

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University)

COIMBATORE - 641013



OPEN SOURCE USAGE SKILLS LABORATORY

BATCH-12

NEWS AGGREGATOR USING DJANGO

PROJECT

WORK-

2020

Submitted by

HEMACHANDRAN G -
1717L01

KALAIARASU M -
1717L02 KALAVATHI

S -1717L03 KARTHIK
M -1717L04 KOMATHI

– 1717L05

NEWS AGGREGATOR USING DJANGO

AIM:

In News Aggregator, a user can add any source he wants from the given catalog and get news from the chosen sources as per his wish. The app allows maximum control over the sources available and highly flexible. A signed user can customize the feed by using his profile, whereas the not signed users have to sign in to have more customization over the feed.

INTRODUCTION:

PURPOSE:

The main purpose of the app is to give a one-stop solution for news from multiple sources to an end-user. The ease of usage and simple UX should encourage the user to get more from the app and provide a rich source of data accessible from one platform.

PROJECT SCOPE:

- One-stop solution for multiple sources.
- A single account to access news from humongous internet articles.
- No need to browse through multiple sources, saves time.
- Having an account is not mandatory.
- Can customize the sources of articles by modifying the profile.

USER CLASSES AND CHARACTERISTICS:

The app has two actors. It is divided into,

- End-user
- Bot

End-User:

- Log in/out of the platform.
- Consume news articles.
- Manage profile.
- Filter articles based on the source.
- Access a huge volume of news stored in the internal databases.

Bot:

- Crawls through multiple RSS feeds.
- Fetch new articles and pass them to the platform.
- Gathers required info for providing Metadata about the news article.

OPERATING ENVIROMENT:

- Operating System : Windows 10 Pro
- Database : MySQL
- Front end : JavaScript, HTML, CSS, Bootstrap
- Back end : Python, Django

➤ Libraries : Pillow, FeedParser,BeautifulSoup, CrispyForms, GUnicorn, Requests

DESIGN IMPLEMENTATION:

CODE:

requirements.txt

```
asgiref==3.2.10
autopep8==1.5.4
beautifulsoup4==4.9.1
certifi==2020.6.20
chardet==3.0.4
Django==3.1.1
django-crispy-forms==1.9.2
feedparser==5.2.1
gunicorn==20.0.4
idna==2.10
Pillow==7.2.0
pycodestyle==2.6.0
python-dateutil==2.8.1
pytz==2020.1
requests==2.24.0
six==1.15.0
soupsieve==2.0.1
sqlparse==0.3.1
toml==0.10.1
urllib3==1.25.10
whitenoise==5.2.0
```

models.py

```
from django.db import models
from django.contrib.auth.models import User
```

```

class Subscription(models.Model):
    name =
    models.CharField(max_length=30)
    thumbnail = models.URLField()
    last_updated = models.DateTimeField()
    site_link = models.URLField()
    feed_link = models.URLField()
    def __str__(self):
    return self.name
class Article(models.Model):
    subscription_name = models.ForeignKey(to=Subscription,
    on_delete=models.CASCADE) published = models.DateTimeField()
    title = models.CharField(max_length=300)
    author = models.CharField(default=' ', max_length=100)
    summary = models.CharField(default=' ',
    max_length=500) media = models.URLField()
    article_id = models.CharField(max_length=200,
    unique=True) article_link = models.URLField()
    def __str__(self):
        return f"{self.subscription_name.name}: {self.title}"

class Profile(models.Model):
    name = models.ForeignKey(to=User,
    on_delete=models.CASCADE) subscription =
    models.ForeignKey(
        to=Subscription, on_delete=models.CASCADE)
    def __str__(self):
    return f"{self.name.username} - {self.subscription.name}"

```

urls.py

```

from django.urls import path

from .views import (HomePageView, SignupView, aggregator,
    HomeListView, SubscriptionListView, profileView,
    addProfile,

```

```

        removeProfile,
    )

urlpatterns = [
    path('', HomeListView.as_view(), name='home'),
    path('signup/', SignupView.as_view(), name='signup'),
    path('subscription/<str:name>', SubscriptionListView.as_view(), name='subscription'),
    path('aggregator/', aggregator, name='aggregator'),
    path('profile/', profileView, name='profile'),
    path('add/<int:id>', addProfile, name='add-profile'),
    path('remove/<int:id>', removeProfile, name='remove-profile'),
]

```

views.py

```

from django.views.generic import TemplateView,
ListView from django.contrib.auth.forms import
UserCreationForm from django.core.paginator import
Paginator
from django.urls import reverse_lazy
from django.views import generic
from django.http import HttpResponseRedirect
from django.shortcuts import render, get_object_or_404
from django.contrib.auth.decorators import
login_requiredimport feedparser as fp

from bs4 import BeautifulSoup
from dateutil.parser import parse as p
import datetime as dt
import pytz
from .models import Subscription, Article, Profile

class
HomePageView(TemplateView):

```

```

template_name = "home.html"
class
SignupView(generic.CreateView):
    form_class = UserCreationForm
    success_url = reverse_lazy('login')
    template_name = 'signup.html'
    def summaryMaker(originalText):
        if len(originalText) >= 400:
            return originalText[0: 395] + "..."
        return originalText
    def aggregator(request):
        updateList = []
        subscriptions = Subscription.objects.all()
        for i in subscriptions:
            feed = fp.parse(i.feed_link)
            if p(feed.entries[0].published) != i.last_updated:
                i.last_updated = p(feed.entries[0].published)
                i.save()
            print('Updating ' + str(i.name))
            updateList.append(i.name)
        for j
        in feed.entries:
            if Article.objects.filter(article_id=j.id).count() == 0:
                pubTime = j.published
                title = j.title
                link = j.link
                iD = j.id
                summary = j.summary
                # Summary
                if i.name == 'Gizmo China' or i.name == 'GSM Arena':
                    htmlToText = BeautifulSoup(summary, "html.parser")
                    summary = htmlToText.text.split('\n')[0]
                    summary = summaryMaker(summary)

