

Religion-Related API

An Islamic information REST API

Mohamed Iheb Mhenni

January, 2024

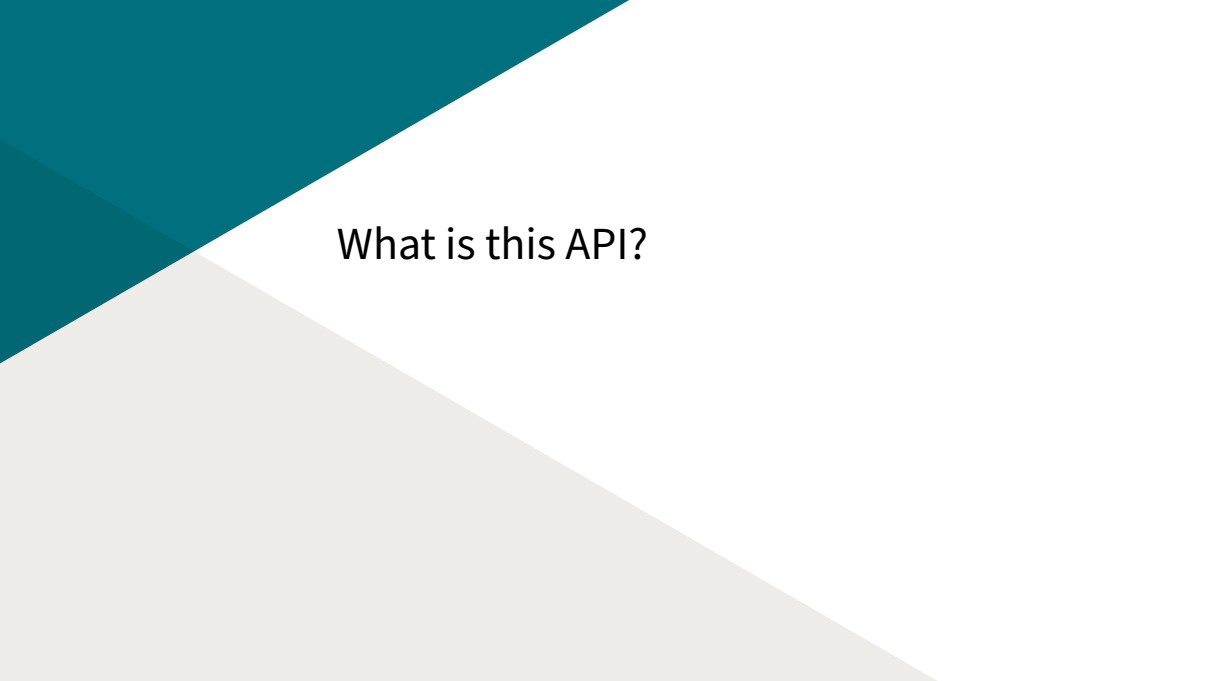
Overview



1. What is this API?

2. Scalability

3. Challenges

The background of the slide is composed of two large, overlapping geometric shapes. A teal-colored shape occupies the top-left corner, while a light gray shape occupies the bottom-left corner. The rest of the slide is white. The text "What is this API?" is centered in the white area.

What is this API?

What is this API?

Definition, Functions and Features

This religion-related API is **an API that helps developers find relevant and up-to-date Islamic information**. It enables users to send GET requests since it depends on two external APIs: Aladhan and Quran cloud APIs.



What is this API?

Interface



WEB SERVICE (IT 325) API PROJECT (Religion Related API Interface)

Get Quran Edition

Enter Quran Edition:

Get Quran Page

Enter Quran Page Number:

Enter Quran Page Number:

Enter Quran Edition:

Enter Quran Edition (e.g., al-jawad):

Get Sajda Ayahs

Enter Quran Edition:

Enter Quran Edition (e.g., al-jawad):

Get Asma Al Husna

Enter Asma Al Husna Numbers:

Enter Asma Al Husna Numbers (e.g., 1,2,3):

Get Prayer Times

Enter City:

Enter City:

Enter Country:

Enter Country:

Get Calendar by City

Enter Year:

Enter Year:

Enter Month:

Enter Month:

Enter City:

Enter City:


Enter Country:

Enter Country:

What is this API?

Technically

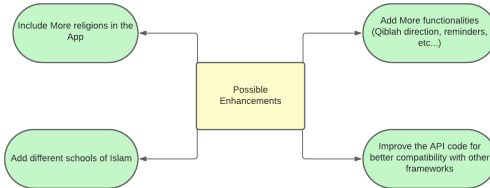




Scalability

Scalability

Possible Enhancements



The background of the slide is composed of two large, overlapping geometric shapes. A teal-colored shape occupies the top-left corner, while a light beige shape occupies the bottom-left corner. The rest of the slide is white. The word "Challenges" is centered in the white area.

Challenges

Challenges

What are the problems I faced and how did I overcome them?

- ▶ Connecting the API's backend code to the Front end part.

Solution: Using .js files and using the right fetching HTTP requests. (<%= %>)

- ▶ Error 404 not found occurs more often.

Solution: Use of flasgger instead of swagger since flasgger is a combination of swagger and flask accessible on localhost:3000

Appendix: script.js and all.py GET request for Quran edition snippets



```
@app.route('/quran-edition', methods=['GET'])
def quran_edition():
    """
    Retrieve Quran edition in text format
    ---
    parameters:
      - name: edition
        in: query
        type: string
        required: true
        description: A Quran edition identifier.
        example: en.asad

    responses:
      200:
        description: Quran edition successfully retrieved
    """
    try:
        edition_identifier = request.args.get('edition')
        quran_edition_data = get_quran_edition(edition_identifier)
        return jsonify(quran_edition_data)

    except Exception as e:
        return jsonify({"error": f"Unexpected error occurred: {e}"}), 500
```

```
function getQuranEdition() {
    const editionInput = document.getElementById('quranEdition');
    const edition = editionInput.value;

    fetch(`http://localhost:3000/quran-edition?edition=${edition}`)
        .then(response => {
            if (!response.ok) {
                throw new Error(`Error: ${response.status}`);
            }
            return response.json();
        })
        .then(data => {
            const resultDiv = document.getElementById('resultQuranEdition');
            resultDiv.innerHTML = `<pre>${JSON.stringify(data, null, 2)}</pre>`;
        })
        .catch(error => {
            console.error(`Error: ${error.message}`);
            const resultDiv = document.getElementById('resultQuranEdition');
            resultDiv.innerHTML = `<p>Error: ${error.message}</p>`;
        });
}
```