Hacker News Scraper: Extracting and Analyzing Top Stories

Introduction

Hacker News, a popular technology and startup-focused news aggregator, is a valuable source of trending discussions in the tech world. This project involves building a web scraper using Python and BeautifulSoup to extract and analyze the most upvoted stories from the Hacker News homepage.

The scraper collects key information, including:

Article Titles :

The headline of each news story.

Article Links :

Direct URLs to the original articles.

Upvotes :

The number of upvotes each story has received. Once the data is extracted, it is structured into a pandas DataFrame, allowing for easy sorting and analysis. The dataset is then sorted in descending order based on upvotes to highlight the most popular and engaging stories.

Objectives

Web Scraping :

Use BeautifulSoup to extract article titles, links, and upvotes from the Hacker News website.

Data Processing :

Organize the extracted data into a structured format using pandas.

Sorting and Analysis :

Rank articles based on their popularity (upvotes).

Insights Generation :

Identify trending topics and high-engagement news articles.

Tech Stack

Python :

The core programming language for scripting.

BeautifulSoup :

For HTML parsing and web scraping.

Requests :

To fetch data from the Hacker News website.

Pandas :

For data manipulation and analysis.

Potential Use Cases

- Identifying the most trending news in tech and startups.
- Automating news aggregation for daily insights.
- Tracking engagement patterns in online discussions.
- This project demonstrates the power of web scraping and data analysis in extracting valuable insights from online news sources.
- Future enhancements could include automating the script to run at scheduled intervals or visualizing trends over time using data visualization tools.

In [15]:
```
from bs4 import BeautifulSoup
import lxml
import requests
import pandas as pd
```

```
response = requests.get("https://news.ycombinator.com/news")
yc_web_page = response.text
soup = BeautifulSoup(yc_web_page, "html.parser")
all_anchor_tags = soup.find_all(name="a")
# print(all_anchor_tags)
# for anchor in all_anchor_tags:
#     print(anchor.getText)
#     print(anchor['href'])

section_anchors= soup.find_all(name="span", class_="titleline")
section_link = soup.find(name="a")["href"]
section_links = [link["href"] for link in soup.find_all(name="a") if "href" in link.attrs]

article_texts = []
article_links = []
for anchor in section_anchors:
    section_titles = anchor.getText()
    article_texts.append(section_titles)


for link in section_links:
    if link.startswith("http"):
        article_links.append(link)

article_upvotes = [int(score.getText().split()[0]) for score in soup.find_all(name="span", class_="score")]

print(article_texts)
print(article_links)
print(article_upvotes)

# Create DataFrame
df = pd.DataFrame({
    "Title": article_texts[:len(article_upvotes)],  # Ensure matching lengths
    "Link": article_links[:len(article_upvotes)],
    "Upvotes": article_upvotes
})

print(df)

# Sort DataFrame by upvotes in descending order
df_sorted = df.sort_values(by="Upvotes", ascending=False)

# Print or save the sorted DataFrame
#print(df_sorted)
```

```
['Linux Running in a PDF (doompdf.dev)', 'Ingesting PDFs and why Gemini 2.0 changes everything (sergey.fyi)', 'P
rogramming SDF Animations of Rick and Morty (danielchasehooper.com)', 'OpenWrt 24.10.0 – First Stable Release (o
penwrt.org)', 'FreeBSD for hi-fi audio: real-time processing, equalizer, MPD and FFmpeg (m4c.pl)', 'Paper Apps (
gladdendesign.com)', 'Okta Bcrypt incident lessons for designing better APIs (n0rdy.foo)', 'Memfault (YC W19) Is
Hiring an Android System (AOSP) Engineer (lever.co)', 'I believe 6502 instruction set is a good first assembly l
anguage (nemanjatrifunovic.substack.com)', 'Deep Reinforcement Learning: Pong from Pixels (2016) (karpathy.githu
b.io)', 'Minimum effective dose (winnielim.org)', 'S1: A $6 R1 competitor? (timkellogg.me)', "Servo's progress i
n 2024 (servo.org)", 'Cameras of 1930s Era (licm.org.uk)', 'Tell HN: Cloudflare is blocking Pale Moon and other
non-mainstream browsers', 'The missing cross-platform OS API for timers (gaultier.github.io)', 'Discord client t
hat works on Win95*, Win98 and above (github.com/discordmessenger)', 'Cloudflare R2 Global Outage (cloudflaresta
tus.com)', 'I wrote a screenplay for a programming language introduction (miksovsky.com)', 'Par: Process languag
e with an interactive playground for exploring concurrency (github.com/faiface)', 'Excavated: 52 Egyptian Mummie
s. Over a Dozen Had Mysterious Golden Tongues (popularmechanics.com)', 'Why is Warner Bros. Discovery putting ol
d movies on YouTube? (tedium.co)', 'The Language Construction Kit (1996, 2012) (zompist.com)', 'Avoiding outrage
fatigue while staying informed (scientificamerican.com)', 'Show HN: Marksmith – a GitHub-style Markdown editor f
or Ruby on Rails (avohq.io)', 'Physics Rediscovered #6: My trebuchet is bigger than yours (michaeldominik.substa
ck.com)', "Software development topics I've changed my mind on (chriskiehl.com)", 'GNU Make Standard Library (jg
c.org)', 'Mystery brain disease patients in New Brunswick say they welcome investigation (ctvnews.ca)', "Markdow
n's Big Brother: Say Hello to AsciiDoc (git-tower.com)"]
['https://news.ycombinator.com', 'https://linux.doompdf.dev/linux.pdf', 'https://www.sergey.fyi/articles/gemini-
flash-2', 'https://danielchasehooper.com/posts/code-animated-rick/', 'https://openwrt.org/releases/24.10/notes-2
4.10.0', 'https://m4c.pl/blog/freebsd-audio-setup-bitperfect-equalizer-realtime/', 'https://gladdendesign.com/co
llections/paper-apps', 'https://n0rdy.foo/posts/20250121/okta-bcrypt-lessons-for-better-apis/', 'https://jobs.le
ver.co/memfault/1904a421-de92-46bf-8864-2965582cd6df', 'https://nemanjatrifunovic.substack.com/p/6502-is-a-good-
starting-point-for', 'http://karpathy.github.io/2016/05/31/rl/', 'https://winnielim.org/journal/minimum-effectiv
e-dose/', 'https://timkellogg.me/blog/2025/02/03/s1', 'https://servo.org/blog/2025/01/31/servo-in-2024/', 'https
://licm.org.uk/livingImage/1930Room.html', 'https://gaultier.github.io/blog/the_missing_cross_platform_os_api_fo
r_timers.html', 'https://github.com/DiscordMessenger/dm', 'https://www.cloudflarestatus.com', 'https://jan.miksо
vsky.com/', 'https://github.com/faiface/par-lang', 'https://www.popularmechanics.com/science/archaeology/a634120
49/golden-mummy-tongues/', 'https://tedium.co/2025/02/05/warner-bros-youtube-full-movie-releases/', 'https://www
.zompist.com/kit.html', 'https://www.scientificamerican.com/podcast/episode/how-to-avoid-outrage-fatigue-and-tun
e-in-without-burning-out/', 'https://avohq.io/blog/ruby-on-rails-markdown-editor-marksmith', 'https://michaeldom
inik.substack.com/p/physics-rediscovered-interlude-my', 'https://chriskiehl.com/article/thoughts-after-10-years'
, 'https://gmsl.jgc.org/', 'https://www.ctvnews.ca/atlantic/new-brunswick/article/good-first-step-nb-mystery-bra
in-disease-patients-welcome-new-investigation/', 'https://www.git-tower.com/blog/asciidoc-quick-guide/', 'https:
//www.ycombinator.com/apply/', 'https://github.com/HackerNews/API', 'https://www.ycombinator.com/legal/', 'https
://www.ycombinator.com/apply/']
[231, 1042, 231, 247, 74, 40, 298, 140, 101, 222, 735, 416, 30, 983, 10, 201, 113, 35, 3, 15, 527, 85, 623, 75,
14, 734, 139, 101, 69]
```

