок 4.4 - Фрагмент кода программы thefilmblog.py.

На рис. 4.5 показан код IndexView в /home/view.py. Когда пользователь открывает главную страницу веб-сайта, соответствующая информация, которая должна отображаться на главной странице, например информация о категории статей и информация о пейзаже, получается посредством запроса GET. Если эта категория не существует или эта страница не существует, будет выведена соответствующая информация об ошибке.

```

class IndexView(View):
    def get(self, request):
        get_blogs()
        #1. Receive info of category
        categories = ArticleCategory.objects.all()
        # 2. Get the category id clicked by user
        cat_id = request.GET.get('cat_id', 1)
        # 3. Query category by the category id
        try:
            category = ArticleCategory.objects.get(id=cat_id)
        except ArticleCategory.DoesNotExist:
            return HttpResponseRedirect('No such category!')
        # 4. Get paging parameters
        page_num = request.GET.get('page_num', 1)
        page_size = request.GET.get('page_size', 6)
        # 5. Query article data by the info of paging
        articles = Article.objects.filter(category=category)
        # 6. Create Paginator
        paginator = Paginator(articles, page_size)
        # 7. Paging
        try:
            page_articles = paginator.page(page_num)
        except EmptyPage:
            return HttpResponseRedirect('Empty Page!')
        #total page
        total_page = paginator.num_pages
        # 8. Translate data to templates

        context = {
            'categories': categories,
            'category': category,
            'articles': page_articles,
            'page_size': page_size,
            'total_page': total_page,
            'page_num': page_num
        }

        return render(request, 'index.html', context=context)

```

Рисунок 4.5 - Фрагмент кода программы /home/view.py.

На рис. 4.5 представлен код ImageCodeView в /users/view.py, который используется в регистрации пользователя.

```

class ImageCodeView(View):
    def get(self, request):
        # 1. Receive the uuid passed by the front end
        uuid = request.GET.get('uuid')
        # 2. Determine whether uuid is obtained
        if uuid is None:
            return HttpResponseRedirect('No uuid is passed.')
        # 3. Generate image verification code (image binary and image content) by calling captcha
        text, image = captcha.generate_captcha()
        # 4. Save the image content to redis
        redis_conn = get_redis_connection('default')
        #key -> 'code'
        #seconds -> Expired seconds (300s)
        #value -> text
        redis_conn.set(name='code', value=text, ex=120)
        # 5. Return image binary
        return HttpResponse(image, content_type='image/jpeg')

```

Рисунок 4.6 - Фрагмент кода программы /users/view.py.

Остальной код можно будет посмотреть на странице github - <https://github.com/YanTsyafen/blog>

ВИЗУАЛЬНЫЙ РЕЗУЛЬТАТ РАБОТЫ

Вся работа над фронтендом сайта реализована с использованием html + CSS + Javascript + DTL, где DTL - язык шаблонов Django, позволяющий интегрировать код python в html.

На рисунке 5.1 показана главная страница, независимо от того, авторизован пользователь или нет.

