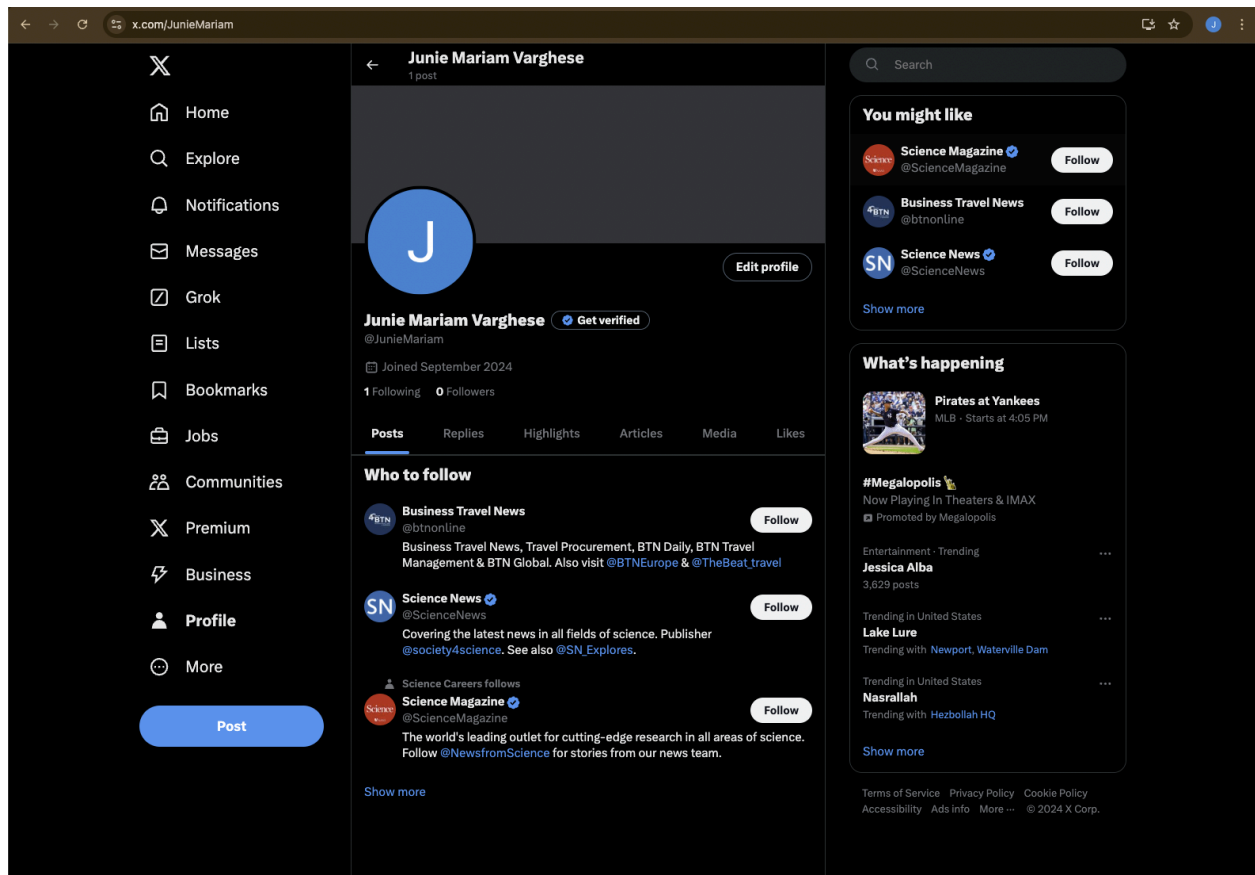
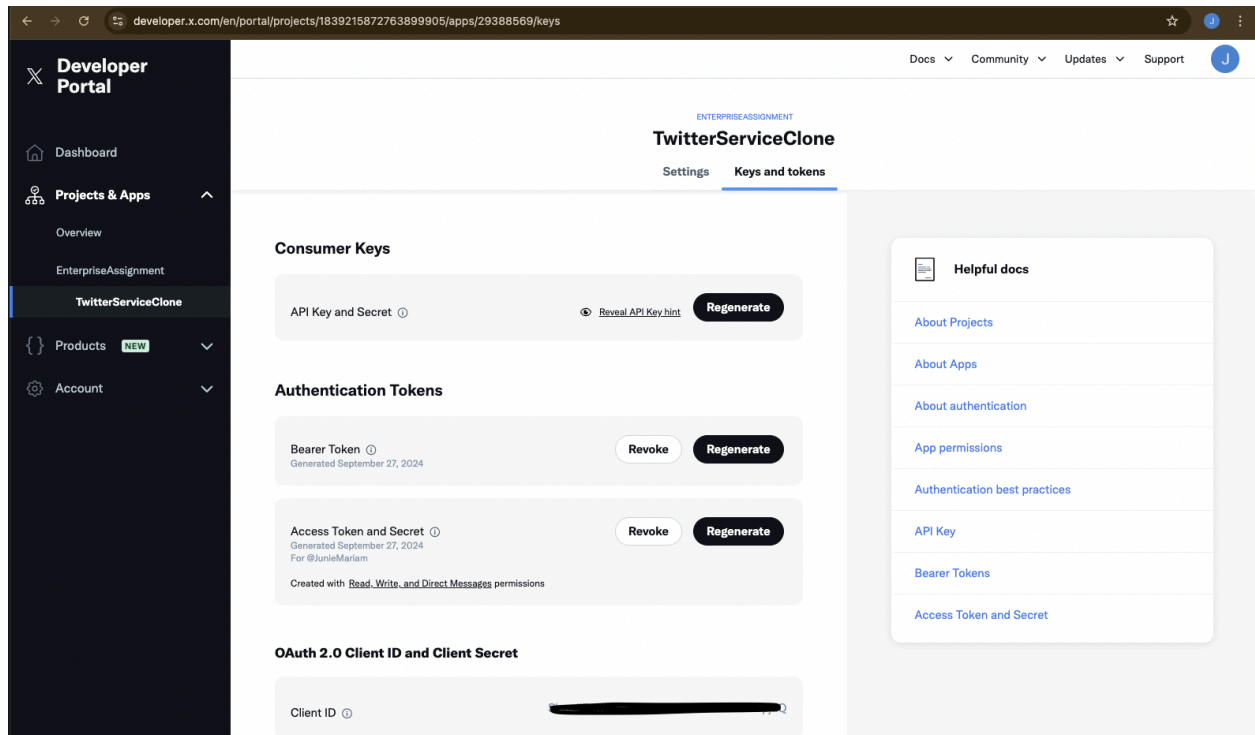