In [3]:
```
df_sorted
```

Out[3]:

	Title	Link	Upvotes
1	Ingesting PDFs and why Gemini 2.0 changes ever...	https://linux.doompdf.dev/linux.pdf	1012
12	Cameras of 1930s Era (licm.org.uk)	https://servo.org/blog/2025/01/31/servo-in-2024/	951
24	Paper Apps (gladdendesign.com)	https://gladdendesign.com/collections/paper-apps	722
9	I believe 6502 instruction set is a good first...	https://winnielim.org/journal/minimum-effectiv...	720
18	Physics Rediscovered #6: My trebuchet is bigge...	https://michaeldominik.substack.com/p/physics-...	604
21	Show HN: Marksmith – a GitHub-style Markdown e...	https://avohq.io/blog/ruby-on-rails-markdown-e...	517
10	S1: A $6 R1 competitor? (timkellogg.me)	https://nemanjatrifunovic.substack.com/p/6502-...	408
4	Okta Bcrypt incident lessons for designing bet...	https://danielchasehooper.com/posts/code-anima...	283
2	OpenWrt 24.10.0 – First Stable Release (openwr...	https://www.sergey.fyi/articles/gemini-flash-2	225
7	Memfault (YC W19) Is Hiring an Android System ...	http://karpathy.github.io/2016/05/31/rl/	208
3	Programming SDF Animations of Rick and Morty (...	https://openwrt.org/releases/24.10/notes-24.10.0	197
15	The missing cross-platform OS API for timers (...	https://gaultier.github.io/blog/the_missing_cr...	193
0	Linux Running in a PDF (doompdf.dev)	https://news.ycombinator.com	181
26	Origami by Meenakshi (origamee.net)	https://origamee.net/	134
8	Minimum effective dose (winnielim.org)	https://jobs.lever.co/memfault/1904a421-de92-4...	124
13	Tell HN: Cloudflare is blocking Pale Moon and ...	https://licm.org.uk/livingImage/1930Room.html	99
27	GNU Make Standard Library (jgc.org)	https://gmsl.jgc.org/	93
6	Deep Reinforcement Learning: Pong from Pixels ...	https://m4c.pl/blog/freebsd-audio-setup-bitper...	90
19	Avoiding outrage fatigue while staying informe...	https://www.scientificamerican.com/podcast/epi...	80
20	The Language Construction Kit (1996, 2012) (zo...	https://www.zompist.com/kit.html	69
28	Mystery brain disease patients in New Brunswic...	https://www.ctvnews.ca/atlantic/new-brunswick/...	59
5	FreeBSD for hi-fi audio: real-time processing,...	https://n0rdy.foo/posts/20250121/okta-bcrypt-l...	55
16	Discord client that works on Win95*, Win98 and...	https://github.com/DiscordMessenger/dm	27
11	Servo's progress in 2024 (servo.org)	https://timkellogg.me/blog/2025/02/03/s1	24
17	I wrote a screenplay for a programming languag...	https://jan.miksovsky.com/	11
22	Why is Warner Bros. Discovery putting old movi...	https://tedium.co/2025/02/05/warner-bros-youtu...	9
14	Cloudflare R2 Global Outage (cloudflarestatus....	https://www.cloudflarestatus.com	7
23	Excavated: 52 Egyptian Mummies. Over a Dozen H...	https://www.popularmechanics.com/science/archa...	6
25	Software development topics I've changed my mi...	https://chriskiehl.com/article/thoughts-after-...	6

In [4]:
```
pip install schedule
```

```
Requirement already satisfied: schedule in c:\users\collins pc\anaconda3\envs\collonel\lib\site-packages (1.2.2)
Note: you may need to restart the kernel to use updated packages.
```

In [5]:
```
import schedule
```

In [6]:
```
 # Save to CSV for tracking trends over time
df_sorted.to_csv("hacker_news_trends.csv", index=False)

print("Data Scraped & Saved Successfully ✅")
```

```
Data Scraped & Saved Successfully ✅
```

Hacker News Scraper: Automation

This project automates the process of scraping Hacker News to track trending stories and visualize their popularity based on upvotes. The script runs on a schedule and performs the following key tasks:

Scrape Hacker News:

The script fetches the Hacker News homepage and extracts article titles, links, and upvotes.

