WEB APP NLP

Daniel Suárez

THE CHALLENGE

We want to you to create a web UI where you call an API that uses NLP to analyze, summarize and extract important information from news articles. This challenge should be solved using Python, you can choose the framework that best suits you.

CHALLENGE ACCEPTED

🔗 https://github.com/jdsuarezj/wj_developerchallenge

FEATURES

- Django
- Modular design template
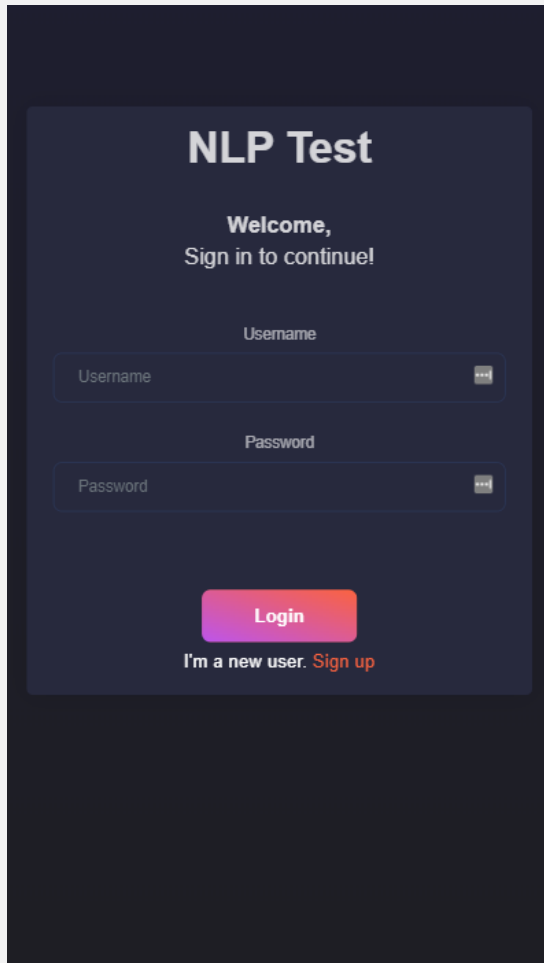- Python 3
- Bootstrap css
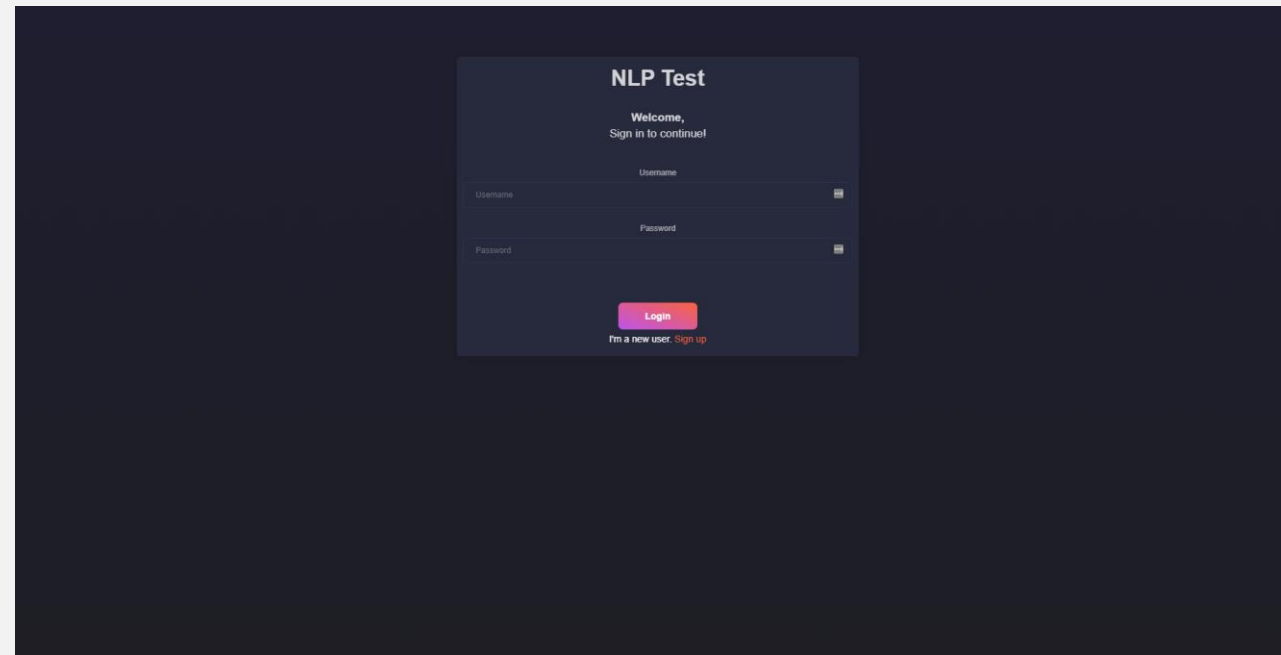- SQLite fake DB
- Aylien API

PART 1
User creation and login

Mobile

Desktop

NLP Test

Welcome,
Sign in to continue!