```

```

try:
    author = j.author except
Exception as e: author = ''
try:
media = j.media_content[0]['url'] except
Exception as e:
try:
media = j.links[1]['href'] except
Exception as e: try:
    media = htmlToText.find('img')['src'] except
Exception as e:
    media = i.thumbnail
    converted_time = p(pubTime).astimezone(pytz.timezone('Asia/Kolkata'))
    a = Article.objects.create(subscription_name=i, published=converted_time, title=title,
author=author, summary=summary, media=media, article_id=iD, article_link=link)
    a.save() print(f" {i.name}
done")

    return render(request, 'aggregator.html', {'updateList': updateList,})

class HomeListView(ListView):
    model = Article template_name
    = "home.html" paginate_by =
    20
    context_object_name = 'allData'

    def get_queryset(self):
        if self.request.user.is_authenticated:
            subs = Profile.objects.filter(name=self.request.user)
            lister = []
            for sub in subs:
                lister.append(sub.subscription)
            return Article.objects.filter(subscription_name_____in=lister).order_by('-published')

```

```
else:
    return Article.objects.all().order_by('-published')

class
    SubscriptionListView(ListView):
        model = Article
        template_name = 'subscription.html'
        paginate_by = 20
        context_object_name = 'allData'

    def get_queryset(self):
        subs = get_object_or_404(Subscription, name=self.kwargs.get('name')) return
        Article.objects.filter(subscription_name=subs).order_by('-published')

    @login_required
    def profileView(request):
        allSubscriptions = Subscription.objects.all()
        profile = Profile.objects.filter(name=request.user)
        lister=[]
        for pr in profile:
            lister.append(pr.subscription.name)
        context = {'subscriptions': allSubscriptions, 'profiles':
            lister} return render(request, 'profile.html', context)

    @login_required
    def addProfile(request, id):
        allSubscriptions = Subscription.objects.all()
        sub = get_object_or_404(Subscription, id=id)
        p = Profile.objects.create(name=request.user, subscription=sub)
        p.save()
        profile = Profile.objects.filter(name=request.user)
        lister = []
        for pr in profile:
            lister.append(pr.subscription.name)
```



```
return render(request, 'profile.html', {'subscriptions': allSubscriptions, 'profiles': lister})
```

```
@login_required
```

```
def removeProfile(request, id):
```

```
    sub = get_object_or_404(Subscription, id=id)
```

```
    p=Profile.objects.filter(name=request.user, subscription=sub).delete()
```

```
    allSubscriptions = Subscription.objects.all()
```

```
    # p.delete()
```

```
    profile = Profile.objects.filter(name=request.user)
```

```
    lister = []
```

```
    for pr in profile:
```

```
        lister.append(pr.subscription.name)
```

```
    return render(request, 'profile.html', {'subscriptions': allSubscriptions, 'profiles': lister})
```

admin.py

```
from django.contrib import admin
```

```
from .models import Subscription, Article, Profile
```

```
admin.site.register(Subscription)
```

```
admin.site.register(Article)
```

```
admin.site.register(Profile)
```

newsaggregator/urls.py

```
from django.contrib import admin
```

```
from django.urls import path, include
```

```
from django.conf import settings
```

```
from django.conf.urls.static import static
```

```
urlpatterns = [
```

```
    path('admin/', admin.site.urls),
```

```
    path('accounts/', include('django.contrib.auth.urls')),path("", include('blogs.urls')),
```

```
] + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
```

settings.py

```
from pathlib import Path
import os
# Build paths inside the project like this: BASE_DIR /
'subdir'. BASE_DIR = Path(__file_
_____).resolve(strict=True).parent.parent
# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'r28x--
^_@0nq8pp^m+=c7(n#p6au2mf*a=8fb^+dz5(jfbzjne'
# SECURITY WARNING: don't run with debug turned on in
production! DEBUG = True
ALLOWED_HOSTS = ['*']
# Application definition INSTALLED_APPS = [
'blogs.apps.BlogsConfig',
'django.contrib.admin',
'django.contrib.auth',
'django.contrib.contenttypes',
'django.contrib.sessions',
'django.contrib.messages',
'django.contrib.humanize',
'whitenoise.runserver_nostatic',
'django.contrib.staticfiles',
'crispy_forms',
]

MIDDLEWARE = [
'django.middleware.security.SecurityMiddleware',
'django.contrib.sessions.middleware.SessionMiddleware',
```

```
'whitenoise.middleware.WhiteNoiseMiddleware',
'django.middleware.common.CommonMiddleware',
'django.middleware.csrf.CsrfViewMiddleware',
'django.contrib.auth.middleware.AuthenticationMiddleware',
'django.contrib.messages.middleware.MessageMiddleware',
'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
```

```
ROOT_URLCONF =
```

```
'newsAggregator.urls' TEMPLATES = [
{
'BACKEND': 'django.template.backends.django.DjangoTemplates',
'DIRS': [os.path.join(BASE_DIR, 'templates')],
'APP_DIRS': True,
'OPTIONS': {
'context_processors': [
'django.template.context_processors.debug',
'django.template.context_processors.request',
'django.contrib.auth.context_processors.auth',
'django.contrib.messages.context_processors.messages',
],
},
},
]
```

```
WSGI_APPLICATION = 'newsAggregator.wsgi.application'
```

```
# Database
```

```
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases
```

```
DATABASES = {
```

```
'default': {
```

```
'ENGINE': 'django.db.backends.sqlite3',
```

```

'NAME': BASE_DIR / 'db.sqlite3',
}
}

# Password validation
# https://docs.djangoproject.com/en/3.1/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
{
'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
},
{
'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
},
{
'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
},
{
'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
},
]

# Internationalization
# https://docs.djangoproject.com/en/3.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N =

True USE_L10N

```

= True USE_TZ

= True

Static files (CSS, JavaScript, Images)

<https://docs.djangoproject.com/en/3.1/howto/static-files/>

STATIC_ROOT = os.path.join(BASE_DIR,

'staticfiles') STATIC_URL = '/static/'

STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]

STATICFILES_STORAGE =

'whitenoise.storage.CompressedManifestStaticFilesStorage'

LOGIN_REDIRECT_URL = 'home'

LOGOUT_REDIRECT_URL = 'home'

CRISPY_TEMPLATE_PACK = 'bootstrap4'

base.html

<!DOCTYPE html>

<html lang="en">

<head>

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-
Vkoo8x4CGsO3+Hh xv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9I
fjh" crossorigin="anonymous">

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>News Aggregator</title>

</head>

<body>

<nav class="navbar sticky-top navbar-expand-sm bg-dark navbar-dark">

<ul class="navbar-nav">

```

<li class="navbar-nav mr-auto">
<a href="{ % url 'home' % }" class="navbar-brand">News Aggregator</a></header>
</li>
</ul>

<div class="navbar-nav ml-auto">
{ % if user.is_authenticated % }
{ % if request.path == '/profile/' % }
  <a href="{ % url 'logout' % }" class="nav-link ml-auto">Logout</a>
{ % else % }
  <a href="{ % url 'profile' % }" class="nav-link ml-auto">Profile</a>
{ % endif % }
{ % else % }
  <a href="{ % url 'login' % }" class="nav-link">Login</a>
{ % endif % }
</div>
</nav>

<div class="container-sm">
{ % if user.is_authenticated == False % }
<div class="text-center">
<legend>
<br>
  <a class="btn btn-outline-info mb-4 mx-auto" href="{ % url 'login' % }">Login to customize the
feed</a>
</legend>
</div>
{ % endif % }
{ % block content % }
{ % endblock content % }
</div>

<script      src="https://code.jquery.com/jquery-3.4.1.slim.min.js"      integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
crossorigin="anonymous"></script>

```

```
<script      src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
crossorigin="anonymous"></script>
```

```
<script      src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"
integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1liqfktj0Uod8GCExl3Og8ifwB6"
crossorigin="anonymous"></script>
```

```
</body>
```

```
</html>
```

home.html

```
{% extends 'base.html' %}
```

```
{% load humanize %}
```

```
{% block content %}
```

```
<div>
```

```
{% if allData.count == 0 %}
```

```
<div class="text-center"><br>
```

```
<a href="{% url 'profile' %}" class="btn btn-outline-info mb-4">Go profile to add sources</a>
```

```
</div>
```

```
{% endif %}
```

```
{% for data in allData %}
```

```
<br>
```

```
<div class="container-sm media border rounded" style="padding:20px">
```

```
<br>
```

```
<a href="{{ data.subscription_name.site_link }}">
```

```

```

```
</a>
```

```
<div class="media-body">
```

```
<h5 class="mt-0"><a href="{{ data.article_link }}" target="_blank">{{ data.title }}</a></h5>
```

```
{% if data.author == '' %}
```

```
<div>
```

```

<h6><a href="{% url 'subscription' data.subscription_name.name
%}">{{ data.subscription_name.name }}
</a><small>{{ data.published|naturaltime }}</small></h6>

</div>

{% else %}

<h6>{{ data.author }} from <a href="{% url 'subscription' data.subscription_name.name
%}">{{ data.subscription_name.name }}
</a><small>{{ data.published|naturaltime }}</small></h6>

{% endif %}

<p>{{ data.summary }}</p>

</div>

</div>

{% endfor %}
<br>

{% if is_paginated %}

{% if page_obj.has_previous %}

<a class="btn btn-outline-info mb-4" href="?page=1">First</a>

<a class="btn btn-outline-info mb-4" href="?page={{ page_obj.previous_page_number
}}">Previous</a>

{% endif %}

{% for num in page_obj.paginator.page_range %}

{% if page_obj.number == num %}

<a class="btn btn-info mb-4" href="?page={{ num }}">{{ num }}</a>

{% elif num > page_obj.number|add: '-3' and num < page_obj.number|add: '3' %}

<a class="btn btn-outline-info mb-4" href="?page={{ num }}">{{ num }}</a>

{% endif %}

{% endfor %}

{% if page_obj.has_next %}

<a class="btn btn-outline-info mb-4" href="?page={{ page_obj.next_page_number
}}">Next</a>

<a class="btn btn-outline-info mb-4" href="?page={{ page_obj.paginator.num_pages

```



```
}}">Last</a>
```

```
{% endif % }
```

```
{% endif % }
```

```
</div>
```

```
{% endblock content % }
```

aggregator.html

```
{% extends 'base.html' % }
```

```
{% load tz % }
```

```
{% block content % }
```

```
<div class="container-sm">
```

<p>The page is made for scraping content from sites. This determines the updates to the site</p>

<p>It will be updated every 10 minutes automatically to parse data. Use the link below to make the parse right now.</p>

```
{% if updateList % }
```

```
{% for update in updateList % }
```

```
<h4>Updated { {update} }</h4>
```

```
{% endfor % }
```

```
{% else % }
```

```
<h4>No Updates</h4>
```

```
{% endif % }
```

```
{% timezone "Asia/Kolkata" % }
```

```
<small>Last Updated at { % now "SHORT_DATETIME_FORMAT" % } </small>
```

```
{% endtimezone % }
```

```
<h1><a href="{ % url 'aggregator' % }" class="btn btn-outline-info mb-4">Update  
Now</a></h1>
```

```
</div>
```

```
<script>
```

```
setTimeout(function(){
```

```
window.location.reload(1);
```

```
    }, 1000 * 60 * 10);  
</script>  
{% endblock content % }
```