TWITTER SERVICE

1. Created a twitter account



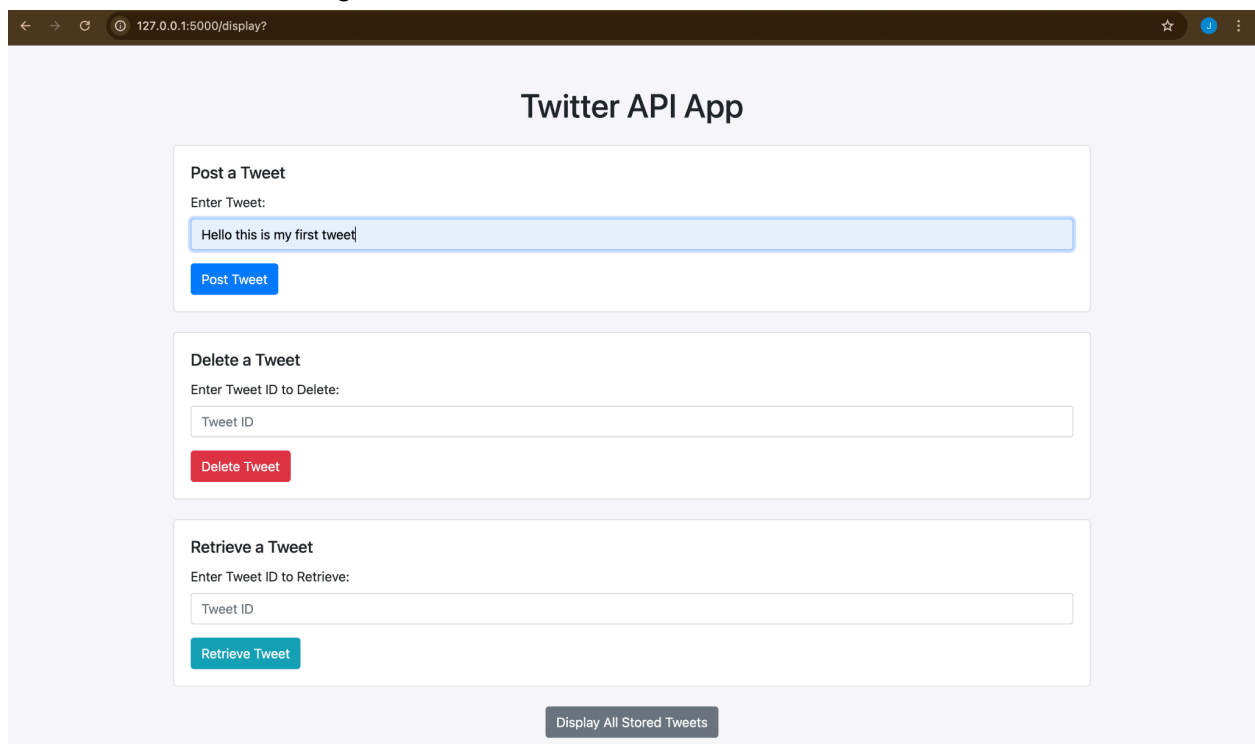
2. Opened developer portal created app and got the secrets for accessing the APIs. Since we are using a free version. We have access to the Post and Delete API. Since Retrieve API is paid we used an SQLite database to store the details of the tweet.