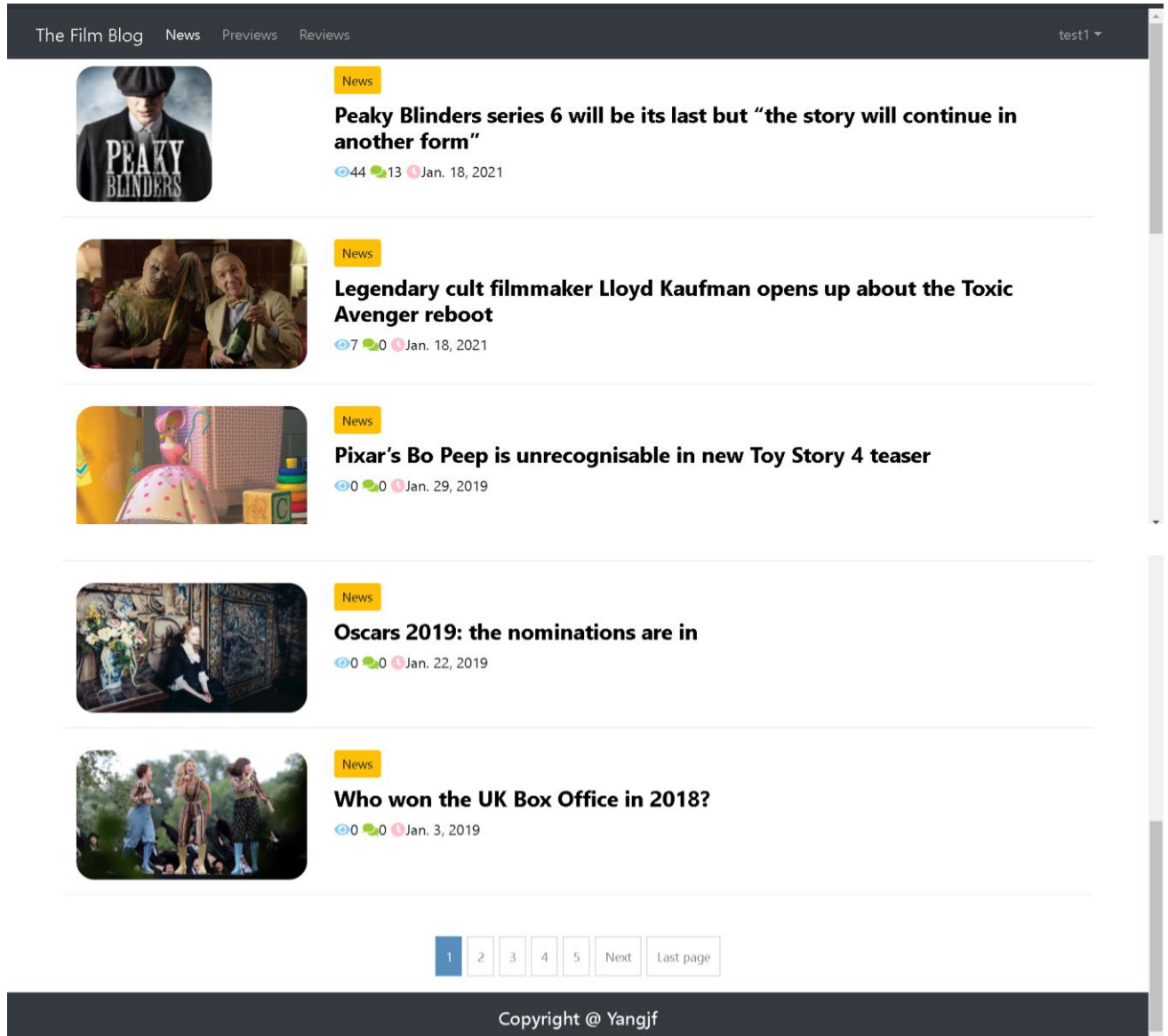


Рисунок 5.1 - Главная страница(News)

На рис. 5.2 показана страница статьи, когда пользователь не авторизован, и появляется сообщение с приглашением «войти в систему».

Peaky Blinders series 6 will be its last but “the story will continue in another form”

Author: test1
Views: 45

Fans awaiting the next season of the BBC's **Peaky Blinders** have some good news and bad news. The good news is production on the sixth season has now resumed after a lengthy lockdown due to the global health crisis, but the bad news however is it will be the series' final season.

The news was announced by series creator and writer Steven Knight, who had previously planned for a seventh season of the series. Despite season six now being the final one, Knight promised that the story will continue through other means.

“**Peaky** is back and with a bang,” said Knight. “After the enforced production delay due to the Covid pandemic, we find the family in extreme jeopardy and the stakes have never been higher. We believe this will be the best series of all and are sure that our amazing fans will love it. While the TV series will be coming to an end, the story will continue in another form.”

ut the world of **Peaky Blinders** will most definitely live on.”

“We are very excited that filming for **Peaky Blinders** has begun and so grateful to everyone for all their hard work to make it happen,” said BBC's Tommy Bulfin. “Steve's scripts for series six are truly remarkable and provide a fitting send-off which we are sure will delight fans.”