Username

Username

Password

Password

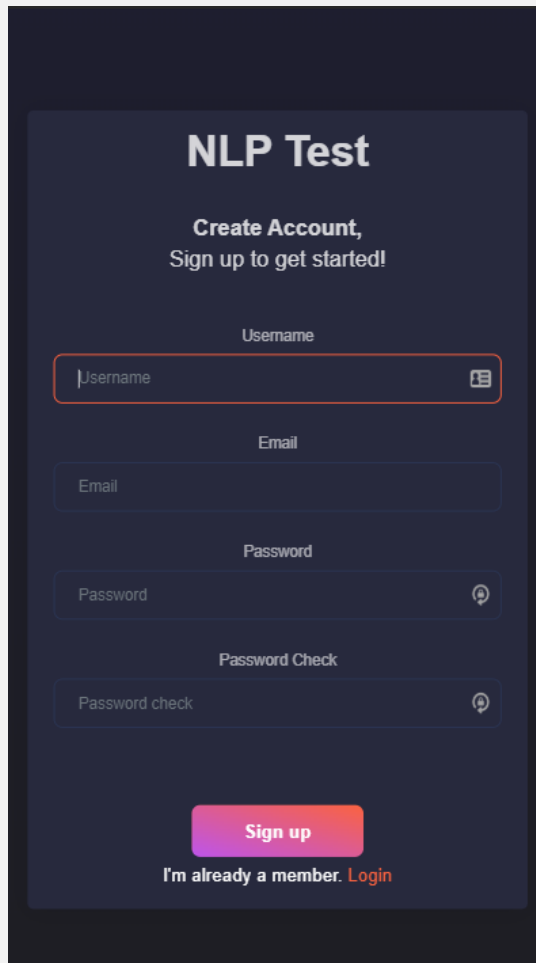**Login**

I'm a new user. Sign up

View designed for authentication and as initial screen.

Authentication with Django middleware without handling errors or alerts.
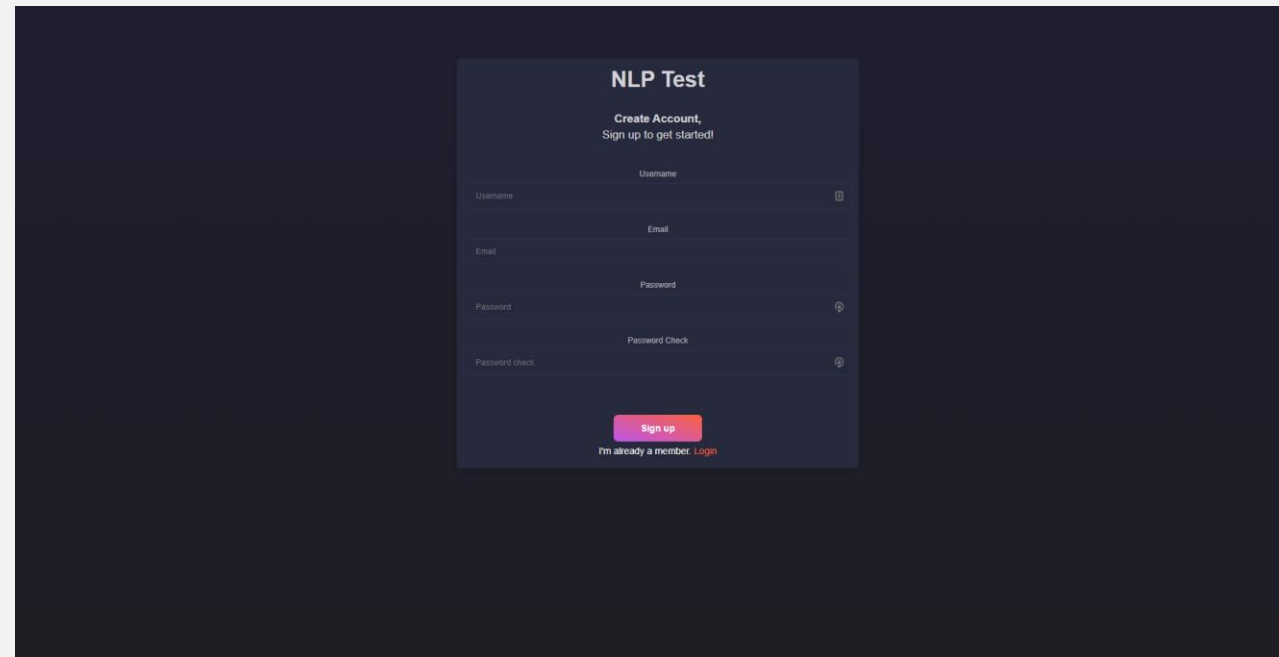
PART 1
User creation and login