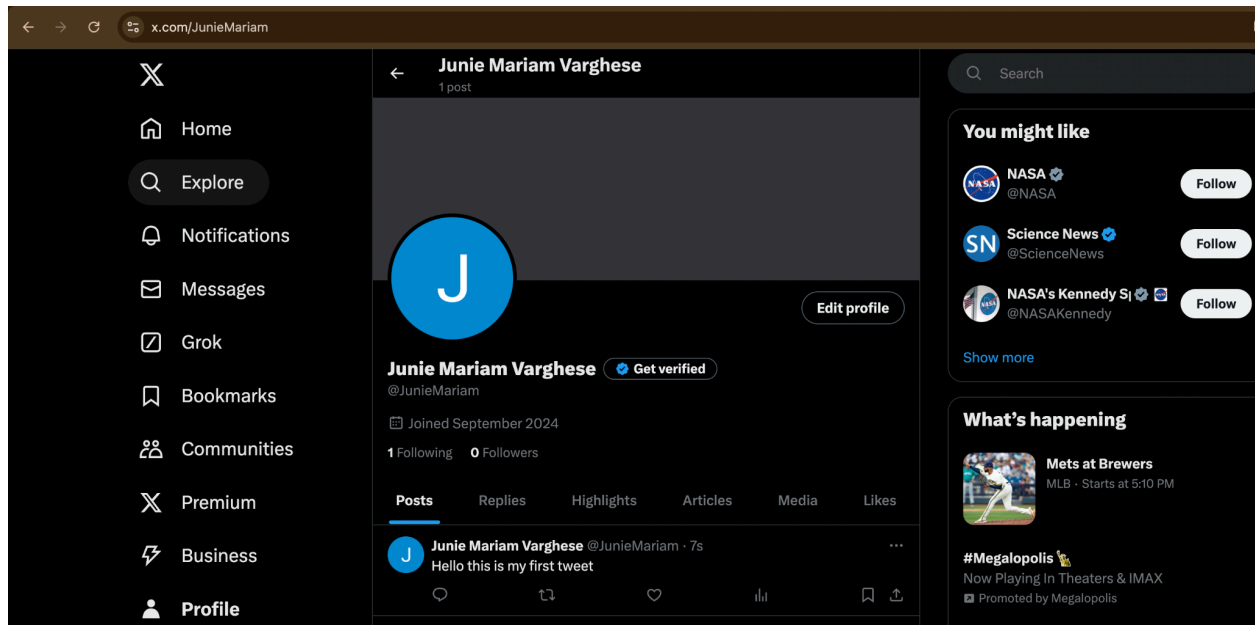


We used python to create the UI and the connections to the database and make the API Calls.