***Peaky Blinders** is set in the lawless streets of Birmingham as it tracks the evolution of leader Tommy Shelby (Cillian Murphy) from backstreet crime lord to legitimate businessman and member of parliament. This new season finds the world thrown into turmoil by the financial crash of 1929. When Tommy Shelby MP is approached by a charismatic politician with a bold vision for Britain, he realizes that his response will impact not only the future of his family but that of the entire nation.*

Peaky Blinders stars Cillian Murphy (**Dunkirk**) as Tommy Shelby, Helen McCrory (**Skyfall**) as Polly Gray, Paul Anderson (**The Revenant**) as Arthur Shelby, Sophie Rundle (**Bodyguard**) as Ada Thorne (**Shelby**), and Finn Cole as Michael Gray. New cast members include Anya Taylor-Joy (**The Witch**) as Gina Gray, Brian Gleeson (**Phantom Thread**) as Jimmy McCavern, Neil Maskell (**In Darkness**) as Winston Churchill, Sam Claflin (**The Nightingale**) as Oswald Mosley and Stephen Graham (**Boardwalk Empire**).

Ricky Church – Follow me on [Twitter](#) for more movie news and nerd talk.

Recommend

Peaky Blinders series 6 will be its last but “the story will continue in another form”

Legendary cult filmmaker Lloyd Kaufman opens up about the Toxic Avenger reboot

The Great Hack | Review

The Gentlemen | Review

Ride Like a Girl | Review

Everything you need to know

about Can You Ever Forgive Me

FIRST LOOK: The Downton Abbey film gets a teaser trailer!

October 2018: All the films coming your way!

WATCH: Check out Pixar's thrilling new trailer for The Incredibles 2

Please [Log in](#) to comment

Рисунок 5.2 – Страница Статьи(пользователь не авторизован)

На рисунках 5.3 и 5.4 показаны страницы входа и регистрации соответственно. Код подтверждения изображения на странице регистрации может отображаться как обычно, и щелкните его, чтобы заменить код подтверждения изображения.

Username

Please enter your username

No account yet? [Register a new account](#)

Password:

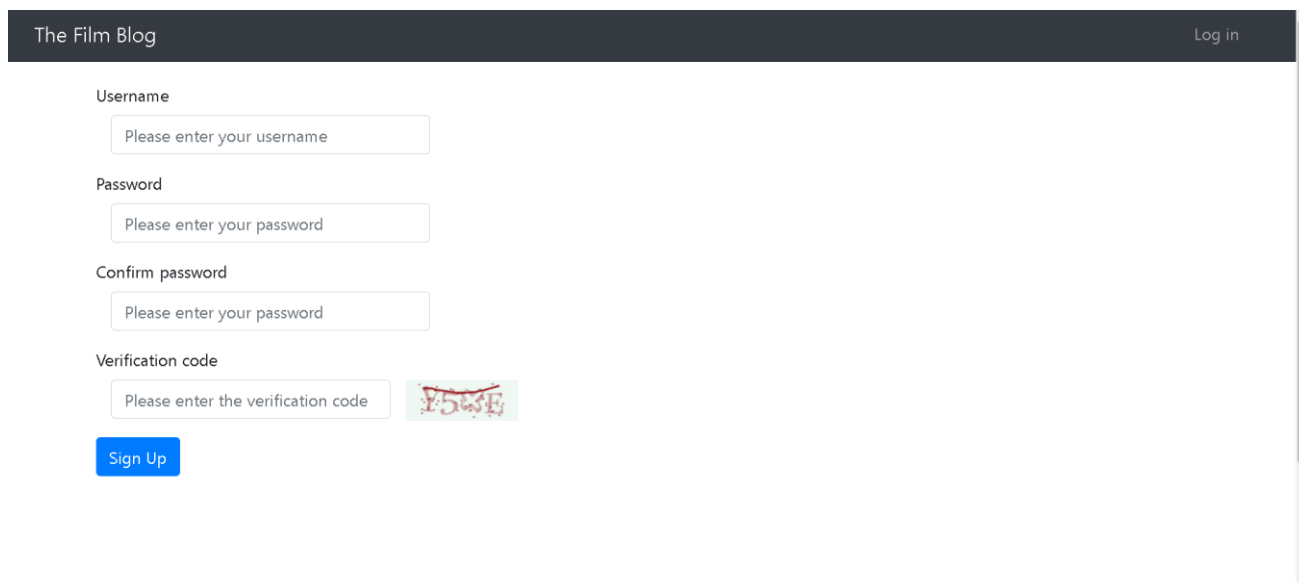
Please enter your password

[Forgot password?](#)