The data is parsed using BeautifulSoup and cleaned to ensure that all lists (titles, links, upvotes) are aligned.

Data Processing:

The extracted data is organized into a Pandas DataFrame, where each row represents an article with its title, link, and upvote count. The DataFrame is then sorted by upvotes in descending order to highlight the most popular articles. Data Storage:

The sorted data is saved into a CSV file (hacker_news_trends.csv) for future analysis, allowing you to track trends over time.

Visualization:

The script generates a bar chart of the top 10 trending stories based on upvotes, using Matplotlib. The bar chart is saved as an image file (hacker_news_top_trending.png) and displayed for immediate insights.

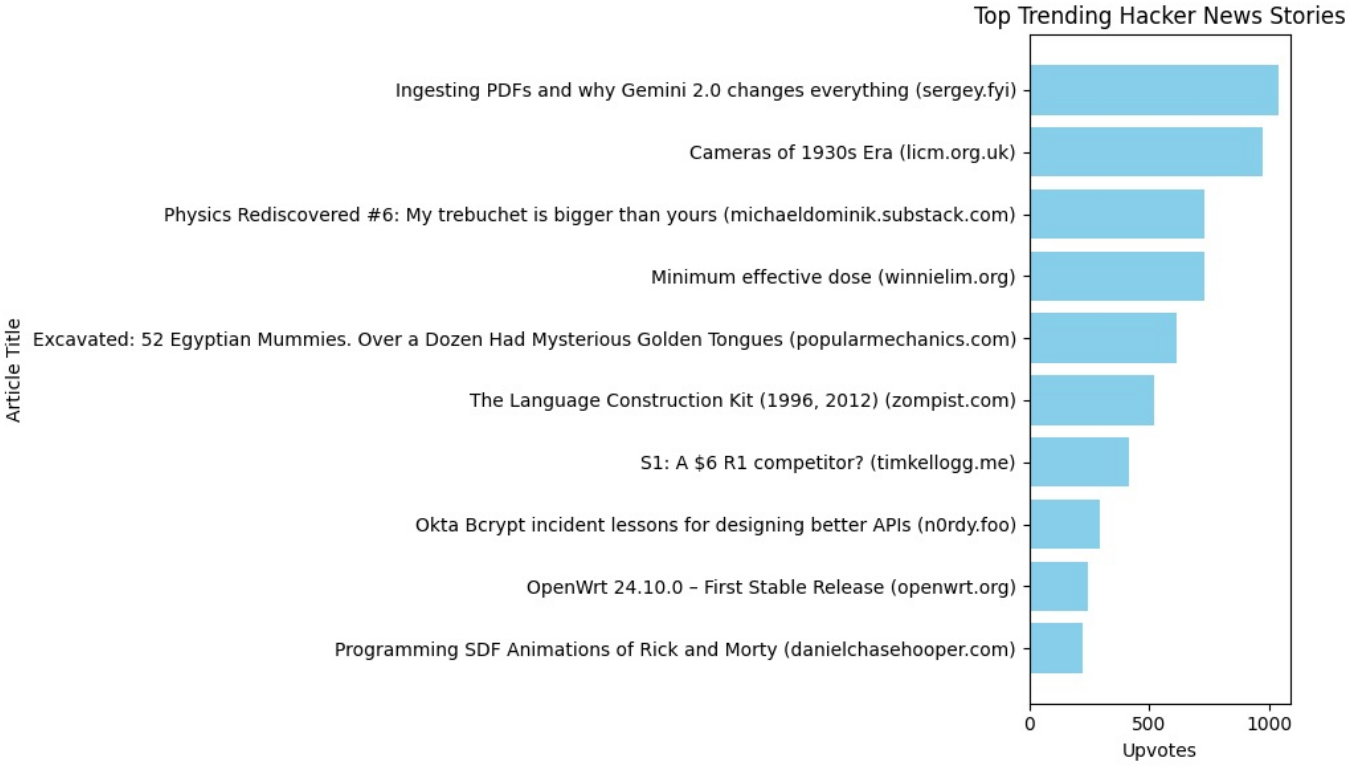
Automation:

The script is automated using the schedule library, running every 0.5 hours (30 minutes), ensuring that the data is up-to-date and trends are captured in real-time. The script operates in a continuous loop, scraping the data and generating updated visualizations at regular intervals. This project can be extended to track trends in other domains or websites, making it a powerful tool for monitoring and visualizing real-time data.