CASE1

1. Lets Post a tweet through our webUI.



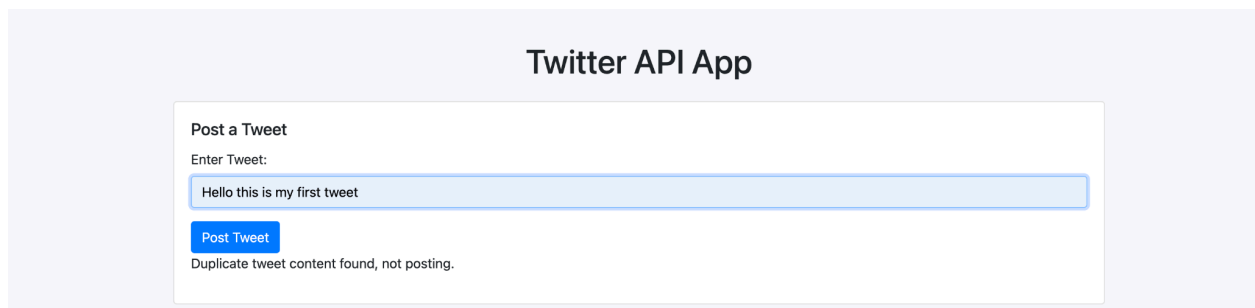


We can find our tweet posted on X.

CASE 2

We also need to test the negative cases. So when we have a post that's having the same content it shouldn't post.

Let's test our WebApp for this scenario.



2. Now lets test to delete the Post. So for that we need the tweetID. We have implemented a button in UI which populates the tweets created through our web application.

Delete a Tweet

Enter Tweet ID to Delete:

Delete Tweet

Retrieve a Tweet

Enter Tweet ID to Retrieve:

Retrieve Tweet

Display All Stored Tweets

Stored Tweets

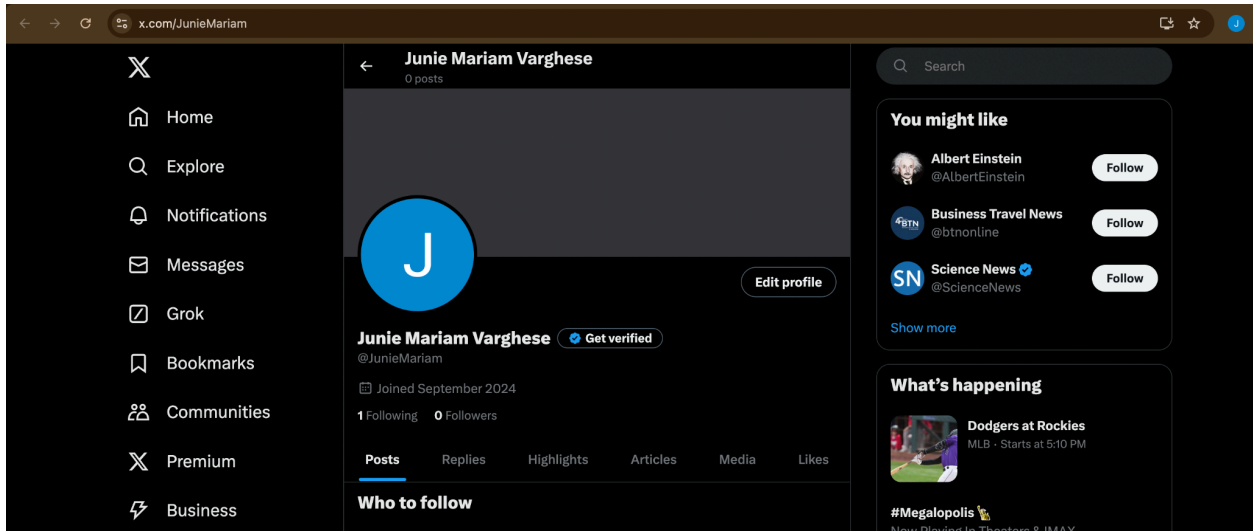
Tweet ID	Content
1839805923503812961	Hello this is my first tweet

Using the tweetId to delete the tweet from X.

Delete a Tweet

Enter Tweet ID to Delete:

Delete Tweet



The post is deleted.

So now lets test some negative cases:
Trying to delete a tweet id that doesn't exist.

Twitter API App

Post a Tweet

Enter Tweet:

Post Tweet

Delete a Tweet

Enter Tweet ID to Delete:

Delete Tweet

Retrieve a Tweet

Enter Tweet ID to Retrieve:

Retrieve Tweet

Display All Stored Tweets

Delete a Tweet

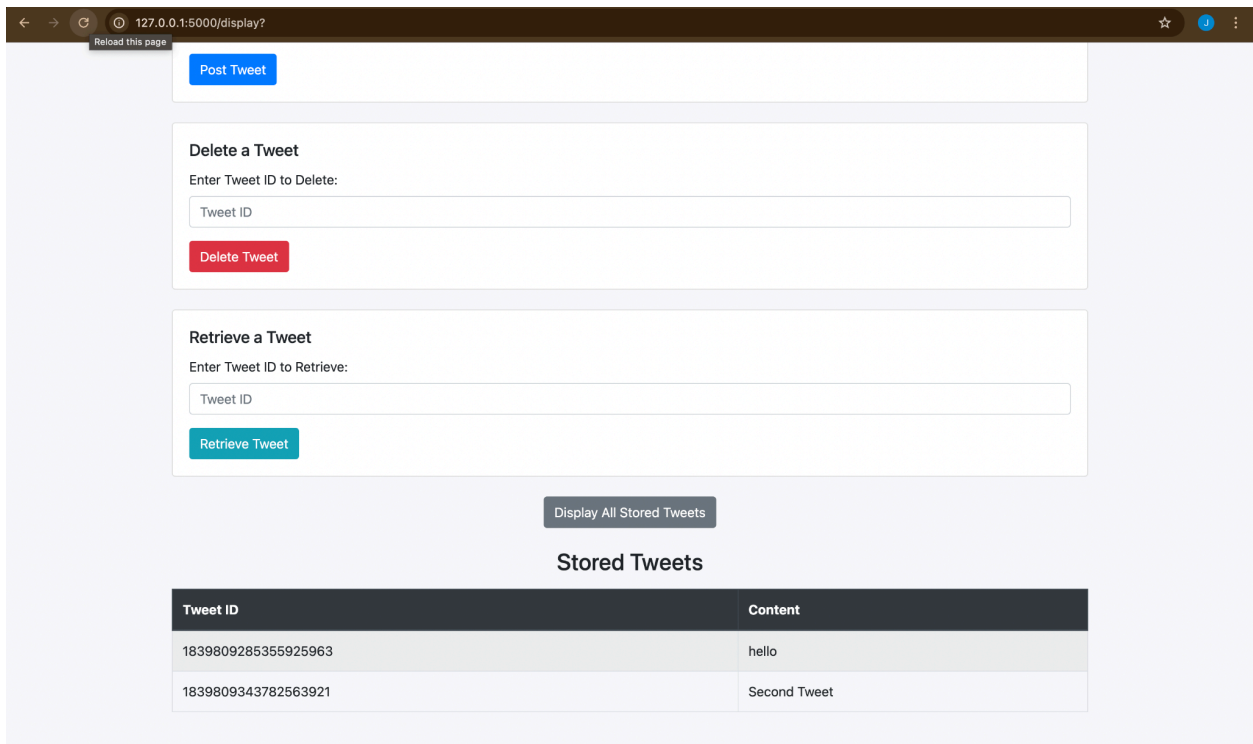
Enter Tweet ID to Delete:

Delete Tweet

Invalid tweet ID: Tweet not found.

3. Retrieve Tweets from tweetId

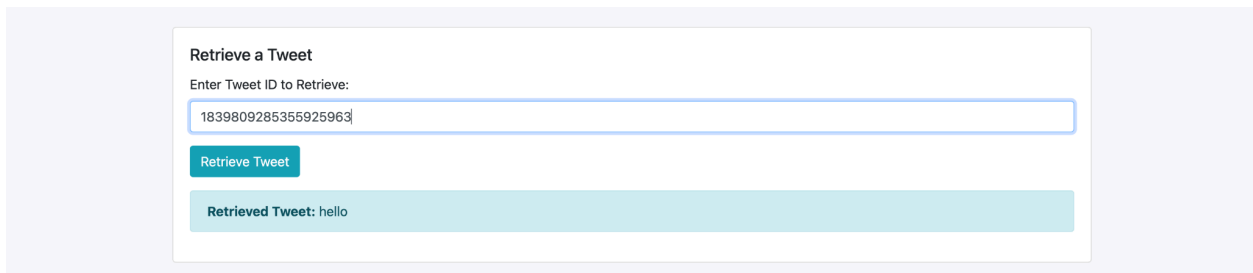
So clicking Display all stored tweets we get to know all tweets with ID. This is implemented to get the tweetId and to validate if the right content is retrieved.



The screenshot shows a web application interface with a browser window at the top displaying the URL '127.0.0.1:5000/display?'. The interface includes three main sections: 'Post Tweet' with a blue button; 'Delete a Tweet' with a text input for 'Tweet ID' and a red 'Delete Tweet' button; and 'Retrieve a Tweet' with a text input for 'Tweet ID' and a teal 'Retrieve Tweet' button. Below these is a 'Display All Stored Tweets' button. At the bottom, a table titled 'Stored Tweets' displays two rows of data.