profile.html

```
{% extends 'base.html' %}  
{% block content %}  
{% if user.is_authenticated %}  
<br>  
<div style="margin-top:20px, width:50%;" class="container-sm border rounded">  
<br>  
<h3>Hello {{user.username|title}},</h3>  
{% comment % } {% for subs in subscriptions %}  
{% if subs.name in profiles %}  
<p><a href="{% url 'remove-profile' subs.id %}">Remove</a> {{subs.name}}</p>  
{% else %}  
<p><a href="{% url 'add-profile' subs.id %}">Add</a> {{subs.name}}</p>  
{% endif %}  
{% endfor % } {% endcomment %}  
<table class="table">  
<thead>  
<tr>  
<th>Source</th>  
<th>Status</th>  
</tr>  
</thead>  
{% for subs in subscriptions %}  
{% if subs.name in profiles %}  
<tr>  
<th>{{subs.name}}</th>  
<th><a href="{% url 'remove-profile' subs.id %}" class="btn btn-outline-danger mb-
```

```

4">Remove</a></th>
</tr>
{% else %}
<tr>
<th>{{ subs.name }}</th>
<th><a href="{% url 'add-profile' subs.id %}" class="btn btn-outline-success mb-4">Add</a></th>
</tr>
{% endif %}
{% endfor %}
</table>
<a href="{% url 'home' %}" class="btn btn-outline-info mb-4">Go to home</a>
</div>
{% endif %}

{% endblock content %}

```

subscription.html

```

{% extends 'base.html' %}
{% load humanize %}
{% block content %}
<div class="container-sm">
<legend>{{ page_obj.paginator.count }} posts from {{ view.kwargs.name }}</legend>
{% for data in allData %}
<br>
<div class="container-sm media border rounded" style="padding:20px">
<br>
<a href="{{ data.subscription_name.site_link }}">
    
</a>
<div class="media-body">

```

```

<h5 class="mt-0"><a href="{{ data.article_link }}" target="_blank">{{ data.title }}</a></h5>
{% if data.author != '' %}
<h6>by {{ data.author }} <small>{{ data.published|naturaltime }}</small></h6>
{% else %}
<small>{{ data.published|naturaltime }}</small>
{% endif %}
<p>{{ data.summary }}</p>
</div>
</div>
{% endfor %}

<br>

{% if is_paginated %}
{% if page_obj.has_previous %}
<a class="btn btn-outline-info mb-4" href="?page=1">First</a>
<a class="btn btn-outline-info mb-4" href="?page={{ page_obj.previous_page_number }}">Previous</a>
{% endif %}

{% for num in page_obj.paginator.page_range %}
{% if page_obj.number == num %}
<a class="btn btn-info mb-4" href="?page={{ num }}">{{ num }}</a>
{% elif num > page_obj.number|add:'-3' and num < page_obj.number|add:'3' %}
<a class="btn btn-outline-info mb-4" href="?page={{ num }}">{{ num }}</a>
{% endif %}
{% endfor %}

{% if page_obj.has_next %}
<a class="btn btn-outline-info mb-4" href="?page={{ page_obj.next_page_number }}">Next</a>
<a class="btn btn-outline-info mb-4" href="?page={{ page_obj.paginator.num_pages }}">Last</a>
{% endif %}

```

```
{% endif % }  
</div>  
{% endblock content % }
```

signup.html

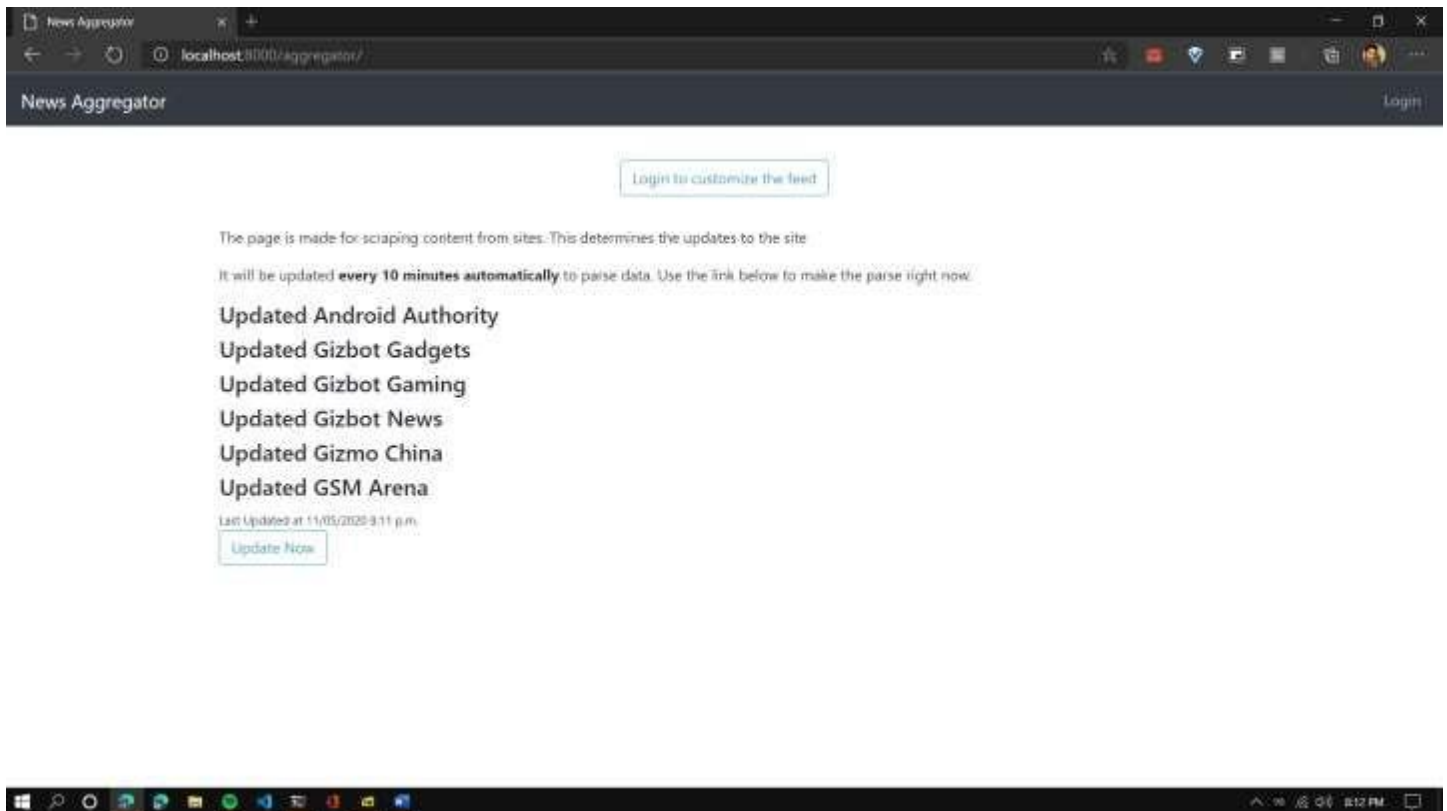
```
{% extends 'base.html' % }  
{% load crispy_forms_tags % }  
{% block content % }  
  
<div style="width:65%; margin: 0 auto;">  
  <h2><a href="{% url 'login' %}">Log In</a> | Sign up</h2>  
  <form method="post">  
    {% csrf_token % }  
    {{ form|crispy }}  
    <button type="submit" class="btn btn-primary">Sign up</button>  
  </form>  
</div>  
{% endblock content % }
```

