Лабораторна робота 2. Розробка додатків на Python і Flask

Команда 13

Сидоренко Альбіна Олександрівна

Гаєва Поліна Денисівна

Шудрик Андрій Олександрович

Файл скрипту і зображення : https://github.com/PolinaHaieva/P_2_13.git

1.

Клонування репозиторію та перехід до директорії проекту

1_folder_structure.png

```
theia@theiadocker-secaccpol:/home/project$ git clone https://github.com/ibm-developer-skills-network/oaqjp-final-project-
emb-ai.git
Cloning into 'oaqjp-final-project-emb-ai'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 16 (delta 3), reused 0 (delta 0), pack-reused 9
Receiving objects: 100% (16/16), 8.21 KiB | 8.21 MiB/s, done.
Resolving deltas: 100% (3/3), done.
theia@theiadocker-secaccpol:/home/project$ ls
oaqjp-final-project-emb-ai
theia@theiadocker-secaccpol:/home/project$ cd oaqjp-final-project-emb-ai
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$ ls
LICENSE  README.md  static  templates
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$
```

2.

Використання Embeddable Watson AI для емоційного аналізу тексту

2a_emotion_detection.png

```python
import requests

def emotion_detector(text_to_analyze):
    url = 'https://sn-watson-emotion.labs.skills.network/v1/watson.runtime.nlp.v1/NlpService/EmotionPredict'
    headers = {"grpc-metadata-mm-model-id": "emotion_aggregated-workflow_lang_en_stock"}
    input_json = { "raw_document": { "text": text_to_analyze } }
    response = requests.post(url, json = input_json, headers=headers)
    return response.text

```

Тестування функції

2b_application_creation.png

```
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$ python3.11
Python 3.11.9 (main, Apr  6 2024, 17:59:24) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from emotion_detection import emotion_detector
>>> print(emotion_detector("I love this new technology")
...
...
KeyboardInterrupt
>>> print(emotion_detector("I love this new technology"))
{"emotionPredictions":[{"emotion":{"anger":0.0132405795, "disgust":0.0020517302, "fear":0.009090992, "joy":0.9699522, "sa
dness":0.054984167}, "target":"", "emotionMentions":[{"span":{"begin":0, "end":26, "text":"I love this new technology"},
"emotion":{"anger":0.0132405795, "disgust":0.0020517302, "fear":0.009090992, "joy":0.9699522, "sadness":0.054984167}}]},
 "producerId":{"name":"Ensemble Aggregated Emotion Workflow", "version":"0.0.1"}}
>>>
```

3.

Модифікація відповіді для користувача

3a_output_formatting.png

```python
import requests
import json

def emotion_detector(text_to_analyze):
    url = 'https://sn-watson-emotion.labs.skills.network/v1/watson.runtime.nlp.v1/NlpService/EmotionPredict'
    headers = {"grpc-metadata-mm-model-id": "emotion_aggregated-workflow_lang_en_stock"}
    input_json = { "raw_document": { "text": text_to_analyze } }

    response = requests.post(url, json=input_json, headers=headers)

    response_data = json.loads(response.text)
    emotions = response_data['emotionPredictions'][0]['emotion']
    dominant_emotion = max(emotions, key=emotions.get)

    result = {
        'anger': emotions['anger'],
        'disgust': emotions['disgust'],
        'fear': emotions['fear'],
        'joy': emotions['joy'],
        'sadness': emotions['sadness'],
        'dominant_emotion': dominant_emotion
    }
    return result
```

Тестування функції

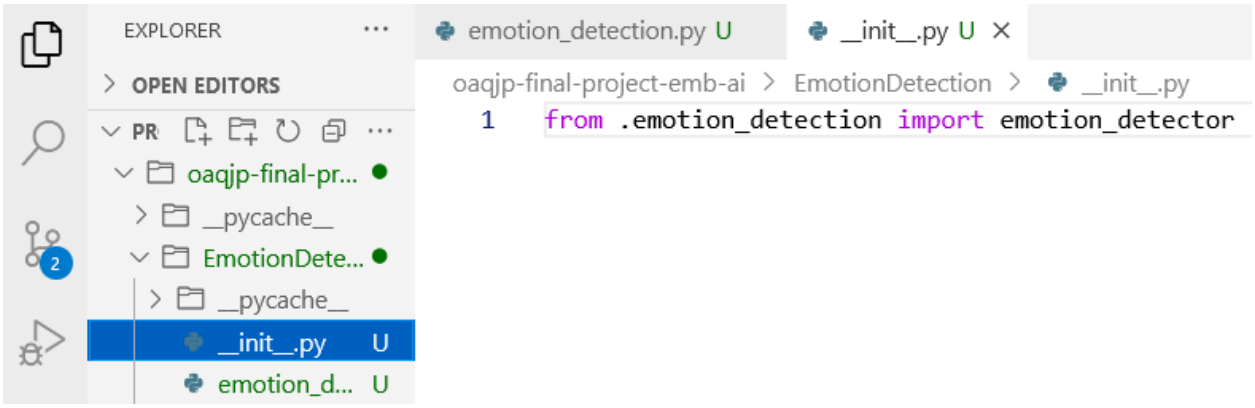
3b_formatted_output_test.png

```
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$ python3.11
Python 3.11.9 (main, Apr  6 2024, 17:59:24) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from emotion_detecction import emotion_detector
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'emotion_detecction'
>>> from emotion_detection import emotion_detector
>>> print(emotion_detector("I am so happy I am doing this"))
{'anger': 0.0043079085, 'disgust': 0.00041127237, 'fear': 0.0037504788, 'joy': 0.9918804, 'sadness': 0.014091322,
 'dominant_emotion': 'joy'}
>>>
```