Tweet ID	Content
1839809285355925963	hello
1839809343782563921	Second Tweet

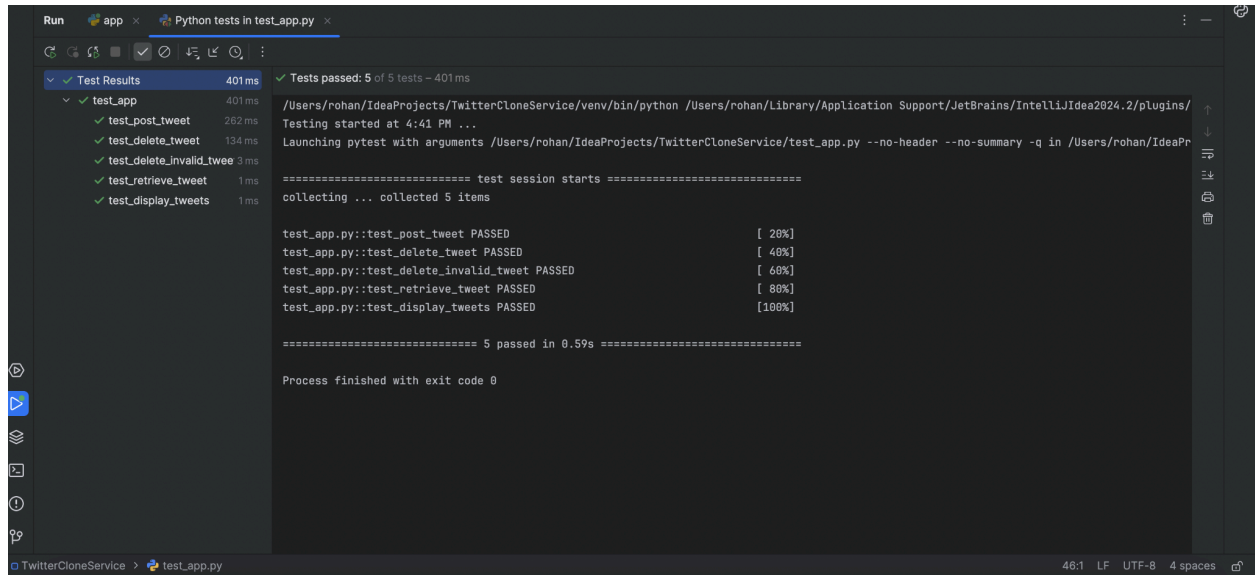
So getting the tweetId from the stored tweets. Lets retrieve the content.



This screenshot focuses on the 'Retrieve a Tweet' section. The 'Enter Tweet ID to Retrieve:' input field contains the ID '1839809285355925963'. Below the input is a teal 'Retrieve Tweet' button. A light blue feedback box at the bottom of the section displays the text 'Retrieved Tweet: hello'.

Since retrieval API is not accessible for free users. We implemented a database to store the tweets and retrieved the information.

Unit Test Results



```
Run app x Python tests in test_app.py x
Test Results 401ms
test_app 401ms
test_post_tweet 262 ms
test_delete_tweet 134 ms
test_delete_invalid_tweet 3 ms
test_retrieve_tweet 1 ms
test_display_tweets 1 ms
Tests passed: 5 of 5 tests - 401 ms
/Users/rohan/IdeaProjects/TwitterCloneService/venv/bin/python /Users/rohan/Library/Application Support/JetBrains/IntelliJ/Idea2024.2/plugins/
Testing started at 4:41 PM ...
Launching pytest with arguments /Users/rohan/IdeaProjects/TwitterCloneService/test_app.py --no-header --no-summary -q in /Users/rohan/IdeaPr
===== test session starts =====
collecting ... collected 5 items

test_app.py::test_post_tweet PASSED [ 20%]
test_app.py::test_delete_tweet PASSED [ 40%]
test_app.py::test_delete_invalid_tweet PASSED [ 60%]
test_app.py::test_retrieve_tweet PASSED [ 80%]
test_app.py::test_display_tweets PASSED [100%]

===== 5 passed in 0.59s =====

Process finished with exit code 0
TwitterCloneService > test_app.py 46:1 LF UTF-8 4 spaces
```

For Reference Github Repository: <https://github.com/juniemariam/TwitterClone>