login.html

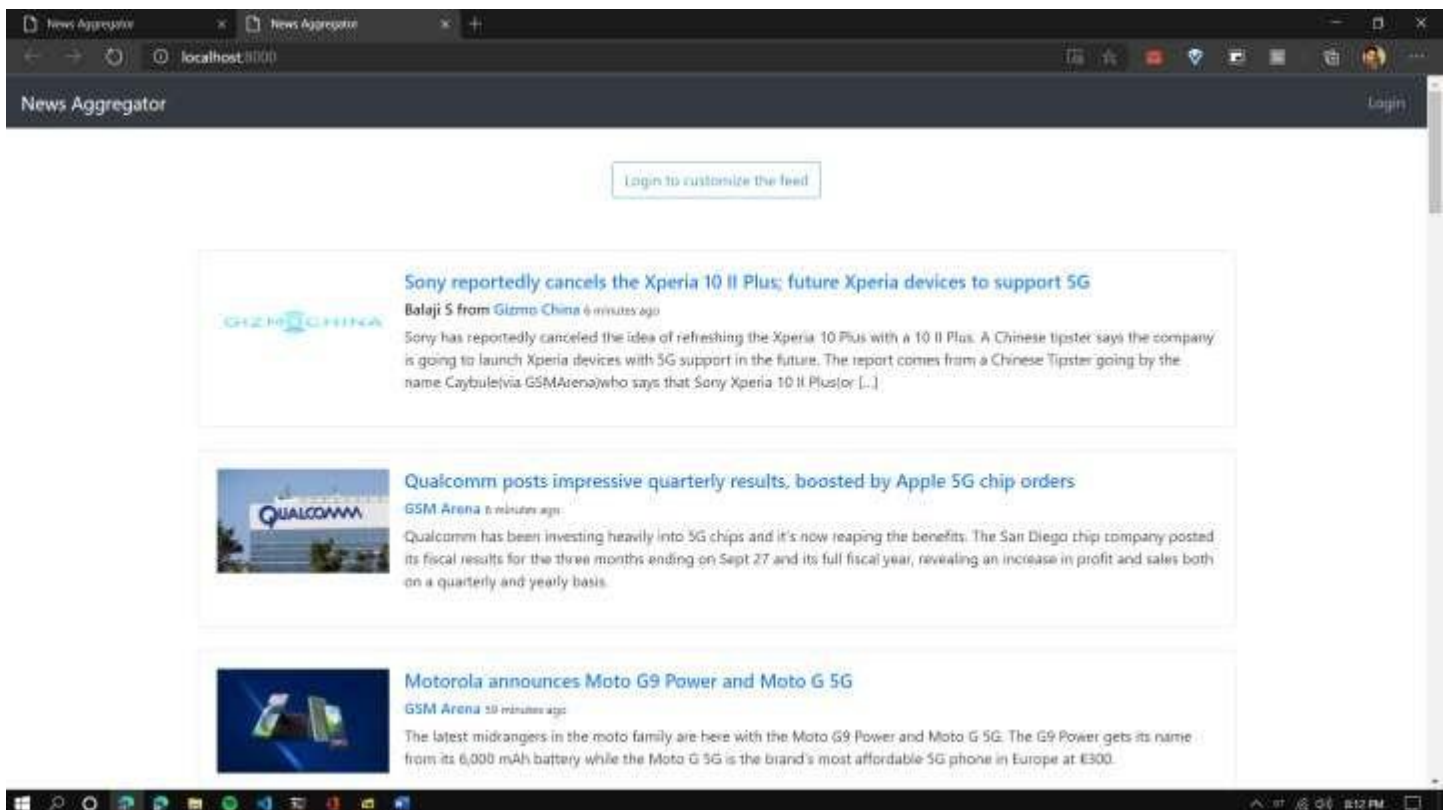
```
{% extends 'base.html' % }  
{% load crispy_forms_tags % }  
{% block content % }  
  
<div style="width:65%; margin: 0 auto;">  
  <h2>Log In | <a href="{% url 'signup' %}">Sign up</a></h2>  
  <form method="post">  
    {% csrf_token % }  
    {{ form|crispy }}  
    <button type="submit" class="btn btn-primary">Log In</button>  
  </form>  
</div>  
{% endblock content % }
```

SCREENSHOTS:

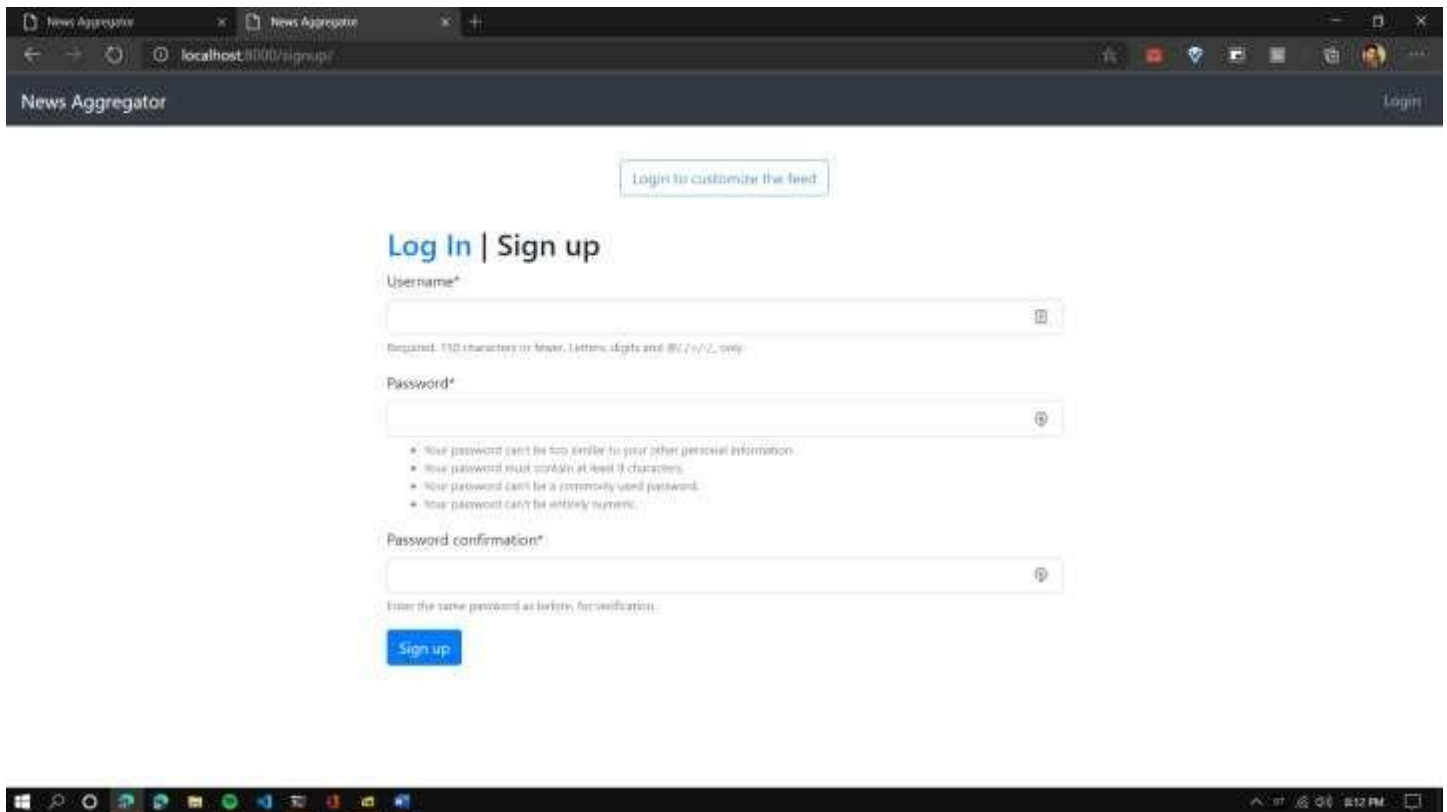
Bot Page:



Homepage (Unsigned):



Signup Page:



The screenshot shows a web browser window with two tabs labeled "News Aggregator". The address bar displays "localhost:8000/signup/". The page header includes the text "News Aggregator" on the left and a "Login" link on the right. A button labeled "Login to customize the feed" is positioned at the top center. Below this, the heading "Log In | Sign up" is displayed, with "Sign up" highlighted in blue. The form contains three input fields: "Username*", "Password*", and "Password confirmation*". The "Password*" field is accompanied by a list of requirements: "New password can't be too similar to your other personal information", "New password must contain at least 8 characters", "New password can't be a commonly used password", and "New password can't be entirely numeric". Below the "Password confirmation*" field, a note states "Enter the same password as before, for verification." A blue "Sign up" button is located at the bottom of the form. The Windows taskbar at the bottom shows the time as 8:12 PM.