☐ Remember me

Login

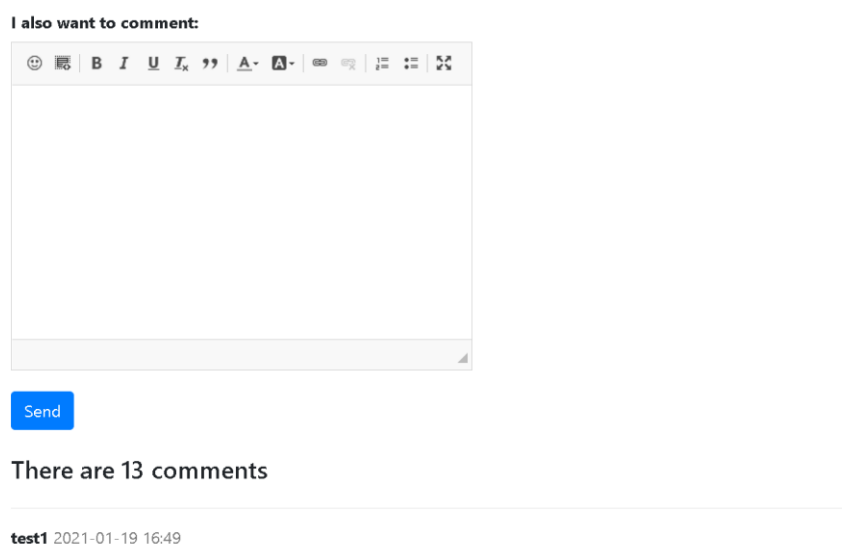
Рисунок 5.3 – Страница Log in



The registration page of 'The Film Blog' features a dark header with the site name on the left and a 'Log in' link on the right. The main content area contains a registration form with the following fields: 'Username' (placeholder: 'Please enter your username'), 'Password' (placeholder: 'Please enter your password'), 'Confirm password' (placeholder: 'Please enter your password'), and 'Verification code' (placeholder: 'Please enter the verification code'). To the right of the verification code field is a small image showing a red, noisy code '1545'. Below the form is a blue 'Sign Up' button.

Рисунок 5.4 – Страница регистрации

На рисунке 5.5 показана страница статьи, когда пользователь авторизован, и появляется область для комментариев.



The article page shows a comment section titled 'I also want to comment:'. It includes a rich text editor with a toolbar containing icons for text formatting (bold, italic, underline, strikethrough, quote), text color, background color, link, unlink, list, and image. Below the editor is a blue 'Send' button. Underneath the button, it says 'There are 13 comments'. A horizontal line separates this from a single comment by 'test1' dated '2021-01-19 16:49'.

Рисунок 5.5 – Страница Статьи(пользователь авторизован)

На рисунках 5.6 и 5.7 соответственно показаны страницы для публикации блогов и изменения личной информации, которые могут появиться только после авторизации пользователя.

[illegible]

Рисунок 5.6 – Страница публикации блогов (пользователь авторизован)


The Film Blog

test1 ▾

Username

admin

Avatar



Upload avatar

选择文件 未选择任何文件

Introduction

Submit

Copyright @ Yangjf

Рисунок 5.7 – Страница редактирования личной информации (пользователь авторизован)

На рисунках с 5.8 по 5.11 показана страница администратора, включая управление пользователей, статей и категорий статей.