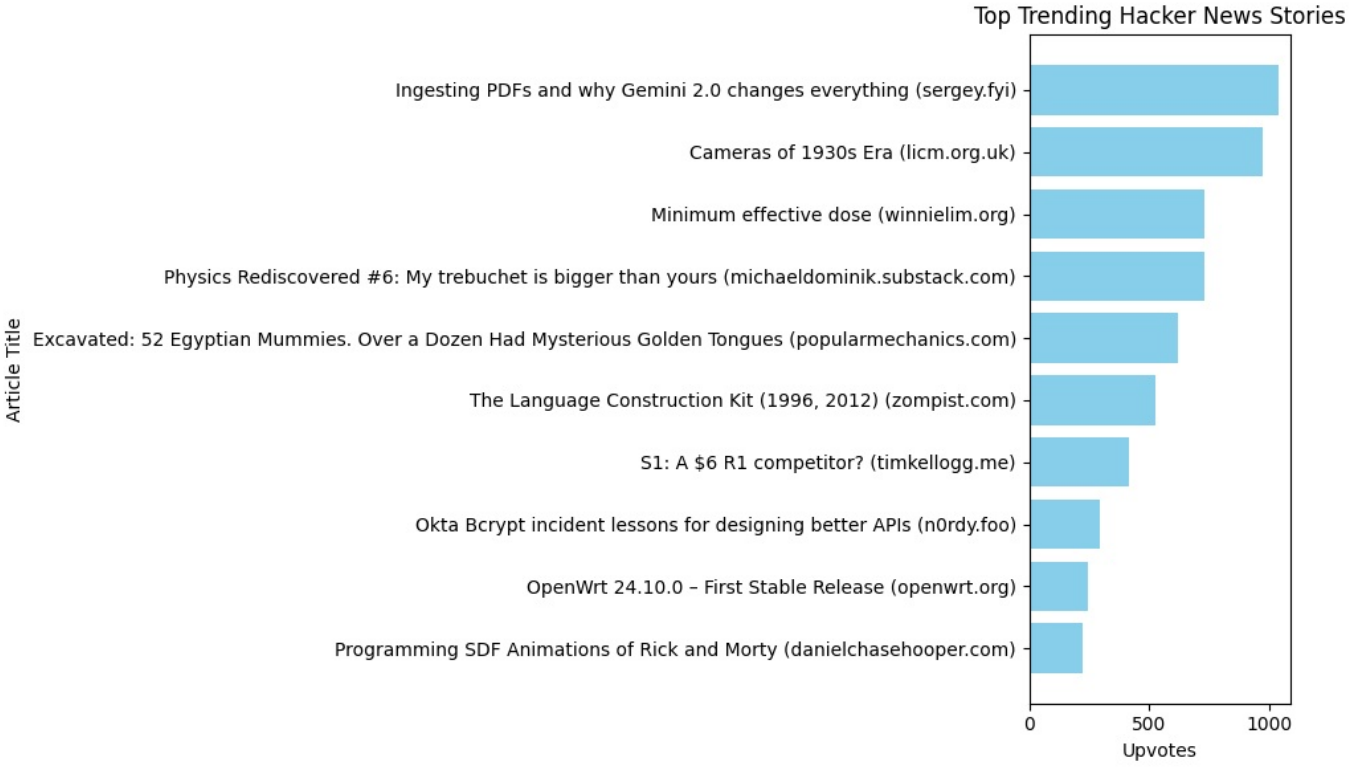
In [12]:
```
from datetime import datetime
import time
import matplotlib.pyplot as plt
def scrape_hacker_news():
    print("Scraping Hacker News...")

    # Fetch Hacker News homepage
    response = requests.get("https://news.ycombinator.com/news")
    yc_web_page = response.text

    # Parse HTML with BeautifulSoup
    soup = BeautifulSoup(yc_web_page, "html.parser")

    # Extract article titles and links
    section_anchors = soup.find_all(name="span", class_="titleline")
    article_texts = [anchor.getText() for anchor in section_anchors]
    article_links = [anchor.find("a")["href"] for anchor in section_anchors]

    # Extract upvotes
    article_upvotes = [
        int(score.getText().split()[0])
        for score in soup.find_all(name="span", class_="score")
    ]

    # Ensure lists have the same length
    min_length = min(len(article_texts), len(article_links), len(article_upvotes))
    article_texts = article_texts[:min_length]
    article_links = article_links[:min_length]
    article_upvotes = article_upvotes[:min_length]

    # Create a DataFrame
    df = pd.DataFrame({
        "Title": article_texts,
        "Link": article_links,
        "Upvotes": article_upvotes
    })

    # Sort DataFrame by upvotes in descending order
    df_sorted = df.sort_values(by="Upvotes", ascending=False)

    # Save to CSV for tracking trends over time
    df_sorted.to_csv("hacker_news_trends.csv", index=False)

    print("Data Scraped & Saved Successfully ✅")

    # Generate visualization
    visualize_top_stories(df_sorted)


def visualize_top_stories(df):
    """ Generate a bar chart of top 10 stories based on upvotes """
    top_n = 10  # Show top 10 stories
    df_top = df.head(top_n)

    plt.figure(figsize=(10, 6))
    plt.barh(df_top["Title"], df_top["Upvotes"], color="skyblue")
    plt.xlabel("Upvotes")
    plt.ylabel("Article Title")
    plt.title("Top Trending Hacker News Stories")
    plt.gca().invert_yaxis()  # Invert y-axis to show highest upvotes on top
    plt.tight_layout()

    # Save figure
    plt.savefig("hacker_news_top_trending.png")
    plt.show()

    print("Visualization Generated & Saved ")
```

```python
# **Schedule the script to run every 0.5 hours**
schedule.every(2).minutes.do(scrape_hacker_news)

# Run the scheduler in a loop
print("Starting Automated Scraper ")
iterations = 0
max_iterations = 2

while iterations < max_iterations:
    schedule.run_pending()
    time.sleep(1)
    iterations += 1

print("Script Completed After Two Iterations. ✅")
```