News Aggregator

Login

Login to customize the feed

Log In | Sign up

Username*

Required: 150 characters or fewer, letters, digits and @/./+/_/~/ only

Password*

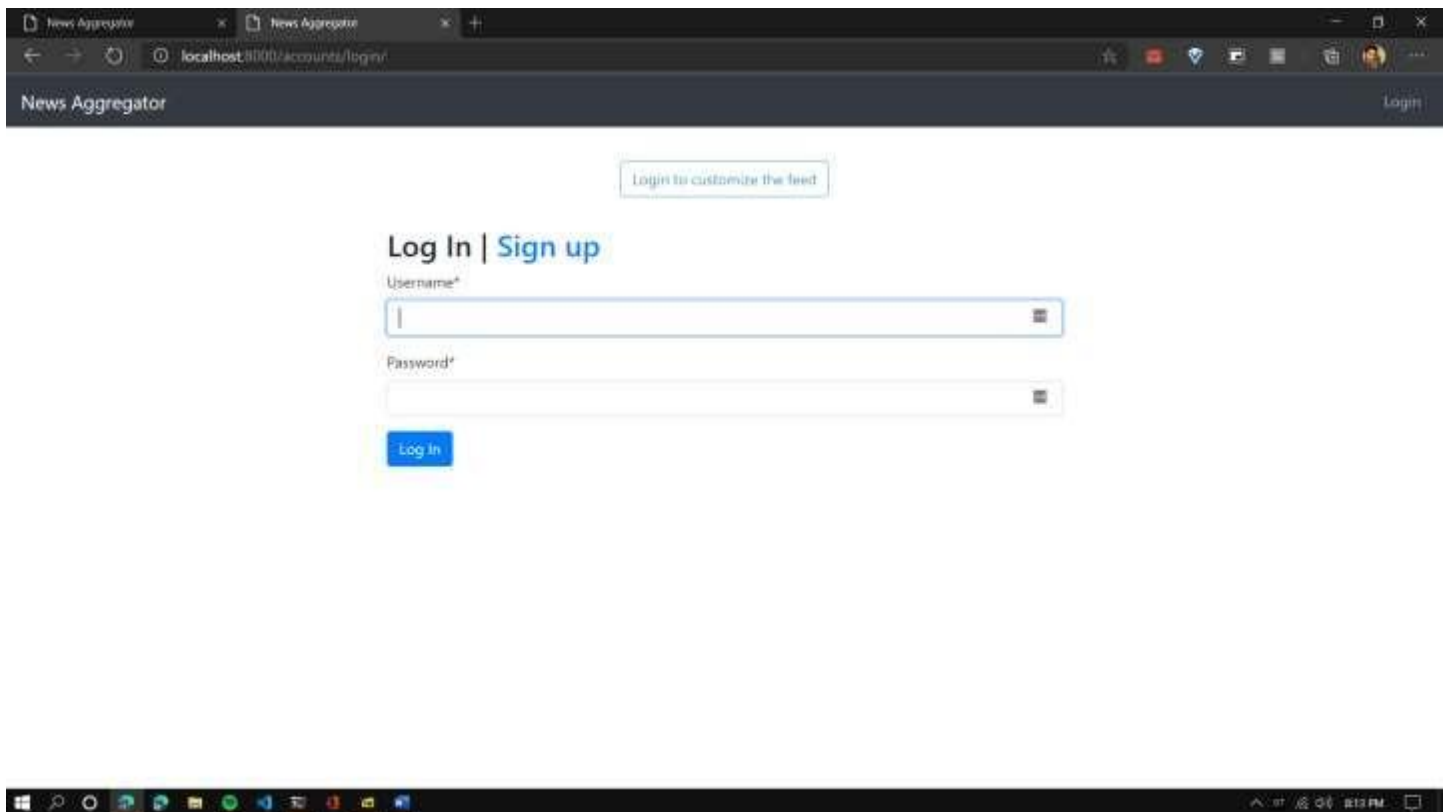
- New password can't be too similar to your other personal information.
- New password must contain at least 8 characters.
- New password can't be a commonly used password.
- New password can't be entirely numeric.

Password confirmation*

Enter the same password as before, for verification.

Sign up

Login Page:



The screenshot shows a web browser window with two tabs labeled "News Aggregator". The address bar displays "localhost:8000/accounts/login/". The page header includes the text "News Aggregator" on the left and a "Login" link on the right. A button labeled "Login to customize the feed" is positioned at the top center. Below this, the heading "Log In | Sign up" is displayed, with "Log In" highlighted in blue. The form contains two input fields: "Username*" and "Password*". A blue "Log In" button is located at the bottom of the form. The Windows taskbar at the bottom shows the time as 8:13 PM.

News Aggregator

Login

Login to customize the feed

Log In | Sign up

Username*

Password*

Log In

Homepage (Signed In):

The screenshot shows a web browser window with the URL `localhost:8000`. The page title is "News Aggregator" and there is a "Profile" link in the top right corner. The main content area displays a list of news articles, each with a thumbnail image, a title, a source, and a timestamp.

Thumbnail	Title	Source	Timestamp
	Sony reportedly cancels the Xperia 10 II Plus; future Xperia devices to support 5G	Balaji S from Gizmo China	5 minutes ago
	Qualcomm posts impressive quarterly results, boosted by Apple 5G chip orders	GSM Arena	7 minutes ago
	Motorola announces Moto G9 Power and Moto G 5G	GSM Arena	an hour ago
	MIUI 12 update for Redmi Note 8 global variant is now available for all users	Sudarsan from Gizmo China	an hour ago

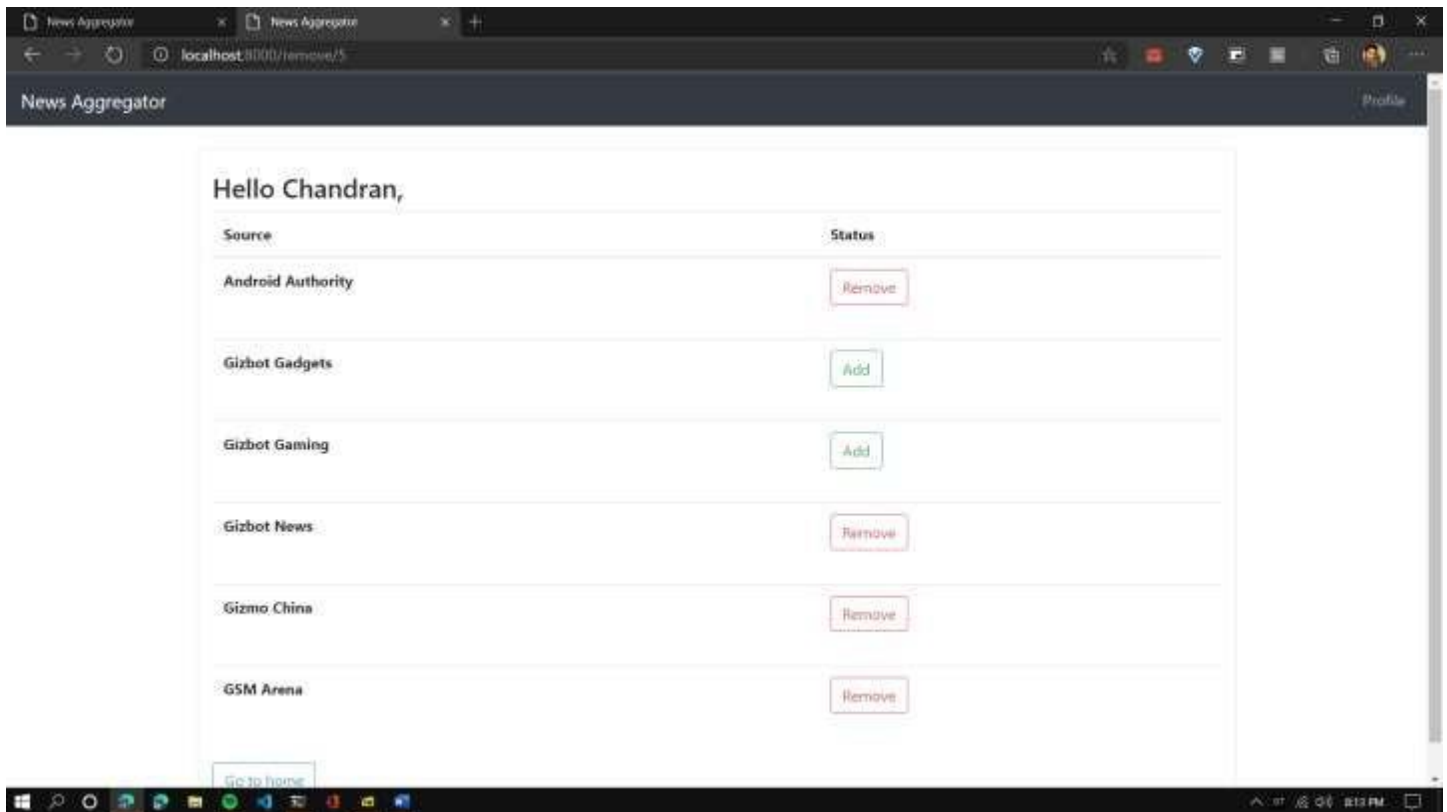
Profile Page:

The screenshot shows a web browser window with the URL `localhost:8000/profile/`. The page title is "News Aggregator" and there is a "Logout" link in the top right corner. The main content area displays a list of sources and their status, with a "Go to home" link at the bottom left.

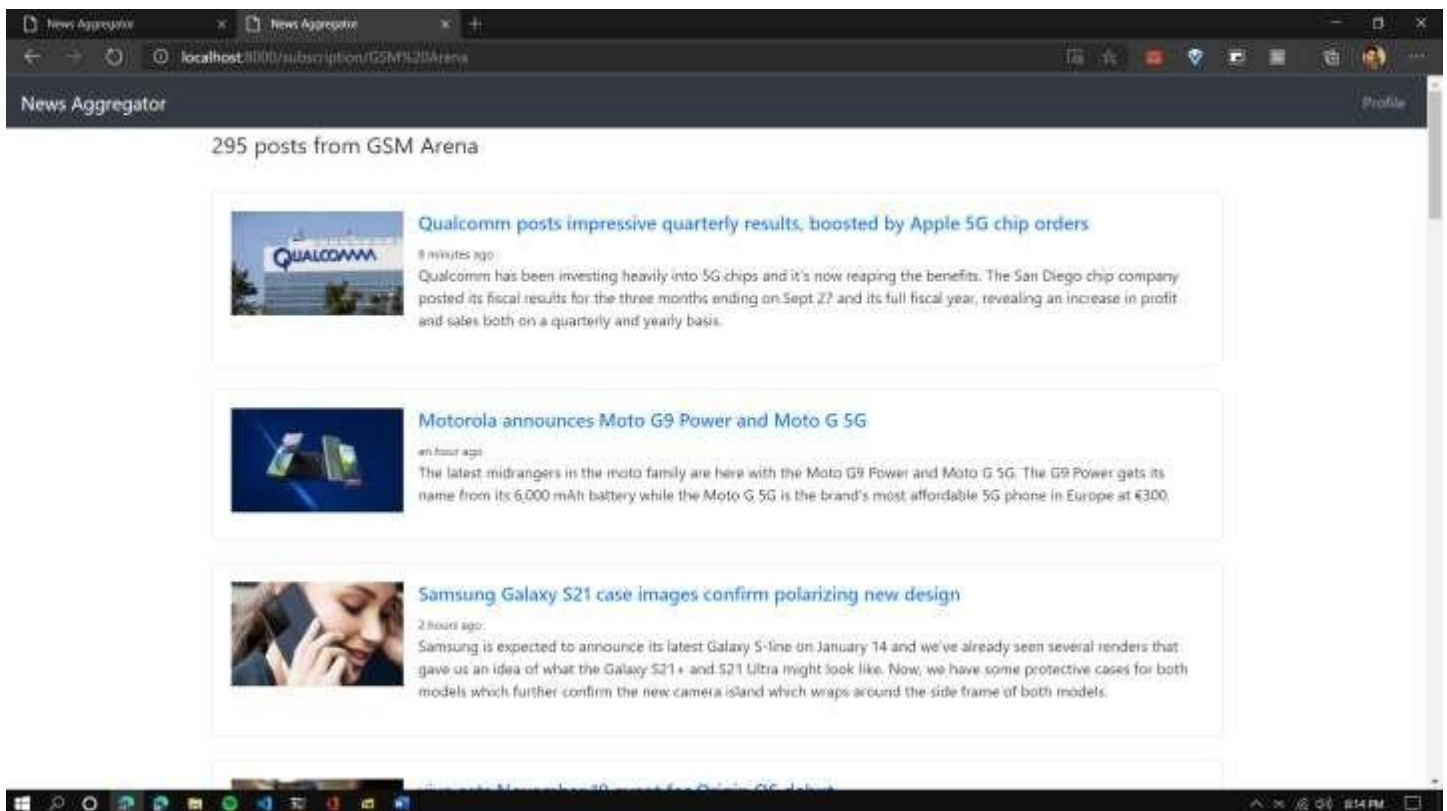
Source	Status
Android Authority	Remove
Gizbot Gadgets	Remove
Gizbot Gaming	Remove
Gizbot News	Remove
Gizmo China	Remove
GSM Arena	Remove

[Go to home](#)

Profile Page (Modified):



Source Page:



Original Article:



CONCLUSION:

By merging multiple sources of articles into a single platform, time and resource gets saved and improves the productivity of the user. Without this, the user feels lost in a jungle, but this provides more reachability to many articles and introduces the user to many sources of news.

RESULT:

Thus, the project has been completed successfully with all the features and modules of the online voting system as per the requirements.