4.

Створення пакету

4a_packaging.png



Тестування пакету

4b_packaging_test.png

```
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$ python3.11
Python 3.11.9 (main, Apr  6 2024, 17:59:24) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from EmotionDetection import emotion_detector
>>> print(emotion_detector("I hate working long hours"))
{'anger': 0.64949876, 'disgust': 0.03718168, 'fear': 0.05612277, 'joy': 0.00862553, 'sadness': 0.1955148, 'dominant_
emotion': 'anger'}
>>>
```

5.

Юніт-тести

5a_unit_testing.png

```python
from EmotionDetection.emotion_detection import emotion_detector
import unittest

class TestEmotionDetection(unittest.TestCase):
    def test_emotion_detection(self):
        test_cases = [
            ("I am glad this happened", 'joy'),
            ("I am really mad about this", 'anger'),
            ("I feel disgusted just hearing about this", 'disgust'),
            ("I am so sad about this", 'sadness'),
            ("I am really afraid that this will happen", 'fear')
        ]

        for text, expected_emotion in test_cases:
            with self.subTest(text=text):
                result = emotion_detector(text)
                self.assertEqual(result['dominant_emotion'], expected_emotion)

if __name__ == '__main__':
    unittest.main()
```

Перевірка юніт-тестами

5b_unit_testing_result.png

```
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$ python3.11 test_emotion_detect
ion.py
.
----------------------------------------------------------------------
Ran 1 test in 0.567s

OK
theia@theiadocker-secaccpol:/home/project/oaqjp-final-project-emb-ai$
```

6.

Деплой застоснку через Flask

6a_server.png

```python
from flask import Flask, render_template, request
from EmotionDetection import emotion_detector

app = Flask(__name__)

@app.route('/emotionDetector', methods=['GET'])
def detect_emotion():
    """
    This endpoint receives a text input via query parameters,
    processes it using the emotion_detector function, and returns the detected emotions.
    """
    text_to_analyze = request.args.get('textToAnalyze')
    response = emotion_detector(text_to_analyze)

    if not text_to_analyze or response.get('dominant_emotion') is None:
        return "Invalid text! Please provide a valid input."

    emotions = ", ".join([f"'{emotion}': {score}" for emotion, score in response.items() if emotion != 'dominant_emotion'])
    dominant_emotion = response['dominant_emotion']

    return f"For the given statement, the system detected the following emotions: {emotions}. The dominant emotion is {dominant_em

@app.route('/')
def index():
    """
    This endpoint renders the main HTML page.
    """
    """
    return render_template('index.html')

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000)
```

Запуск додатку і перевірка роботи

6b_deployment_test.png

7.

Обробка виключних випадків в функції

7a_error_handling_function.png

```python
def emotion_detector(text_to_analyze):
    url = 'https://sn-watson-emotion.labs.skills.network/v1/watson.runtime.nlp.v1/NlpService/EmotionPredict'
    headers = {"grpc-metadata-mm-model-id": "emotion_aggregated-workflow_lang_en_stock"}
    input_json = { "raw_document": { "text": text_to_analyze } }

    response = requests.post(url, json = input_json, headers=headers)

    if response.status_code == 400:
        return {
            'anger': None,
            'disgust': None,
            'fear': None,
            'joy': None,
            'sadness': None,
            'dominant_emotion': None,
     }


    response_data = json.loads(response.text)
    emotions = response_data['emotionPredictions'][0]['emotion']
    dominant_emotion = max(emotions, key=emotions.get)

    result = {
        'anger': emotions['anger'],
        'disgust': emotions['disgust'],
        'fear': emotions['fear'],
        'joy': emotions['joy'],
        'sadness': emotions['sadness'],
        'dominant_emotion': dominant_emotion
    }

    return result
```