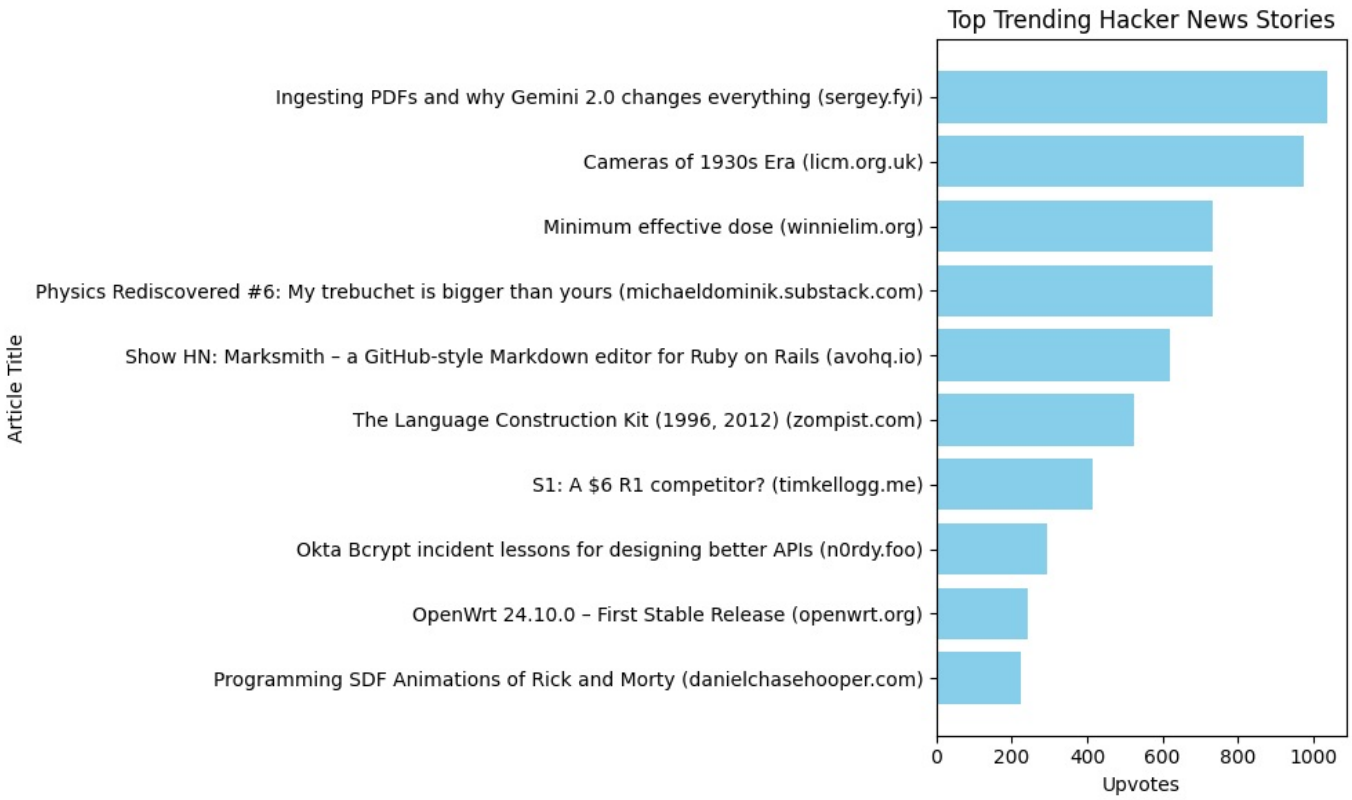
```
Starting Automated Scraper
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```



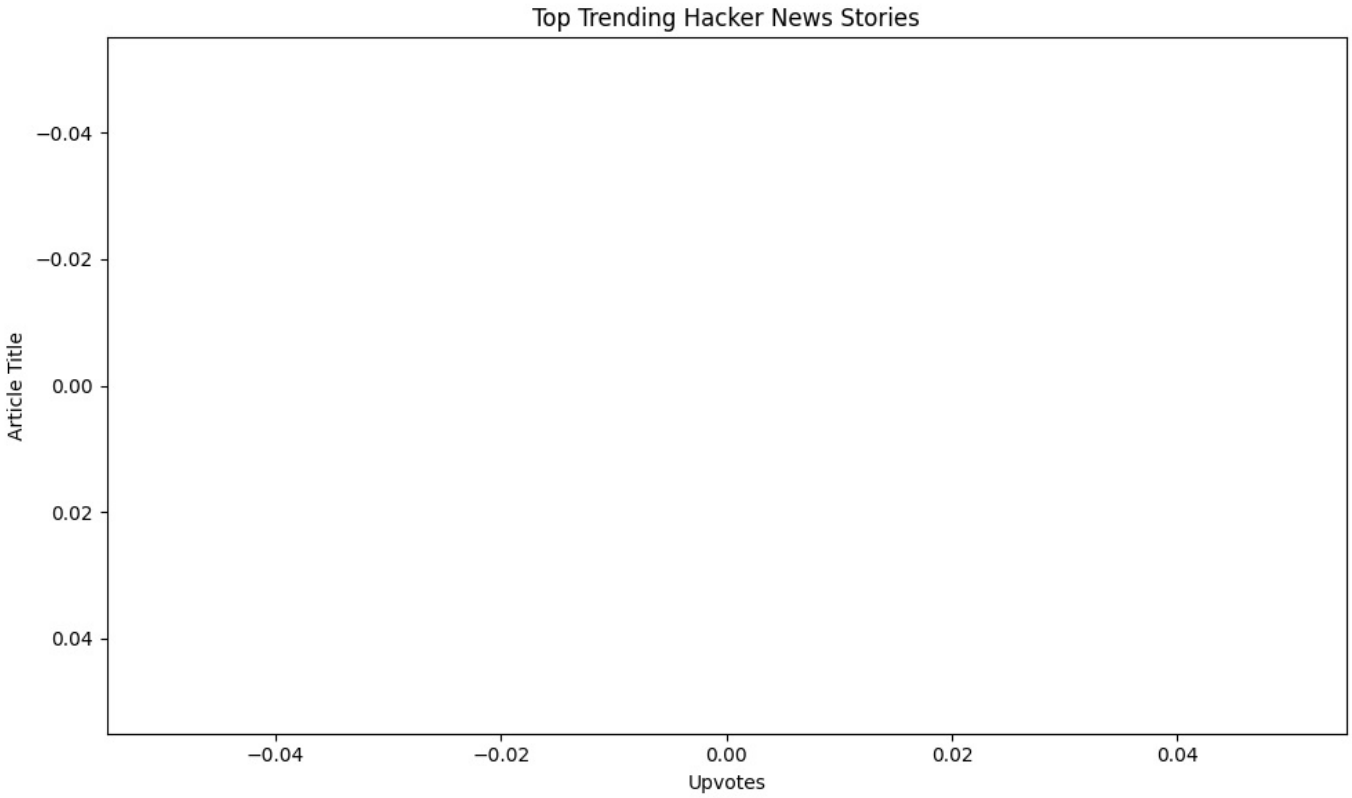