Mobile

Desktop

User creation and validation view.

Authentication with Django middleware without handling errors or alerts.

PART 2
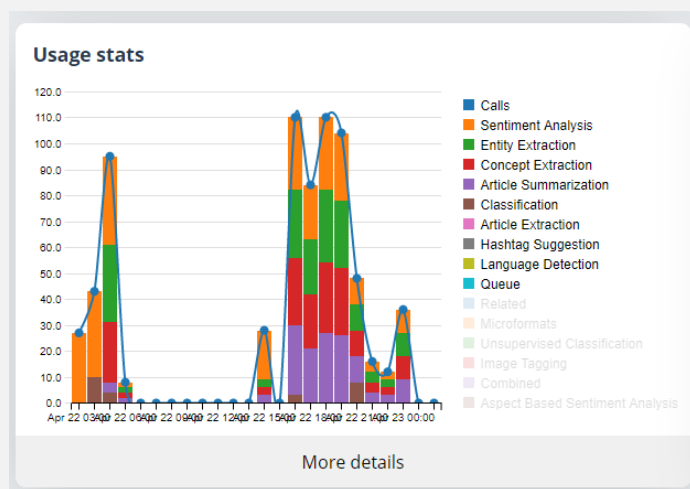Connect to Text API

Aylien - Python SDK

1. Aylien account

Usage stats

Legend:
- Calls
- Sentiment Analysis
- Entity Extraction
- Concept Extraction
- Article Summarization
- Classification
- Article Extraction
- Hashtag Suggestion
- Language Detection
- Queue
- Related
- Microformats
- Unsupervised Classification
- Image Tagging
- Combined
- Aspect Based Sentiment Analysis

More details

2. *Library installation from the requirements*

3. Import and config the API

```python
from django.http import HttpResponse, JsonResponse
from .forms import ScanForm
import json
import sys

# Library for aylien textapi
from aylienapiclient import textapi
# Credentials API
client = textapi.Client('ID', 'KEY') #CHANGE THIS with your Aylien's credentials
```

3. API Requests for each endpoint

```python
# IndexView
# Render the index with the responses from the API
@login_required(login_url="/login/")
def index(request):
    if request.method == "POST":
        form = ScanForm(request.POST)
        if form.is_valid():
            # URL from the TextArea in the index view
            urlInput = form.cleaned_data.get('urlInput')
            # Params for request the API
            paramInput = {'url': urlInput}
            # Sentiment endpoint
            sentiment = client.Sentiment(paramInput)
            # Entities endpoint
            entities = client.Entities(paramInput)
            # Concepts endpoint
            concepts = client.Concepts(paramInput)
            conceptsResult = [ v for v in concepts['concepts'].values() ]
            # Summarize endpoint
            summarize = client.Summarize(paramInput)

            # Return the data to the view
            return render(
                request, 'pages/scan.html',
                {
                    'form': form,
                    'text': sentiment['text'],
                    'polarity': sentiment['polarity'],
                    'subjectivity': sentiment['subjectivity'],
                    'polarity_confidence': (sentiment['polarity_confidence'])*100,
                    'subjectivity_confidence': (sentiment['subjectivity_confidence'])*100,
                    'entities': entities['entities'],
                    'concepts': conceptsResult,
                    'summarize': summarize['sentences']
                }
            )
    else:
        form = ScanForm()
    return render(request, 'index.html', {'form': form})
```

PART 3
Build UI

Where the user enters the URL.

In this view it is explained that it is NLP, which is Aylien. It is displayed clearly and easy to use and with a single button. Also, some author references are left talking about NLP and its benefits.

PART 3
Build UI

Scan

Where is the analysis.

The article analysis is shown in an orderly and easy to understand way. The classification endpoint presented errors with some articles.

PART 3
Build UI

Extra: User profile

A quick layout of the fake user profile page.

GRACIAS

Daniel Suárez ✉ *jdsuarezj@unal.edu.co*