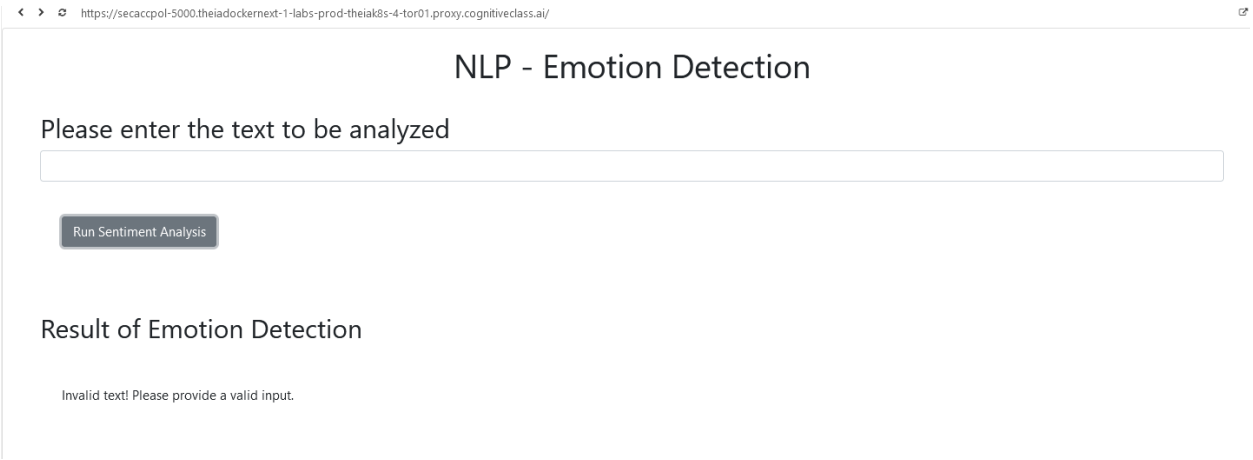
Обробка виключних випадків в сервері

7b_error_handling_server.png

```python
from flask import Flask, render_template, request
from EmotionDetection import emotion_detector

app = Flask(__name__)

@app.route('/emotionDetector', methods=['GET'])
def detect_emotion():
    """
    This endpoint receives a text input via query parameters,
    processes it using the emotion_detector function, and returns the detected emotions.
    """
    text_to_analyze = request.args.get('textToAnalyze')

    if not text_to_analyze:
        return "Invalid text! Please provide a valid input."

    response = emotion_detector(text_to_analyze)
    dominant_emotion = response.get('dominant_emotion')

    if dominant_emotion is None:
        return "Invalid text! Please provide a valid input."

    emotions = ", ".join([f"'{emotion}': {score}" for emotion, score in response.items() if emotion != 'dominant_emotion'])
    result = (f"For the given statement, the system detected the following emotions: {emotions}. "
              f"The dominant emotion is {dominant_emotion}.")

    return result

@app.route('/')
def index():
    """
    This endpoint renders the main HTML page.
    """
    return render_template('index.html')

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000)
```

Перевірка обробки виключень(пустого вводу)

7c_error_handling_interface.png



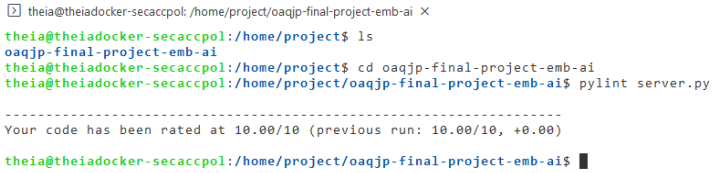
8.

Стилістична модифікація коду
8a_server_modified.png

```python
from flask import Flask, render_template, request
from EmotionDetection import emotion_detector

app = Flask(__name__)

@app.route('/emotionDetector', methods=['GET'])
def detect_emotion():
    """
    Endpoint to detect emotions from a given text.

    Receives a text input via query parameters, processes it using the
    emotion_detector function, and returns the detected emotions.

    Returns:
        str: A formatted string containing the emotions and their scores,
        or an error message if the input is invalid.
    """
    text_to_analyze = request.args.get('textToAnalyze')

    if not text_to_analyze:
        return "Invalid text! Please provide a valid input."

    response = emotion_detector(text_to_analyze)
    dominant_emotion = response.get('dominant_emotion')

    if dominant_emotion is None:
        return "Invalid text! Please provide a valid input."

    emotions = [
        f"'{emotion}': {score}"
        for emotion, score in response.items()
        if emotion != 'dominant_emotion'
    ]
    result = (
        f"For the given statement, the system detected the following emotions: "
        f"{', '.join(emotions)}. The dominant emotion is {dominant_emotion}."
    )

    return result

@app.route('/')
def index():
    """
    Renders the main HTML page.

    Returns:
        str: The rendered HTML content for the index page.
    """
    return render_template('index.html')

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000)
```

Перевірка через pylint

8b_static_code_analysis.png



Висновок: в ході цієї роботи ми створили застосунок із використанням фреймворку Flask.

А також:

Обробляли HTTP/HTTPS запити, рендерили шаблони, деплоїли, обробляти виключні ситуації.

Отримали досвід роботи із бібліотеками json, request, Embeddable Watson AI

Навчились робити кращим стиль коду, використовуючи PyLint