```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```



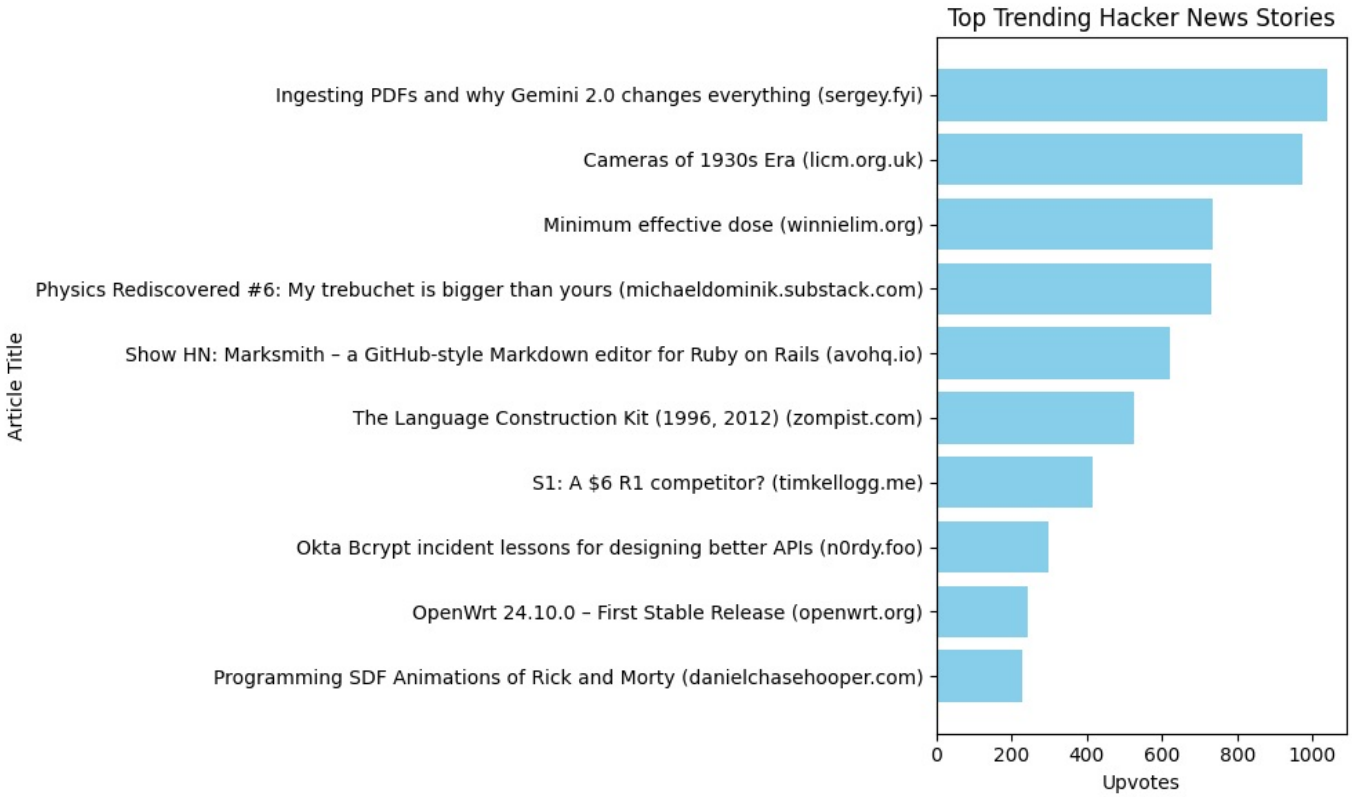
```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```



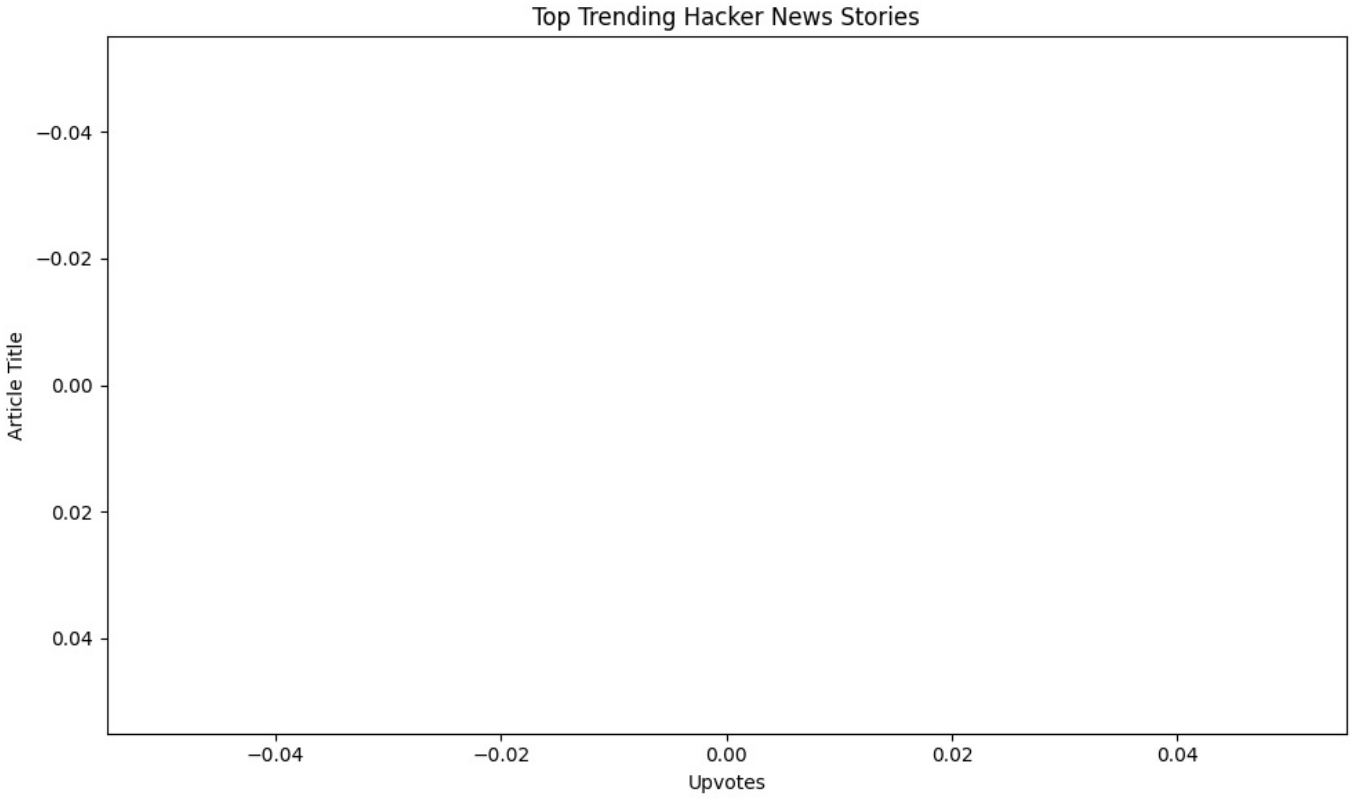

