Flask-based API for name similarity validation

```
from flask import Flask, request, jsonify
from difflib import SequenceMatcher

app = Flask(__name__)

# Utility function: Calculate similarity between two strings
def calculate_similarity(string1, string2):
    return SequenceMatcher(None, string1, string2).ratio()

# Utility function: Validate and clean input names
def clean_name(name):
    return name.lower().strip()

# Core function: Calculate name match percentage
def calculate_name_match(name_aadhar, name_input, weights=None):
    if weights is None:
        weights = {"full_name": 0.7, "tokens": 0.3}

    # Normalize names
    name_aadhar = clean_name(name_aadhar)
    name_input = clean_name(name_input)

    # Full name similarity
    full_name_similarity = calculate_similarity(name_aadhar,
name_input)

    # Token-based similarity
    aadhar_tokens = name_aadhar.split()
    input_tokens = name_input.split()
    token_similarities = [
        calculate_similarity(aadhar_token, input_token)
        for aadhar_token, input_token in zip(aadhar_tokens,
input_tokens)
    ]

    # Include unmatched tokens
    extra_tokens = max(len(aadhar_tokens), len(input_tokens)) -
len(token_similarities)
    token_similarities.extend([0] * extra_tokens)

    token_average_similarity = sum(token_similarities) /
len(token_similarities)

    # Weighted match percentage
    match_percentage = (
        weights["full_name"] * full_name_similarity +
        weights["tokens"] * token_average_similarity
    ) * 100

    return round(match_percentage, 2)
```

```python
# Flask route for name validation
@app.route('/validate-name', methods=['POST'])
def validate_name():
    try:
        # Get JSON input
        data = request.json
        name_aadhar = data.get("name_aadhar", "").strip()
        name_input = data.get("name_input", "").strip()

        # Validate input
        if not name_aadhar or not name_input:
            return jsonify({"error": "Both 'name_aadhar' and 'name_input' are required."}), 400

        # Calculate match percentage
        match_percentage = calculate_name_match(name_aadhar, name_input)
        return jsonify({"match_percentage": match_percentage})

    except Exception as e:
        return jsonify({"error": f"An unexpected error occurred: {str(e)}"}), 500

# Test endpoint to check if API is running
@app.route('/')
def index():
    return "Name Validation API is running!"

if __name__ == '__main__':
    app.run(debug=True)
```

Includes:-
return status success or error key also in response

```
from flask import Flask, request, jsonify
from difflib import SequenceMatcher

app = Flask(__name__)

def calculate_similarity(string1, string2):
    return SequenceMatcher(None, string1, string2).ratio()

def clean_name(name):
    return name.lower().strip()

def calculate_name_match(name_aadhar, name_input, weights=None):
    if weights is None:
        weights = {"full_name": 0.7, "tokens": 0.3}

    name_aadhar = clean_name(name_aadhar)
    name_input = clean_name(name_input)

    full_name_similarity = calculate_similarity(name_aadhar,
name_input)

    aadhar_tokens = name_aadhar.split()
    input_tokens = name_input.split()
    token_similarities = [
        calculate_similarity(aadhar_token, input_token)
        for aadhar_token, input_token in zip(aadhar_tokens,
input_tokens)
    ]

    extra_tokens = max(len(aadhar_tokens), len(input_tokens)) -
len(token_similarities)
    token_similarities.extend([0] * extra_tokens)

    token_average_similarity = sum(token_similarities) /
len(token_similarities)

    match_percentage = (
        weights["full_name"] * full_name_similarity +
        weights["tokens"] * token_average_similarity
    ) * 100

    return round(match_percentage, 2)

@app.route('/validate-name', methods=['POST'])
def validate_name():
    try:
        data = request.json
        name_aadhar = data.get("name_aadhar", "").strip()
```

```
        name_input = data.get("name_input", "").strip()

        if not name_aadhar or not name_input:
            return jsonify({
                "status": "error",
                "error": "Both 'name_aadhar' and 'name_input' are
required."
            }), 400

        match_percentage = calculate_name_match(name_aadhar,
name_input)
        return jsonify({
            "status": "success",
            "match_percentage": match_percentage
        })

    except Exception as e:
        return jsonify({
            "status": "error",
            "error": f"An unexpected error occurred: {str(e)}"
        }), 500

@app.route('/')
def index():
    return "Name Validation API is running!"

if __name__ == '__main__':
    app.run(debug=True)
```

Added regex validation for inputs and replaced `jsonify` with direct dictionary responses.

```
from flask import Flask, request
from difflib import SequenceMatcher
import re

app = Flask(__name__)
def calculate_similarity(string1, string2):
    return SequenceMatcher(None, string1, string2).ratio()
def clean_name(name):
    return name.lower().strip()
def is_valid_name(name):
    return bool(re.match(r'^[a-zA-Z\s]+$', name))

def calculate_name_match(name_aadhar, name_input, weights=None):
    if weights is None:
        weights = {"full_name": 0.7, "tokens": 0.3}

    name_aadhar = clean_name(name_aadhar)
    name_input = clean_name(name_input)

    full_name_similarity = calculate_similarity(name_aadhar,
name_input)

    aadhar_tokens = name_aadhar.split()
    input_tokens = name_input.split()
    token_similarities = [
        calculate_similarity(aadhar_token, input_token)
        for aadhar_token, input_token in zip(aadhar_tokens,
input_tokens)
    ]

    extra_tokens = max(len(aadhar_tokens), len(input_tokens)) -
len(token_similarities)
    token_similarities.extend([0] * extra_tokens)

    token_average_similarity = sum(token_similarities) /
len(token_similarities)

    match_percentage = (
        weights["full_name"] * full_name_similarity +
        weights["tokens"] * token_average_similarity
    ) * 100

    return round(match_percentage, 2)

@app.route('/validate-name', methods=['POST'])
def validate_name():
    try:
        data = request.json
        if not data:
```

```
            return {"status": "error", "error": "Request body must
be JSON."}, 400

        name_aadhar = data.get("name_aadhar", "").strip()
        name_input = data.get("name_input", "").strip()
        if not name_aadhar or not name_input:
            return {"status": "error", "error": "Both
'name_aadhar' and 'name_input' are required."}, 400

        if not (is_valid_name(name_aadhar) and
is_valid_name(name_input)):
            return {"status": "error", "error": "Invalid name
format. Only letters and spaces are allowed."}, 400

        match_percentage = calculate_name_match(name_aadhar,
name_input)
        return {"status": "success", "match_percentage":
match_percentage}

    except Exception as e:
        return {"status": "error", "error": f"An unexpected error
occurred: {str(e)}"}, 500

@app.route('/')
def index():
    return "Name Validation API is running!"

@app.route('/test', methods=['GET'])
def test():
    return {"status": "success", "message": "Test endpoint
working"}

if __name__ == '__main__':
    app.run(debug=True)
```