Django administration

WELCOME, ADMIN. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups [+ Add](#) [Change](#)

HOME

Article management [+ Add](#) [Change](#)

Category management [+ Add](#) [Change](#)

USERS

User_Management [+ Add](#) [Change](#)

Recent actions

My actions

[+ Reviews](#)
Category management

[+ Previews](#)
Category management

[+ News](#)
Category management

Рисунок 5.8 – Страница administration

Django administration WELCOME, **ADMIN**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Home > Category management

Select Category management to change ADD CATEGORY MANAGEMENT +

Action: Go 0 of 3 selected

<input type="checkbox"/>	CATEGORY MANAGEMENT
<input type="checkbox"/>	Reviews
<input type="checkbox"/>	Previews
<input type="checkbox"/>	News

3 Category management

Рисунок 5.9 – Страница administration(Category)

Django administration WELCOME, **ADMIN**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Users > User_Management

Select User_Management to change ADD USER_MANAGEMENT +

Action: Go 0 of 4 selected

<input type="checkbox"/>	USER_MANAGEMENT
<input type="checkbox"/>	thefilmblog
<input type="checkbox"/>	test2
<input type="checkbox"/>	admin
<input type="checkbox"/>	test1

4 User_Management

Рисунок 5.10 – Страница administration(User)

Django administration WELCOME, **ADMIN**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Home > Article management

Select Article management to change ADD ARTICLE MANAGEMENT +

Action: Go 0 of 48 selected

<input type="checkbox"/>	ARTICLE MANAGEMENT
<input type="checkbox"/>	Peaky Blinders series 6 will be its last but "the story will continue in another form"
<input type="checkbox"/>	Legendary cult filmmaker Lloyd Kaufman opens up about the Toxic Avenger reboot
<input type="checkbox"/>	Ride Like a Girl Review
<input type="checkbox"/>	Artemis Fowl Review
<input type="checkbox"/>	Emma. Review
<input type="checkbox"/>	1917 Review
<input type="checkbox"/>	The Gentlemen Review
<input type="checkbox"/>	Marriage Story Review
<input type="checkbox"/>	The Great Hack Review
<input type="checkbox"/>	April 2019: All the films coming your way!
<input type="checkbox"/>	March 2019: All the films coming your way!

Рисунок 5.11 – Страница administration(Article)

ТЕСТИРОВАНИЕ

Этот системный тест требует ручного тестирования.

Во время теста была обнаружена следующая ошибка: область комментариев также появлялась, когда пользователь не был авторизован (как показано на рисунке 6.1). Но обычно область для комментариев должна появляться только тогда, когда пользователь авторизован, поэтому изменила как показанное на рисунке 6.2: добавила `v-show = "is_login"` в код тега области комментариев. Результат после изменения показан на Рисунке 6.3.

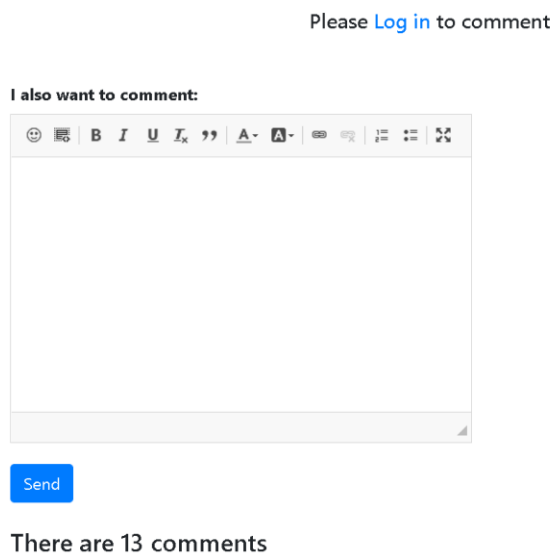


Рисунок 6.1 – Страница Статьи(пользователь не авторизован)

```
<h5 class="row justify-content-center" v-show="!is_login"><p>Please <a href="{% url 'users:login' %}">Log in</a> to comment</p>
</h5>
<br>
<div v-show="is_login">
  <form method="POST">
    {% csrf_token %}
    <input type="hidden" name="id" value="{{ article.id }}">
    <div class="form-group"><label for="body"><strong>I also want to comment:</strong></label>
      <div>
        <div class="django-ckeditor-widget" data-field-id="id_body" style="...">
          <textarea cols="40" id="id_body" name="content" rows="10" required data-processed="0" :data-config="data_config">
            </textarea>
          </div>
        </div>
      </div>
    <button type="submit" class="btn btn-primary">Send</button>
  </form>
</div>
```