```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```



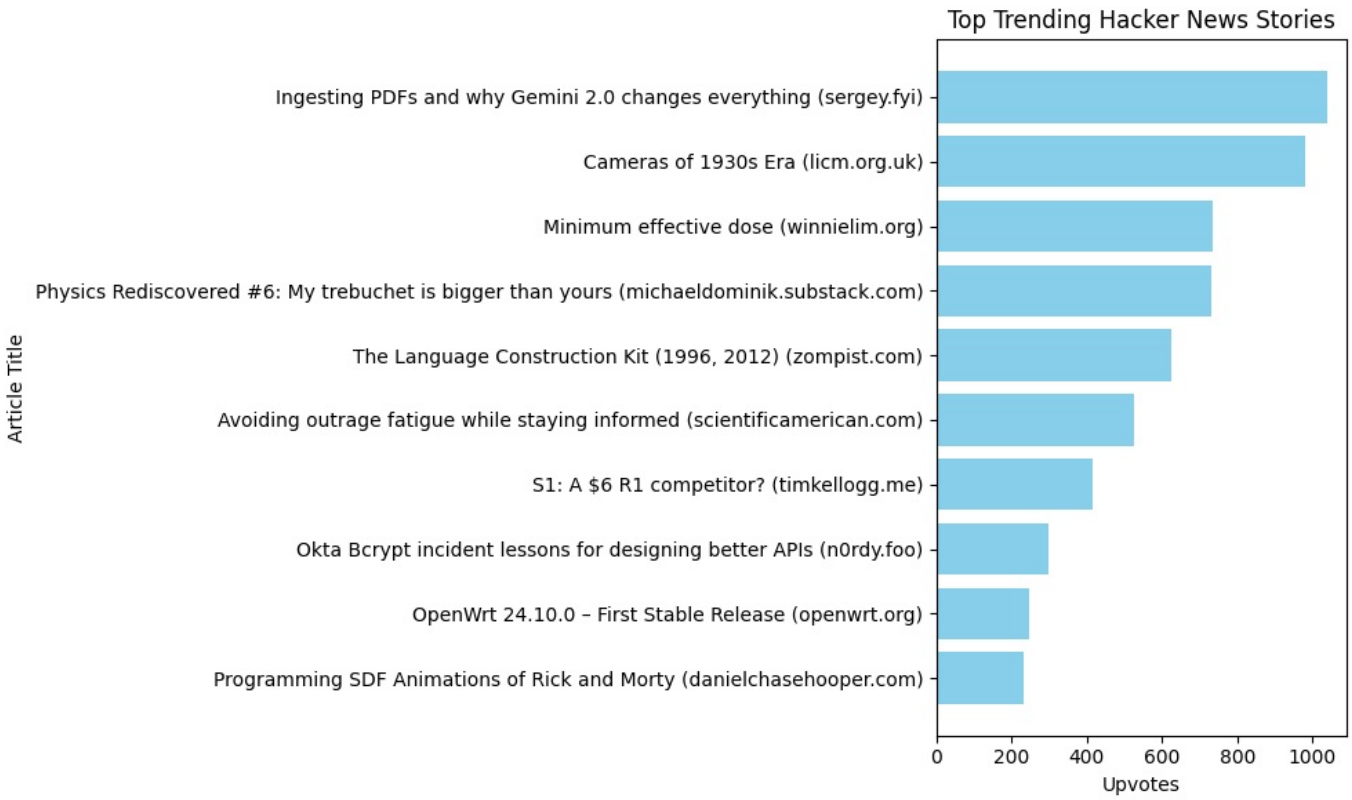

```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```



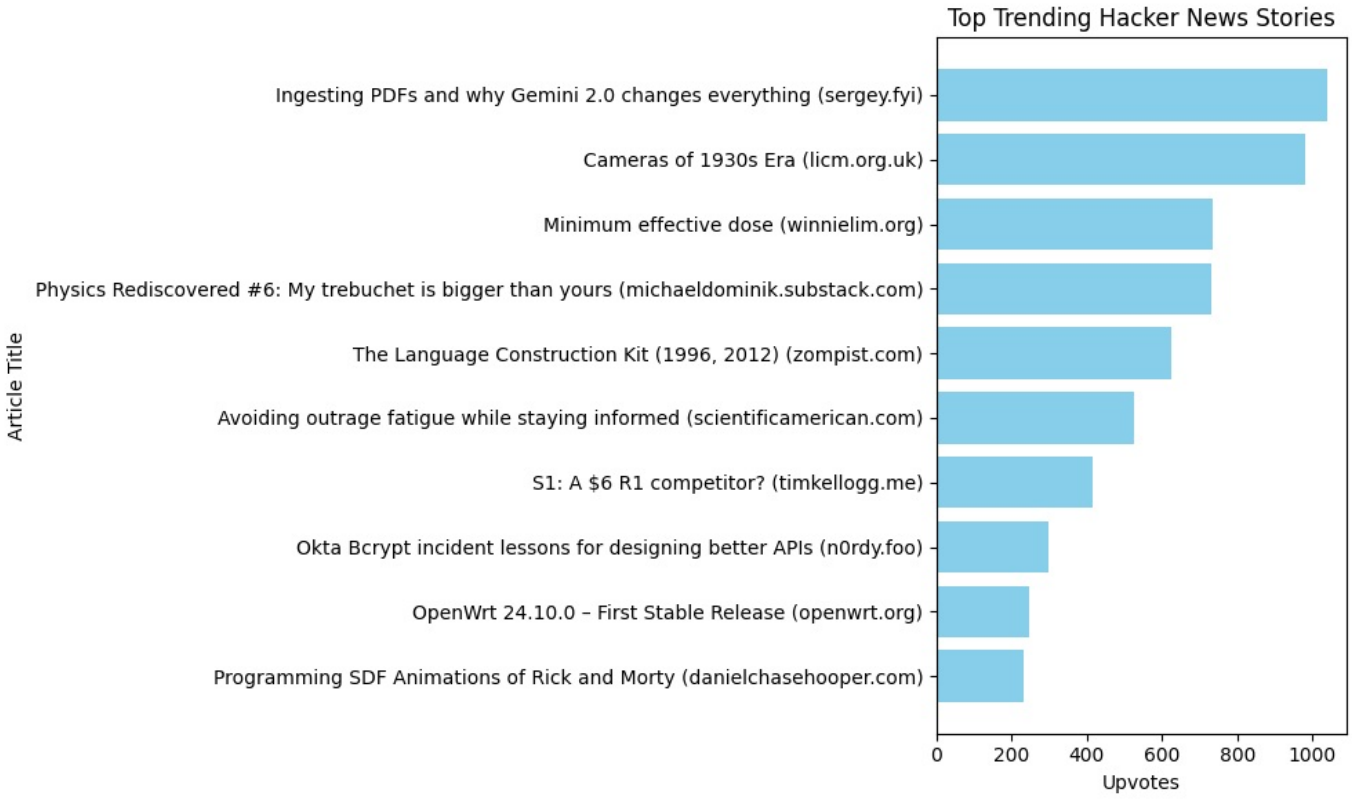

