**University of Leeds School of Computing**

**COMP3011, 2023-2024**

**Web Services and Web Data**

A RESTful API for

News Aggregation

By

insert your name here

insert your university number and email here

**Date:** 2024/3/26

# Introduction

Due to personal issues, I was unable to access the pythonanywhere account that I originally created myself via my school email. Apologies here. The site that works now is http://hanting0625.pythonanywhere.com

Server admin

Username: hanting

Password: 86109113Mht

The program implements most of it, but there are some problems with the delete function and the post function of the program. This is probably because the database was not interfaced properly when it was created. So it can't be realised when it is created by client. The client displays normal when accessing other people's websites.

# The Database

Author Model

The Author model is designed as a custom user model that inherits from AbstractBaseUser, where the username field is used to uniquely identify each author, and the name field is used to store the author's name.

The unique identifier field for login is specified by setting USERNAME\_FIELD to 'username'.

NewsStory Model

The NewsStory model represents a news story and contains several fields that describe specific information about the story. Fields such as headline store the title of the story. category and region are used to categorise the news story, representing the category and region of the story, respectively.

- author is a ForeignKey that points to the Author model and establishes a relationship between the news story and its author. When an author is deleted, the associated news story is cascaded (on\_delete=models.CASCADE).

- date and details store when and what the news story was published, respectively.

# The APIs

1. LoginView

LoginView handles the login request, using the POST method to receive the username and password, and then calling the authenticate function to verify the credentials.

If the authentication is successful, the user can proceed to Post news and Delete news.

- A successful response contains a welcome message and HTTP 200 status code, while a failed response returns an error message and HTTP 401 status code.

2. LogoutView

LogoutView logs out the current user. When a POST request is received, it calls Django's logout function to clear the user's session. The response contains a logout confirmation message and HTTP 200 status code.

3. NewsStoryList

The NewsStoryList view supports GET and POST requests for retrieving and creating news stories.

GET requests are processed using query parameters to filter news categories, regions, and dates. If the values of the query parameters are supplied different from the default values, the appropriate filters are applied. All news is displayed if there are no corresponding filters.

POST request processing requires the user to authenticate first. After authentication, the news story data is read from the request body and a new NewsStory instance is created and saved.

4. News story details

Used to handle the deletion of news stories. It inherits from generics.DestroyAPIView and can find and delete specific news stories based on the primary key. If the user is not logged in and tries to delete a news story, the system will return an HTTP 401 status code.

5. Post Viewset

Provides a complete view set for Post models, including create, retrieve, update and delete operations.

I use ModelViewSet to reduce the complexity of the code.

# The Client

When the user enters the correct website, the user is prompted for specific commands such as 'login', 'logout', 'post', 'news', 'list', 'delete', or 'exit'. These commands allow the user to perform different actions.

After the user has entered the correct website, the user will be prompted to 'Enter command (login, logout, post, news, list, delete, exit):' to enter the following actions, and different functions will be called for different inputs. Where login(): implements the login function by sending a POST request to the login endpoint of the API. The login will continue to prompt the user for a username and password, and the client will package the data and upload it to the website. If the login is successful, the login status will be retained and subsequent post and delet functions will check if the user is logged in.

The implementation of this is to emulate a session with parameters, allowing the login to be maintained across functions.

The delete\_story() function is called when the user wants to delete a published news story. First, the function checks to see if the user is logged in. If not, the user is prompted to log in first.

- If the user is logged in, the function prompts the user to enter a unique identifier for the news story to be deleted (e.g., the story's ID or key value).

The post\_story() function is called when the user wants to post a new news story. The situation is similar to DELETE. Users who are not logged in cannot post news stories.

. Once the user is logged in, the function prompts the user to enter the contents of the news story, such as the headline, category, region, and details.

- When a user requests to view news, the fetch\_news() function sends a GET request via the user's session to the news story's API endpoint. The function does not require the user to authenticate, since fetching news stories is usually a public operation. The server responds to the request and returns JSON data containing all the news stories.

- If the request is successful (HTTP status code 200), the user is presented with a list of news stories; otherwise, an error message is displayed.

- When the user wants to access the news outlet's catalogue, the list() function sends an authentication-free GET request to the news service's catalogue API endpoint.

- When the request succeeds, the user is presented with a formatted list of news organisations; if the request fails, the user is informed that an error has occurred.

- Unfortunately there are some problems with the implementation of post\_story(): and delete\_story(): on the implementation api, making it impossible to test the success of the client's functionality.

Logout implementation is very simple. If the user calls "logout". The function will first check if the user is logged in or not. If the user is logged in. A special parameter determines if the user is logged out or not. In this way, the effect of logout can be achieved.