Рисунок 6.2 – Изменение кода detail.html

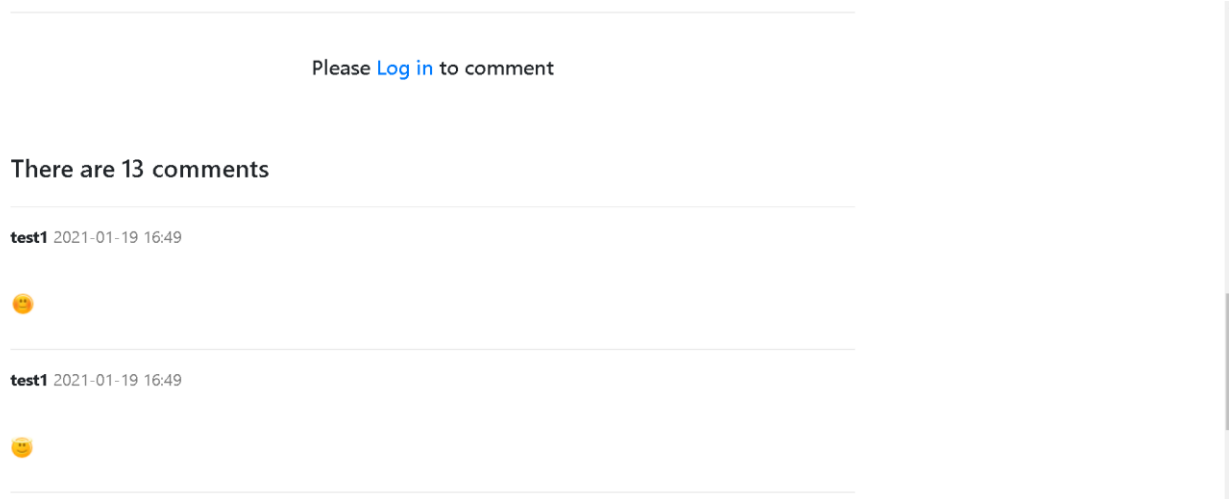


Рисунок 6.3 – Страница Статьи(пользователь не авторизован) после изменения

ЗАКЛЮЧЕНИЕ

Реализовано веб-приложение с функцией регистрации, входа в систему, изменения паролей, изменения личной информации пользователя, публикации блога, публикации комментариев, агрегации новостей со сторонних сайтов (<https://thefilm.blog/>), рекомендации статей и пейджинга.

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)

Студент Ян Цзяфэн

Кафедра Компьютерных образовательных технологий группы Р33212

Руководитель Штенников Д.Г., СПб НИУ ИТМО, кафедра КОТ, доцент
(Фамилия, И., О., место работы, должность)

Дисциплина Разработка веб-приложений

Наименование темы Проектирование, реализация и тестирование блога о фильмах с функцией агрегации новостей из сторонних источников.

Задание Проектирование веб-приложения с функцией агрегации новостей из сторонних источников. Реализация данного приложения на языке python с использованием фреймворка Django. Дальнейшее тестирование и исправление ошибок найденных во время тестирования. Оформление пояснительной записки.

Студент Дата «_____» _____ 2020г.

Подпись

Дата

Руководитель Дата «_____» _____ 2020 г.

Подпись

Дата

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

АННОТАЦИЯ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)

Студент Ян Цзяфэн

Кафедра Компьютерных образовательных технологий **группы** P33212

Руководитель Штенников Д.Г., СПб НИУ ИТМО, кафедра КОТ, доцент
(Фамилия, И., О., место работы, должность)

Дисциплина Разработка веб-приложений

Наименование темы Проектирование, реализация и тестирование блога о фильмах с функцией агрегации новостей из сторонних источников.