```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```



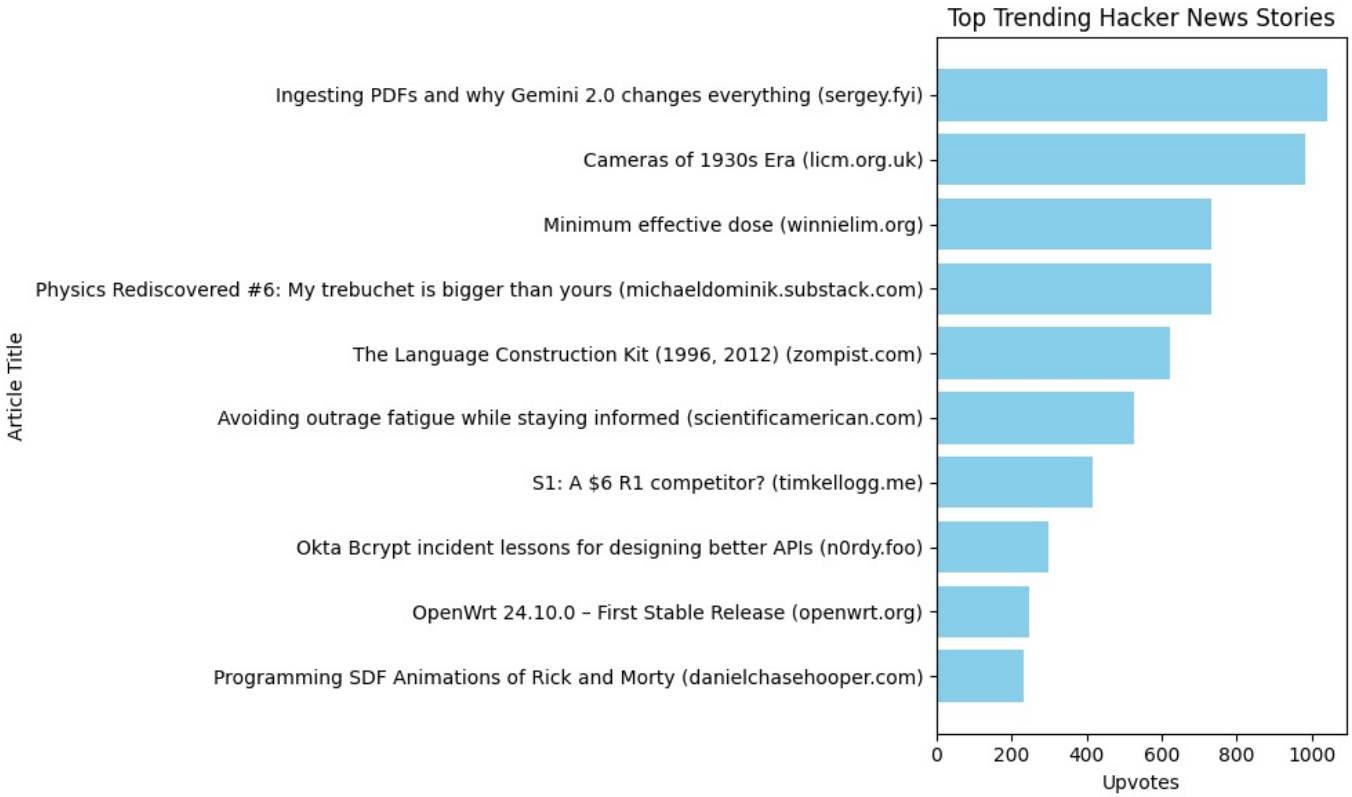

```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```




```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```




```
Visualization Generated & Saved
Scraping Hacker News...
Data Scraped & Saved Successfully ✅
```




```
Visualization Generated & Saved
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[12], line 76
     74 while True:
     75     schedule.run_pending()
---> 76     time.sleep(1)

KeyboardInterrupt:
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: