TUNIS BUSINESS SCHOOL
UNIVERSITY OF TUNIS

IT 325 Project Report of a Comprehensive Analysis of Flask API for Islamic Information

Mohamed Iheb Mhenni

January 21, 2024

Contents

List of Figures

List of Tables

Abstract

This report presents a detailed analysis of a Flask-based API developed for retrieving Islamic information. The API encompasses functionalities such as fetching prayer times, Quran pages, sajda ayahs, Asma Al Husna (Names of Allah), and Quran editions. The report covers the methodology, implementation details, code listings, API documentation, results, discussion, and a conclusion.

1 Overview

The Flask API for Islamic information is designed to serve as a versatile tool for developers aiming to integrate Islamic data into their applications. The API achieves this by utilizing external APIs, such as the Aladhan API and Alquran Cloud API, to fetch accurate and up-to-date information. This implies that the API acts as an intermediary, connecting to existing data sources to provide developers with a streamlined method of accessing Islamic information.

2 Introduction

The primary purpose of the API is to offer developers a convenient and efficient tool for accessing up-to-date Islamic information. The development process involves multiple stages and steps, all of which contribute to the creation of a fully functional API. The subsequent sections of the report will delve into the specifics of these stages, providing a comprehensive understanding of how the API was conceptualized, designed, and implemented.

3 Methodology

The development approach for the Flask API is centered around the use of Python as the primary programming language. Python's flexibility and versatility make it well-suited for this project, which involves integrating multiple aspects of Islamic data. The methodology also incorporates various libraries, including Flask, flasgger, flask$_cors, andrequests.Additionally, theAPIintegratestwoexternalAPIs|AladhanandAlquranAPIcl$

4 Implementation Details

This section provides a detailed examination of the key components and technologies used in implementing the Flask API. It covers the Flask framework, which serves as the foundational structure for the API, and the requests library, which facilitates the making of HTTP requests. The integration of flasgger for Swagger documentation is highlighted, emphasizing its role in simplifying the documentation process with a user-friendly interface. Flask-CORS is introduced to address cross-origin resource sharing, a crucial aspect when dealing with data from various origins. The combination of these technologies and libraries ensures a robust and well-documented implementation of the API.

5 Code Listing: Fetching Prayer Times

```python
import requests
from flask import Flask, jsonify, request
from flasgger import Swagger
from flask_cors import CORS


app = Flask(__name__)
swagger = Swagger(app)
CORS(app)

def fetch_prayer_times(city, country):
    try:
        url = f"http://api.aladhan.com/v1/calendarByCity?city={city}&country={country}&method=2"
        response = requests.get(url)
        info = response.json()

        if "data" in info:
            timing = info["data"][0]["timings"]
            return timing
        else:
            return None

    except Exception as e:
        return f"Unexpected error occurred: {e}"

@app.route('/prayer-times', methods=['GET'])
def get_prayer_times():
    """
    Retrieve prayer times for a specific location
    ---
    parameters:
      - name: city
        in: query
        type: string
        required: true
      - name: country
        in: query
        type: string
        required: true
    responses:
      200:
        description: Prayer times successfully retrieved
    """
    city = request.args.get('city')
    country = request.args.get('country')
    prayer_timings = fetch_prayer_times(city, country)
```

Listing 1: Fetching Prayer Times

Explanation: The `fetch_prayer_times` function sends an HTTP request to the Aladhan API to retrieve prayer times for a specific city and country. The obtained data is then processed, and the timings are extracted and returned. Error handling is implemented to manage unexpected exceptions.

6 Code Listing: Fetching Quran Page

```python
import requests
from flask import Flask, jsonify, request
from flasgger import Swagger

app = Flask(__name__)
swagger = Swagger(app)

def fetch_quran_page(page, edition, offset=None, limit=None):
    try:
        url = f"http://api.alquran.cloud/v1/page/{page}/{edition}"
        params = {'offset': offset, 'limit': limit}
        response = requests.get(url, params=params)

        if response.status_code == 200:
            page_data = response.json()
            return page_data
        else:
            return f"Error: Unable to fetch Quran page data. Status code: {
    response.status_code}"

    except Exception as e:
        return f"Unexpected error occurred: {e}"

@app.route('/quran-page', methods=['GET'])
def quran_page():
    """
    Retrieve a page from a particular Quran edition
    ---
    parameters:
      - name: page
        in: query
        type: integer
        required: true
        description: The page number.
        example: 1
      - name: edition
        in: query
        type: string
        required: true
        description: A Quran edition identifier.
        example: en.asad
      - name: offset
        in: query
        type: integer
        required: false
        description: Offset ayahs in a page by the given number.
      - name: limit
        in: query
        type: integer
        required: false
        description: The number of ayahs that the response will be limited to.

    responses:
      200:
        description: Quran page successfully retrieved
    """
    try:
        page = int(request.args.get('page'))
        edition = request.args.get('edition')
        offset = request.args.get('offset', default=None, type=int)
        limit = request.args.get('limit', default=None, type=int)

        page_data = fetch_quran_page(page, edition, offset, limit)
        return jsonify(page_data)

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

if __name__ == '__main__':
    app.run(debug=True, port=3000)
```
Listing 2: Fetching Quran Page

Explanation: The `fetch_quran_page` function queries the Alquran Cloud API to obtain data for a specific page from a particular Quran edition. The function allows optional parameters for offset and limit, providing flexibility in retrieving a subset of ayahs. Proper error handling is implemented to manage potential issues.

7 Code Listing: Fetching Asma Al Husna

```python
import requests
from flask import Flask, jsonify, request
from flasgger import Swagger

app = Flask(__name__)
swagger = Swagger(app)

def fetch_asma_al_husna(names):
    try:
        url = f"http://api.aladhan.com/v1/asmaAlHusna/{names}"
        response = requests.get(url)

        if response.status_code == 200:
            asma_al_husna_data = response.json()
            return asma_al_husna_data
        else:
            return f"Error: Unable to fetch Asma Al Husna data. Status code: {response.status_code}"

    except Exception as e:
        return f"Unexpected error occurred: {e}"

@app.route('/asma-al-husna', methods=['GET'])
def asma_al_husna():
    """
    Retrieve Asma Al Husna (Names of Allah) with Arabic text, transliteration, and meaning
    ---
    parameters:
      - name: numbers
        in: query
        type: string
        required: false
        description: Names are numbered from 1 to 99. If not specified, all names will be returned.
        example: "1,2,3"

    responses:
      200:
        description: Asma Al Husna successfully retrieved
    """
    try:
        numbers = request.args.get('numbers')

        asma_al_husna_data = fetch_asma_al_husna(numbers)
        return jsonify(asma_al_husna_data)

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

if __name__ == '__main__':
    app.run(debug=True, port=3000)
```

Listing 3: Fetching Asma Al Husna

Explanation: The `fetch_asma_al_husna` function fetches Asma Al Husna (Names of Allah) data from the Aladhan API. The API provides information with Arabic text, transliteration, and meaning. The function handles potential errors and returns the data if successful.

8 Code Listing: Fetching Sajda Ayahs

```python
import requests
from flask import Flask, jsonify, request
from flasgger import Swagger

app = Flask(__name__)
swagger = Swagger(app)

def fetch_sajda_ayahs(edition):
    try:
        url = f"http://api.alquran.cloud/v1/sajda/{edition}"
        response = requests.get(url)

        if response.status_code == 200:
            sajda_data = response.json()
            return sajda_data
        else:
            return f"Error: Unable to fetch sajda ayahs data. Status code: {
    response.status_code}"

    except Exception as e:
        return f"Unexpected error occurred: {e}"

@app.route('/sajda-ayahs', methods=['GET'])
def sajda_ayahs():
    """
    Retrieve sajda ayahs from a particular Quran edition
    ---
    parameters:
      - name: edition
        in: query
        type: string
        required: true
        description: A Quran edition identifier.
        example: en.asad

    responses:
      200:
        description: Sajda ayahs successfully retrieved
    """
    try:
        edition = request.args.get('edition')

        sajda_data = fetch_sajda_ayahs(edition)
        return jsonify(sajda_data)

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

if __name__ == '__main__':
    app.run(debug=True, port=3000)
```

Listing 4: Fetching Sajda Ayahs

Explanation: The `fetch_sajda_ayahs` function retrieves information about Sajda (prostration) ayahs from a particular Quran edition using the Alquran Cloud API. The function handles errors and returns the data if the request is successful.

9 Code Listing: Fetching Calendar by City

```python
from flask import Flask, jsonify, request
from flasgger import Swagger
from flask_cors import CORS
import requests

app = Flask(__name__)
CORS(app)
swagger = Swagger(app)

def fetch_prayer_times_by_city(year, month, city, country):
    try:
        url = f"http://api.aladhan.com/v1/calendarByCity/{year}/{month}"
        params = {
            'city': city,
            'country': country,

        }

        response = requests.get(url, params=params)
        prayer_times_data = response.json()

        if "data" in prayer_times_data:
            return prayer_times_data["data"]
        else:
            return None

    except Exception as e:
        return f"Unexpected error occurred: {e}"

@app.route('/calendar-by-city', methods=['GET'])
def get_prayer_times_by_city():
    """
    Retrieve prayer times for a specific calendar month by city
    ---
    parameters:
      - name: year
        in: query
        type: integer
        required: true
      - name: month
        in: query
        type: integer
        required: true
      - name: city
        in: query
        type: string
        required: true
      - name: country
        in: query
        type: string
        required: true
    responses:
      200:
        description: Prayer times successfully retrieved
    """
    try:
        year = int(request.args.get('year'))
        month = int(request.args.get('month'))
        city = request.args.get('city')
        country = request.args.get('country')

        prayer_times = fetch_prayer_times_by_city(year, month, city, country)
        return jsonify(prayer_times)

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

if __name__ == "__main__":
    app.run(debug=True, port=3000)
```

Listing 5: Fetching Calendar by City

Explanation: The `fetch_prayer_times_by_city` function retrieves prayer times for a specific calendar month by city using the Aladhan API. The function takes parameters such as year, month, city, and country to customize the request.

10 Code Listing: Fetching Quran Edition

```python
import requests
from flask import Flask, jsonify, request
from flasgger import Swagger

app = Flask(__name__)
swagger = Swagger(app)

def get_quran_edition(edition):
    try:
        url = f"http://api.alquran.cloud/v1/quran/{edition}"
        response = requests.get(url)
        if response.status_code == 200:
            quran_data = response.json()
            return quran_data
        else:
            return f"Error: Unable to fetch Quran data. Status code: {response.
    status_code}"

    except Exception as e:
        return f"Unexpected error occurred: {e}"

@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

if __name__ == "__main__":
    app.run(debug=True, port=3000)
```

Listing 6: Fetching Quran Edition

Explanation: The `get_quran_edition` function fetches information about a specific Quran edition in text format from the Alquran Cloud API. The function handles errors and returns the data if the request is successful.

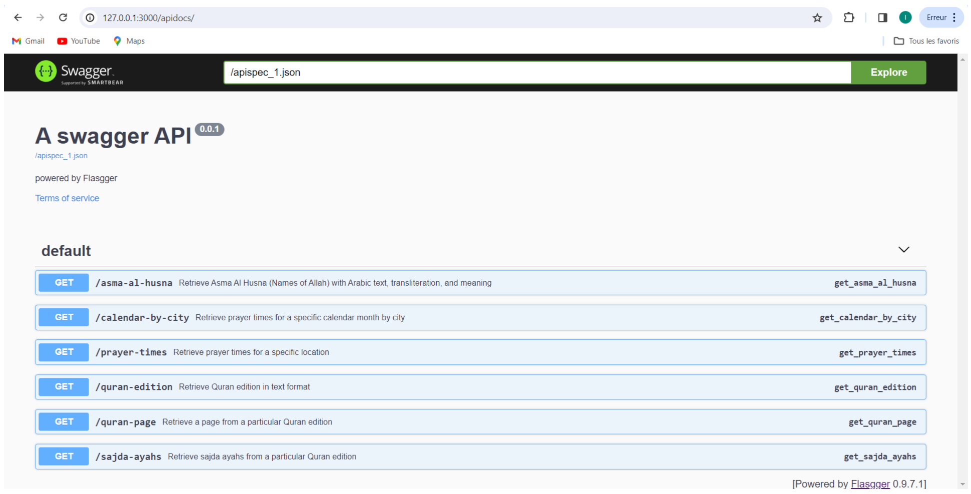
11 API Documentation

The API documentation is automatically generated using Swagger. Developers can refer to the Swagger UI for detailed information on each endpoint, including parameters, responses, and example usage. The output of the execution has JSON as a file format. The Swagger documentation is accessible at the following endpoint:

```
http://localhost:3000/apidocs
```

12 Results

This section presents the results obtained by using the Flask API on the Swagger Interface. It includes sample outputs showcasing the functionality and accuracy of the API in providing Islamic information.


127.0.0.1:3000/apidocs/

/apispec_1.json Explore

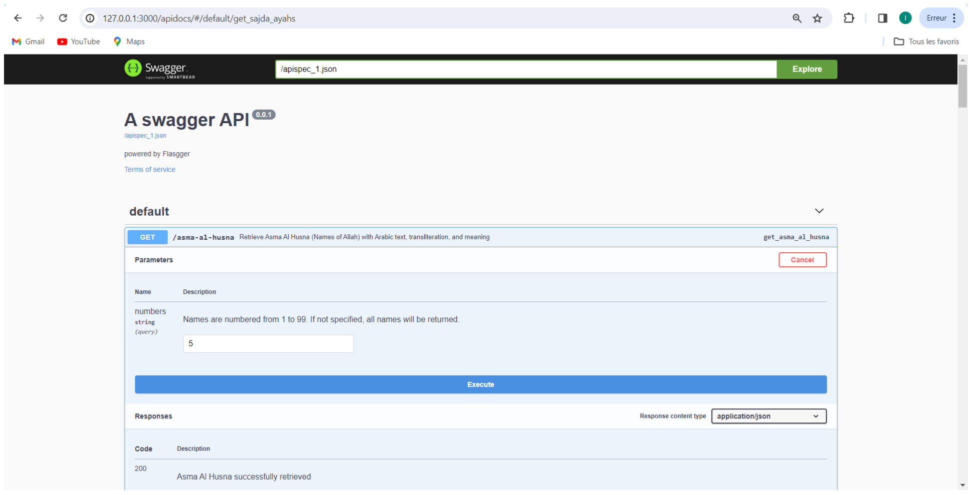
A swagger API 0.0.1
/apispec_1.json
powered by Flasgger
Terms of service

default

GET /asma-al-husna Retrieve Asma Al Husna (Names of Allah) with Arabic text, transliteration, and meaning get_asma_al_husna
GET /calendar-by-city Retrieve prayer times for a specific calendar month by city get_calendar_by_city
GET /prayer-times Retrieve prayer times for a specific location get_prayer_times
GET /quran-edition Retrieve Quran edition in text format get_quran_edition
GET /quran-page Retrieve a page from a particular Quran edition get_quran_page
GET /sajda-ayahs Retrieve sajda ayahs from a particular Quran edition get_sajda_ayahs

[Powered by Flasgger 0.9.7.1]

Figure 1: Swagger Interface:


127.0.0.1:3000/apidocs/#/default/get_sajda_ayahs

/apispec_1.json Explore

A swagger API 0.0.1
/apispec_1.json
powered by Flasgger
Terms of service

default

GET /asma-al-husna Retrieve Asma Al Husna (Names of Allah) with Arabic text, transliteration, and meaning get_asma_al_husna

Parameters Cancel

Name	Description
numbers string (query)	Names are numbered from 1 to 99. If not specified, all names will be returned. 5

Execute

Responses Response content type application/json

Code	Description
200	Asma Al Husna successfully retrieved

Figure 2: Input of the 5th of Allah:

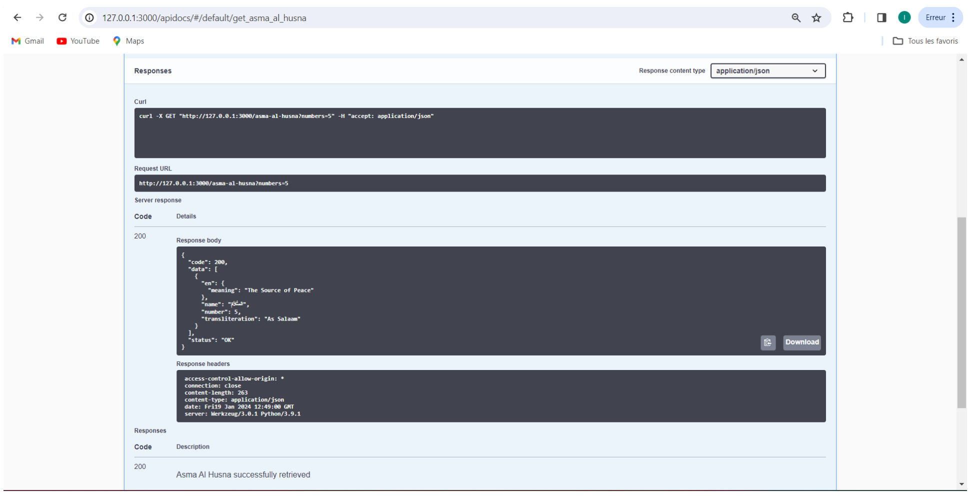
13 Front-End Implementation: HTML and JavaScript

The front end of the application is implemented using HTML, CSS, and JavaScript to create an intuitive and user-friendly interface for interacting with the Flask API. The following sections explain the HTML structure and JavaScript functions used in the front end.

13.1 HTML Code

The HTML code defines the structure of the web page and includes input fields, buttons, and result containers for each API endpoint.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>API INTERFACE</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="container">
```



127.0.0.1:3000/apidocs/#/default/get_asma_al_husna

Responses — Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/asma-al-husna?numbers=5" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/asma-al-husna?numbers=5
```

Server response

Code	Details
200	Response body

```
{
  "code": 200,
  "data": [
    {
      "en": {
        "meaning": "The Source of Peace"
      },
      "name": "السَّلَامُ",
      "number": 5,
      "transliteration": "As Salaam"
    }
  ],
  "status": "OK"
}
```

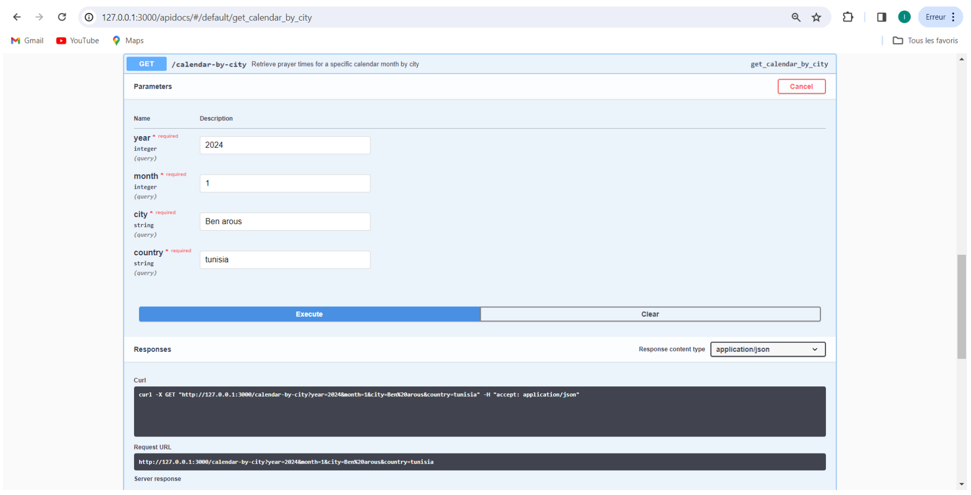
Response headers

```
access-control-allow-origin: *
connection: close
content-length: 263
content-type: application/json
date: Fri19 Jan 2024 12:49:00 GMT
server: Werkzeug/3.0.1 Python/3.9.1
```

Responses

Code	Description
200	Asma Al Husna successfully retrieved

Figure 3: Output of the 5th name of Allah:



127.0.0.1:3000/apidocs/#/default/get_calendar_by_city

GET /calendar-by-city Retrieve prayer times for a specific calendar month by city — get_calendar_by_city

Parameters

Name	Description
year * required, integer (query)	2024
month * required, integer (query)	1
city * required, string (query)	Ben arous
country * required, string (query)	tunisia

Responses — Response content type: application/json

Curl

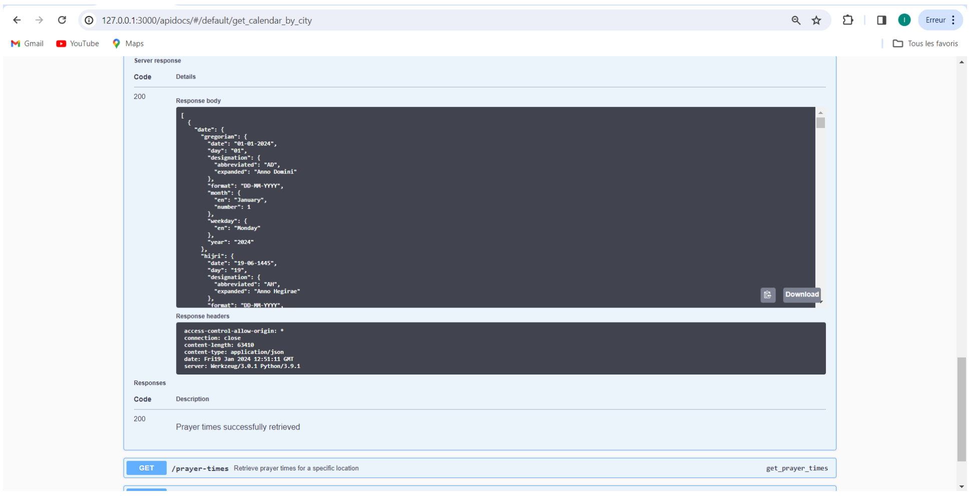
```
curl -X GET "http://127.0.0.1:3000/calendar-by-city?year=2024&month=1&city=Ben%20arous&country=tunisia" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/calendar-by-city?year=2024&month=1&city=Ben%20arous&country=tunisia
```

Server response

Figure 4: Input of Calendar by City on Swagger:



127.0.0.1:3000/apidocs/#/default/get_calendar_by_city

Server response

Code	Details
200	Response body

```
[
  {
    "date": {
      "gregorian": {
        "date": "01-01-2024",
        "day": "01",
        "designation": {
          "abbreviated": "AD",
          "expanded": "Anno Domini"
        },
        "format": "DD-MM-YYYY",
        "month": {
          "en": "January",
          "number": 1
        },
        "weekday": {
          "en": "Monday"
        },
        "year": "2024"
      },
      "hijri": {
        "date": "19-06-1445",
        "day": "19",
        "designation": {
          "abbreviated": "AH",
          "expanded": "Anno Hegirae"
        },
        "format": "DD-MM-YYYY",
```

Response headers

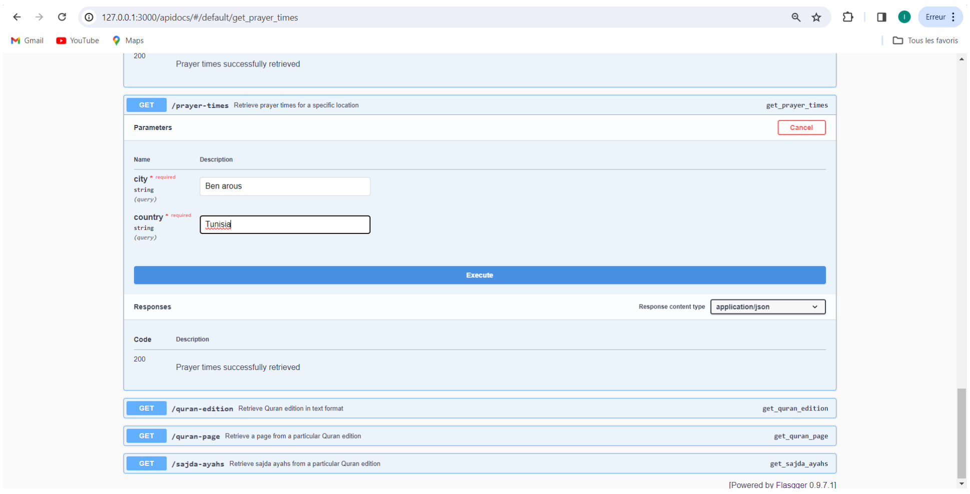
```
access-control-allow-origin: *
connection: close
content-length: 63410
content-type: application/json
date: Fri19 Jan 2024 12:51:11 GMT
server: Werkzeug/3.0.1 Python/3.9.1
```

Responses

Code	Description
200	Prayer times successfully retrieved

GET /prayer-times Retrieve prayer times for a specific location — get_prayer_times

Figure 5: Output of Calendar by City on Swagger:


127.0.0.1:3000/apidocs/#/default/get_prayer_times

200 Prayer times successfully retrieved

GET /prayer-times Retrieve prayer times for a specific location get_prayer_times

Parameters

Name	Description
city * required string (query)	Ben arous
country * required string (query)	Tunisia

Execute

Responses — Response content type: application/json

Code	Description
200	Prayer times successfully retrieved

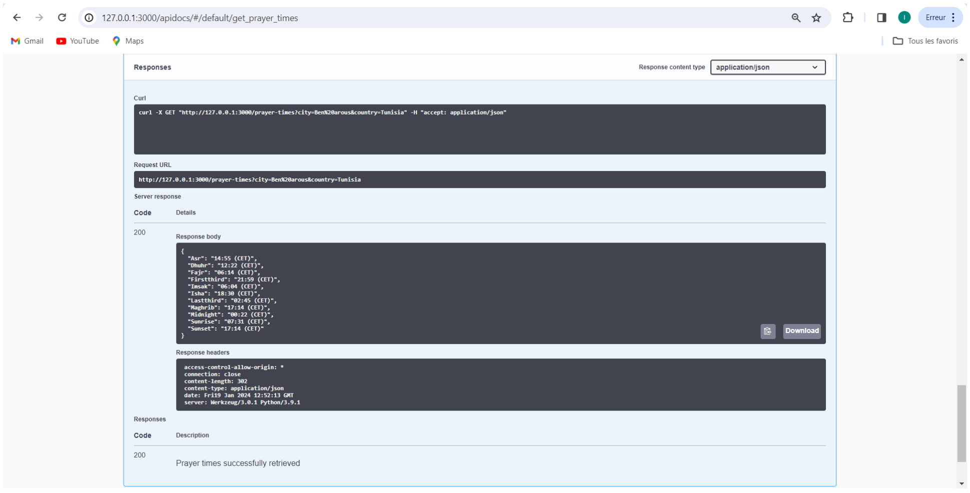
GET /quran-edition Retrieve Quran edition in text format get_quran_edition

GET /quran-page Retrieve a page from a particular Quran edition get_quran_page

GET /sajda-ayahs Retrieve sajda ayahs from a particular Quran edition get_sajda_ayahs

[Powered by Flasgger 0.9.7.1]

Figure 6: Input of prayer times:


127.0.0.1:3000/apidocs/#/default/get_prayer_times

Responses — Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/prayer-times?city=Ben%20arous&country=Tunisia" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/prayer-times?city=Ben%20arous&country=Tunisia
```

Server response

Code: 200

Response body

```
{
  "Asr": "14:55 (CET)",
  "Dhuhr": "12:22 (CET)",
  "Fajr": "06:14 (CET)",
  "Firstthird": "21:59 (CET)",
  "Imsak": "06:04 (CET)",
  "Isha": "18:30 (CET)",
  "Lastthird": "02:45 (CET)",
  "Maghrib": "17:14 (CET)",
  "Midnight": "00:22 (CET)",
  "Sunrise": "07:31 (CET)",
  "Sunset": "17:14 (CET)"
}
```

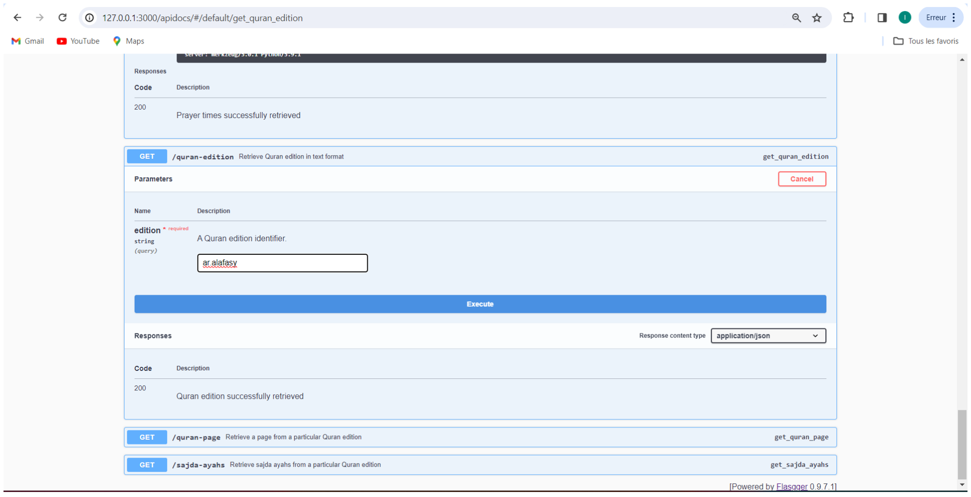
Response headers

```
access-control-allow-origin: *
connection: close
content-length: 302
content-type: application/json
date: Fri19 Jan 2024 12:52:13 GMT
server: Werkzeug/3.0.1 Python/3.9.1
```

Responses

Code	Description
200	Prayer times successfully retrieved

Figure 7: Output of prayer times:


127.0.0.1:3000/apidocs/#/default/get_quran_edition

Responses

Code	Description
200	Prayer times successfully retrieved

GET /quran-edition Retrieve Quran edition in text format get_quran_edition

Parameters

Name	Description
edition * required string (query)	A Quran edition identifier. ar.alafasy

Execute

Responses — Response content type: application/json

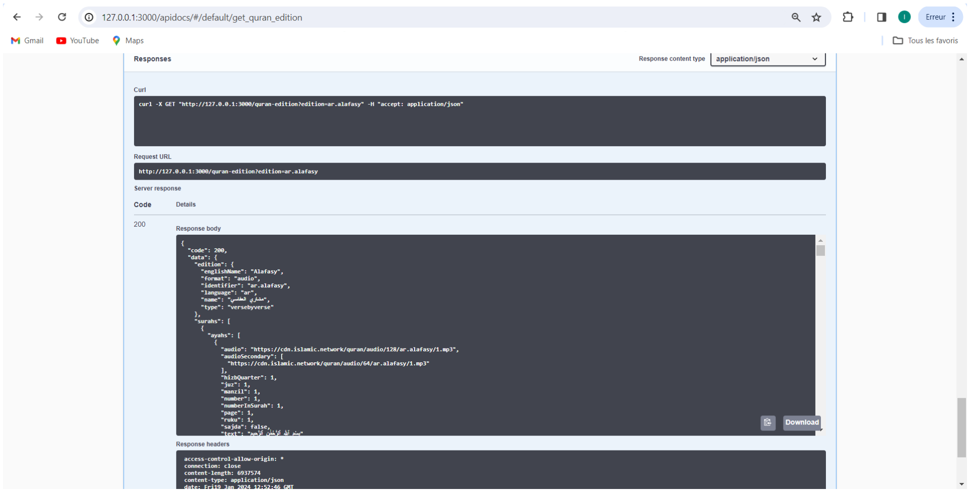
Code	Description
200	Quran edition successfully retrieved

GET /quran-page Retrieve a page from a particular Quran edition get_quran_page

GET /sajda-ayahs Retrieve sajda ayahs from a particular Quran edition get_sajda_ayahs

[Powered by Flasgger 0.9.7.1]

Figure 8: Input of Quran Edition:



127.0.0.1:3000/apidocs/#/default/get_quran_edition

Responses

Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/quran-edition?edition=ar.alafasy" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/quran-edition?edition=ar.alafasy
```

Server response

Code: 200

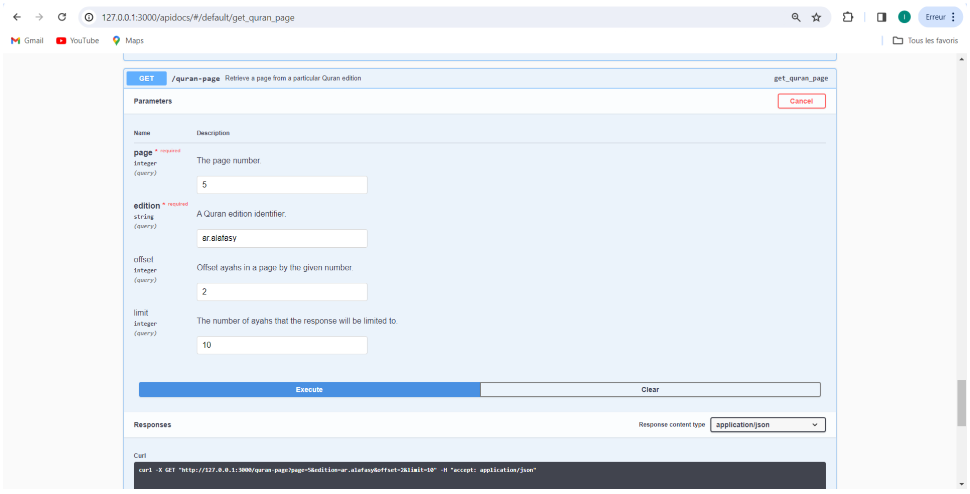
Response body

```
{
  "code": 200,
  "data": {
    "edition": {
      "englishName": "Alafasy",
      "format": "audio",
      "identifier": "ar.alafasy",
      "language": "ar",
      "name": "مشاري العفاسي",
      "type": "versebyverse"
    },
    "surahs": [
      {
        "ayahs": [
          {
            "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/1.mp3",
            "audioSecondary": [
              "https://cdn.islamic.network/quran/audio/64/ar.alafasy/1.mp3"
            ],
            "hizbQuarter": 1,
            "juz": 1,
            "manzil": 1,
            "number": 1,
            "numberInSurah": 1,
            "page": 1,
            "ruku": 1,
            "sajda": false,
            "text": "بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ"
```

Response headers

```
access-control-allow-origin: *
connection: close
content-length: 6937574
content-type: application/json
date: Fri19 Jan 2024 12:52:46 GMT
```

Figure 9: Output of Quran Edition:



127.0.0.1:3000/apidocs/#/default/get_quran_page

GET /quran-page Retrieve a page from a particular Quran edition — get_quran_page

Parameters

Name	Description
page * required, integer (query)	The page number. 5
edition * required, string (query)	A Quran edition identifier. ar.alafasy
offset, integer (query)	Offset ayahs in a page by the given number. 2
limit, integer (query)	The number of ayahs that the response will be limited to. 10

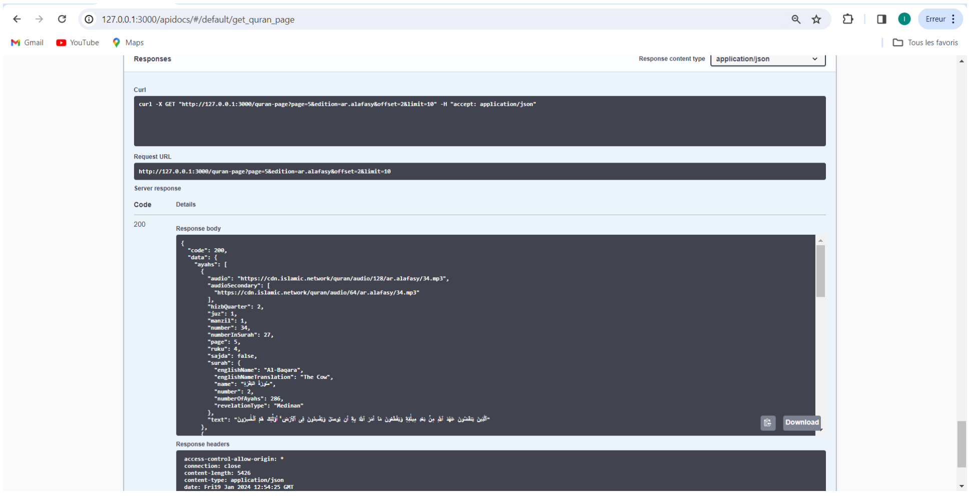
Responses

Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/quran-page?page=5&edition=ar.alafasy&offset=2&limit=10" -H "accept: application/json"
```

Figure 10: Input of Quran page:



127.0.0.1:3000/apidocs/#/default/get_quran_page

Responses

Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/quran-page?page=5&edition=ar.alafasy&offset=2&limit=10" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/quran-page?page=5&edition=ar.alafasy&offset=2&limit=10
```

Server response

Code: 200

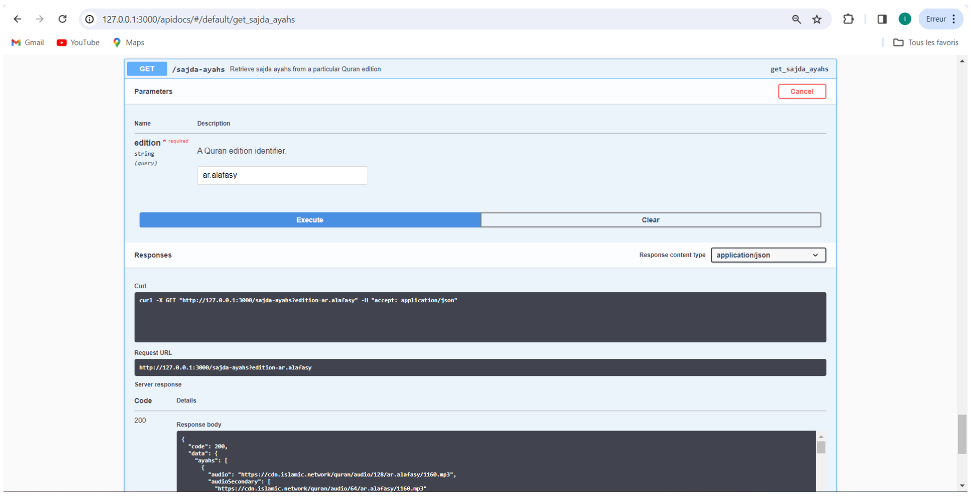
Response body

```
{
  "code": 200,
  "data": {
    "ayahs": [
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/34.mp3",
        "audioSecondary": [
          "https://cdn.islamic.network/quran/audio/64/ar.alafasy/34.mp3"
        ],
        "hizbQuarter": 2,
        "juz": 1,
        "manzil": 1,
        "number": 34,
        "numberInSurah": 27,
        "page": 5,
        "ruku": 4,
        "sajda": false,
        "surah": {
          "englishName": "Al-Baqara",
          "englishNameTranslation": "The Cow",
          "name": "سُورَةُ البَقَرَةِ",
          "number": 2,
          "numberOfAyahs": 286,
          "revelationType": "Medinan"
        },
        "text": "ٱلَّذِينَ يَنقُضُونَ عَهْدَ ٱللَّهِ مِنۢ بَعْدِ مِيثَٰقِهِۦ وَيَقْطَعُونَ مَآ أَمَرَ ٱللَّهُ بِهِۦٓ أَن يُوصَلَ وَيُفْسِدُونَ فِى ٱلْأَرْضِ ۚ أُو۟لَٰٓئِكَ هُمُ ٱلْخَٰسِرُونَ"
      },
      {
```

Response headers

```
access-control-allow-origin: *
connection: close
content-length: 5426
content-type: application/json
date: Fri19 Jan 2024 12:54:25 GMT
```

Figure 11: Output of Quran page:



`/sajda-ayahs` Retrieve sajda ayahs from a particular Quran edition `get_sajda_ayahs`

Parameters

Name	Description
edition * required string (query)	A Quran edition identifier. ar.alafasy

Responses

Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/sajda-ayahs?edition=ar.alafasy" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/sajda-ayahs?edition=ar.alafasy
```

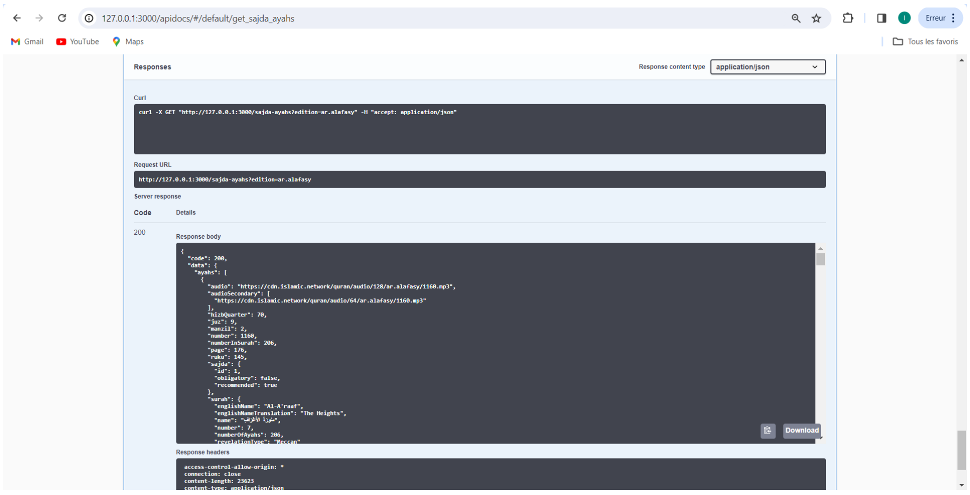
Server response

Code Details

200 Response body

```
{
  "code": 200,
  "data": {
    "ayahs": [
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/1160.mp3",
        "audioSecondary": [
          "https://cdn.islamic.network/quran/audio/64/ar.alafasy/1160.mp3"
```

Figure 12: Input of Sajdah ayah:



Responses

Response content type: application/json

Curl

```
curl -X GET "http://127.0.0.1:3000/sajda-ayahs?edition=ar.alafasy" -H "accept: application/json"
```

Request URL

```
http://127.0.0.1:3000/sajda-ayahs?edition=ar.alafasy
```

Server response

Code Details

200 Response body

```
{
  "code": 200,
  "data": {
    "ayahs": [
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/1160.mp3",
        "audioSecondary": [
          "https://cdn.islamic.network/quran/audio/64/ar.alafasy/1160.mp3"
        ],
        "hizbQuarter": 70,
        "juz": 9,
        "manzil": 2,
        "number": 1160,
        "numberInSurah": 206,
        "page": 176,
        "ruku": 145,
        "sajda": {
          "id": 1,
          "obligatory": false,
          "recommended": true
        },
        "surah": {
          "englishName": "Al-A'raaf",
          "englishNameTranslation": "The Heights",
          "name": "سُورَةُ الأَعْرَافِ",
          "number": 7,
          "numberOfAyahs": 206,
          "revelationType": "Meccan"
```

Response headers

```
access-control-allow-origin: *
connection: close
content-length: 23623
content-type: application/json
```

Figure 13: Output of Sajdah ayah:

```
<h1>WEB SERVICE (IT 325) API PROJECT (Religion Related API Interface)</h1>


<div class="endpoint">
    <h2>Get Quran Edition</h2>
    <label for="quranEdition">Enter Quran Edition:</label>
    <input type="text" id="quranEdition" placeholder="Enter Quran Edition (e.g., en.asad)">
    <button onclick="getQuranEdition()">Get Quran Edition</button>
    <div id="resultQuranEdition"></div>
</div>

<div class="endpoint">
    <h2>Get Quran Page</h2>
    <label for="page">Enter Quran Page Number:</label>
    <input type="number" id="page" placeholder="Enter Quran Page Number">
    <label for="pageEdition">Enter Quran Edition:</label>
    <input type="text" id="pageEdition" placeholder="Enter Quran Edition (e.g., en.asad)">
    <button onclick="getQuranPage()">Get Quran Page</button>
    <div id="resultQuranPage"></div>
</div>

<div class="endpoint">
    <h2>Get Sajda Ayahs</h2>
    <label for="sajdaEdition">Enter Quran Edition:</label>
    <input type="text" id="sajdaEdition" placeholder="Enter Quran Edition (e.g., en.asad)">
    <button onclick="getSajdaAyahs()">Get Sajda Ayahs</button>
    <div id="resultSajdaAyahs"></div>
</div>

<div class="endpoint">
    <h2>Get Asma Al Husna</h2>
    <label for="asmaNumbers">Enter Asma Al Husna Numbers:</label>
    <input type="text" id="asmaNumbers" placeholder="Enter Asma Al Husna Numbers (e.g., 1,2,
    <button onclick="getAsmaAlHusna()">Get Asma Al Husna</button>
    <div id="resultAsmaAlHusna"></div>
</div>

<div class="endpoint">
    <h2>Get Prayer Times</h2>
    <label for="prayerCity">Enter City:</label>
    <input type="text" id="prayerCity" placeholder="Enter City">
    <label for="prayerCountry">Enter Country:</label>
    <input type="text" id="prayerCountry" placeholder="Enter Country">
    <button onclick="getPrayerTimes()">Get Prayer Times</button>
    <div id="resultPrayerTimes"></div>
</div>

<div class="endpoint">
    <h2>Get Calendar by City</h2>
    <label for="calendarYear">Enter Year:</label>
    <input type="number" id="calendarYear" placeholder="Enter Year">
    <label for="calendarMonth">Enter Month:</label>
    <input type="number" id="calendarMonth" placeholder="Enter Month">
    <label for="calendarCity">Enter City:</label>
    <input type="text" id="calendarCity" placeholder="Enter City">
    <label for="calendarCountry">Enter Country:</label>
    <input type="text" id="calendarCountry" placeholder="Enter Country">
    <button onclick="getCalendarByCity()">Get Calendar by City</button>
```

```
            <div id="resultCalendarByCity"></div>
        </div>

    </div>

    <script src="script.js"></script>
</body>
</html}
```

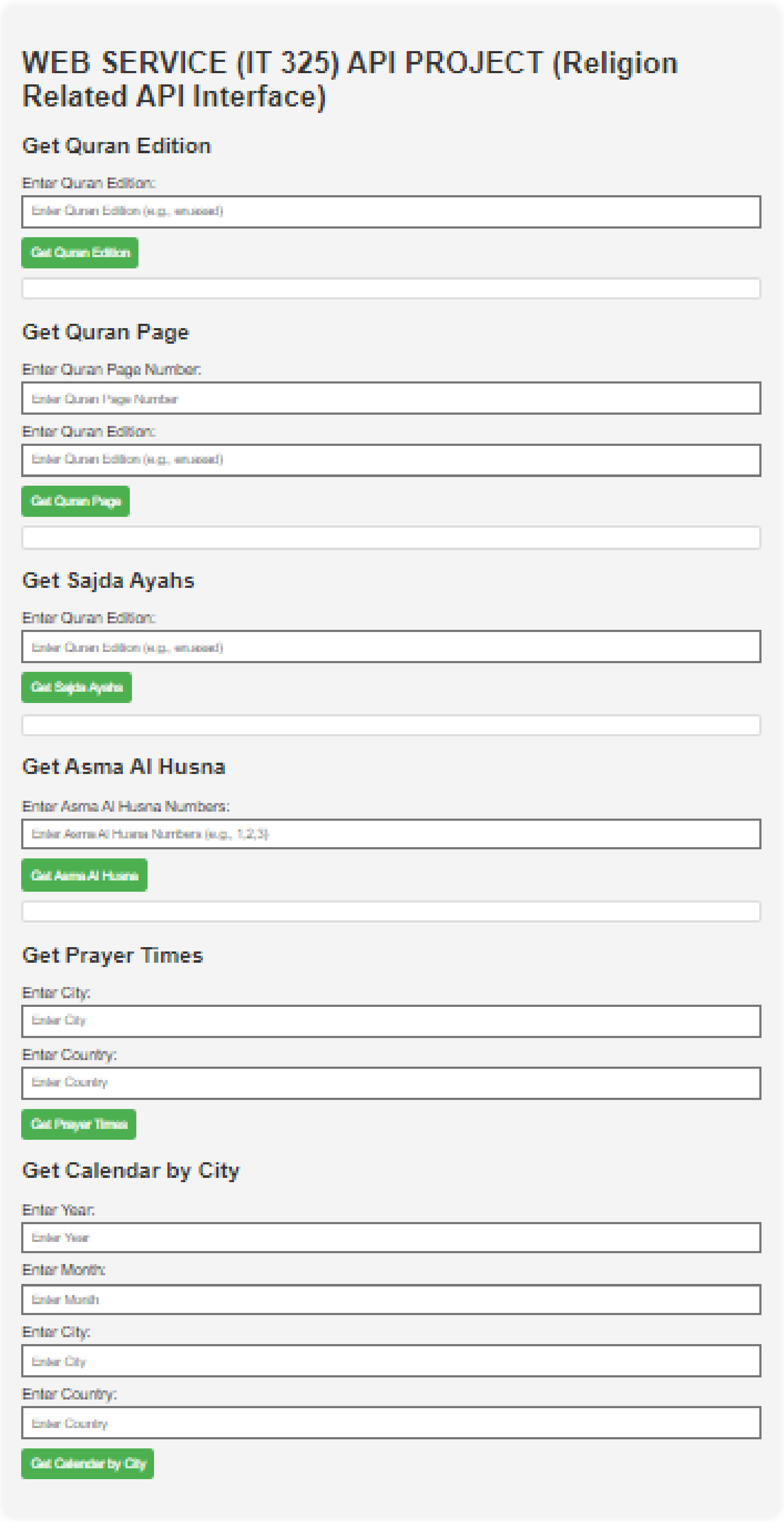
13.2 JavaScript Code

The JavaScript code includes functions for each API endpoint, making asynchronous requests to the Flask server using the Fetch API. Each function handles the response and updates the corresponding result container in the HTML.

```
// JavaScript functions for API interactions
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>`;
        });
}

function getQuranPage() {
    const pageInput = document.getElementById('page');
    const editionInput = document.getElementById('pageEdition');

    const page = pageInput.value;
    const edition = editionInput.value;

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

function getSajdaAyahs() {
    const editionInput = document.getElementById('sajdaEdition');
    const edition = editionInput.value;

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

function getAsmaAlHusna() {
    const numbersInput = document.getElementById('asmaNumbers');
    const numbers = numbersInput.value;

    fetch(`http://localhost:3000/asma-al-husna?numbers=${numbers}`)
        .then(response => {
            if (!response.ok) {
                throw new Error(`Error: ${response.status}`);
            }
            return response.json();
        })
        .then(data => {
            const resultDiv = document.getElementById('resultAsmaAlHusna');
            resultDiv.innerHTML = `<pre>${JSON.stringify(data, null, 2)}</pre>`;
        })
        .catch(error => {
            console.error(`Error: ${error.message}`);
            const resultDiv = document.getElementById('resultAsmaAlHusna');
            resultDiv.innerHTML = `<p>Error: ${error.message}</p>`;
        });
}
function getPrayerTimes() {
    const cityInput = document.getElementById('prayerCity');
    const countryInput = document.getElementById('prayerCountry');

    const city = cityInput.value;
    const country = countryInput.value;

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

function getCalendarByCity() {
    const yearInput = document.getElementById('calendarYear');
    const monthInput = document.getElementById('calendarMonth');
    const cityInput = document.getElementById('calendarCity');
    const countryInput = document.getElementById('calendarCountry');

    const year = yearInput.value;
    const month = monthInput.value;
    const city = cityInput.value;
    const country = countryInput.value;

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

These JavaScript functions handle user input, make API requests, and update the result containers on the web page. The Fetch API is used for asynchronous communication with the Flask server, ensuring a responsive user interface.

13.3 Results After The Implementation of The Front End Part



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

Get Quran Edition

Enter Quran Edition:

Enter Quran Edition (e.g., en.asad)

Get Quran Edition

Get Quran Page

Enter Quran Page Number:

Enter Quran Page Number

Enter Quran Edition:

Enter Quran Edition (e.g., en.asad)

Get Quran Page

Get Sajda Ayahs

Enter Quran Edition:

Enter Quran Edition (e.g., en.asad)

Get Sajda Ayahs

Get Asma Al Husna

Enter Asma Al Husna Numbers:

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

Get Asma Al Husna

Get Prayer Times

Enter City:

Enter City

Enter Country:

Enter Country

Get Prayer Times

Get Calendar by City

Enter Year:

Enter Year

Enter Month:

Enter Month

Enter City:

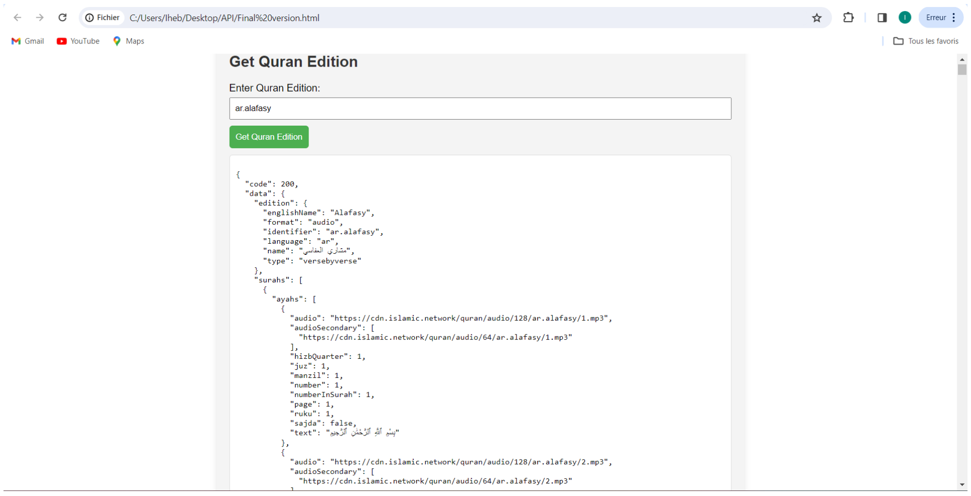
Enter City

Enter Country:

Enter Country

Get Calendar by City

Figure 14: Front end Interface:



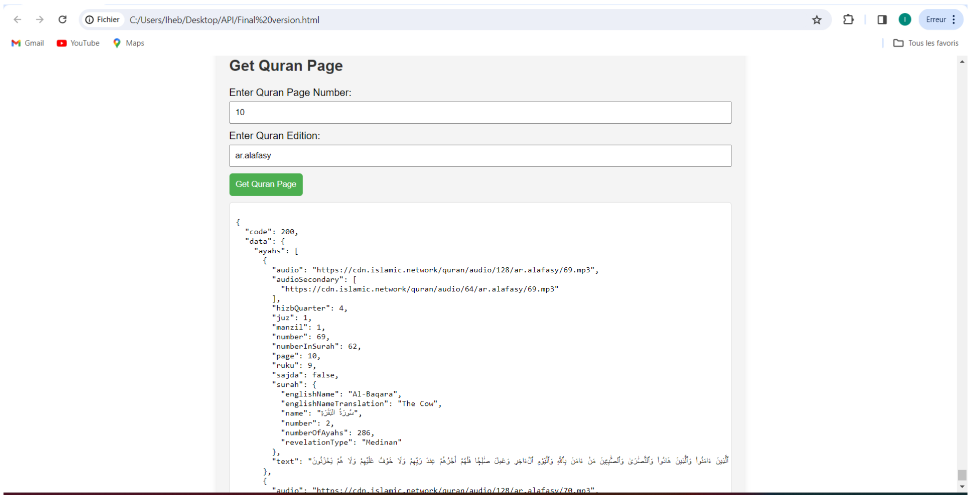
Get Quran Edition

Enter Quran Edition: ar.alafasy

Get Quran Edition

```
{
  "code": 200,
  "data": {
    "edition": {
      "englishName": "Alafasy",
      "format": "audio",
      "identifier": "ar.alafasy",
      "language": "ar",
      "name": "مشاري العفاسي",
      "type": "versebyverse"
    },
    "surahs": [
      {
        "ayahs": [
          {
            "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/1.mp3",
            "audioSecondary": [
              "https://cdn.islamic.network/quran/audio/64/ar.alafasy/1.mp3"
            ],
            "hizbQuarter": 1,
            "juz": 1,
            "manzil": 1,
            "number": 1,
            "numberInSurah": 1,
            "page": 1,
            "ruku": 1,
            "sajda": false,
            "text": "بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ"
          },
          {
            "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/2.mp3",
            "audioSecondary": [
              "https://cdn.islamic.network/quran/audio/64/ar.alafasy/2.mp3"
```

Figure 15: Output of Quran Edition:



Get Quran Page

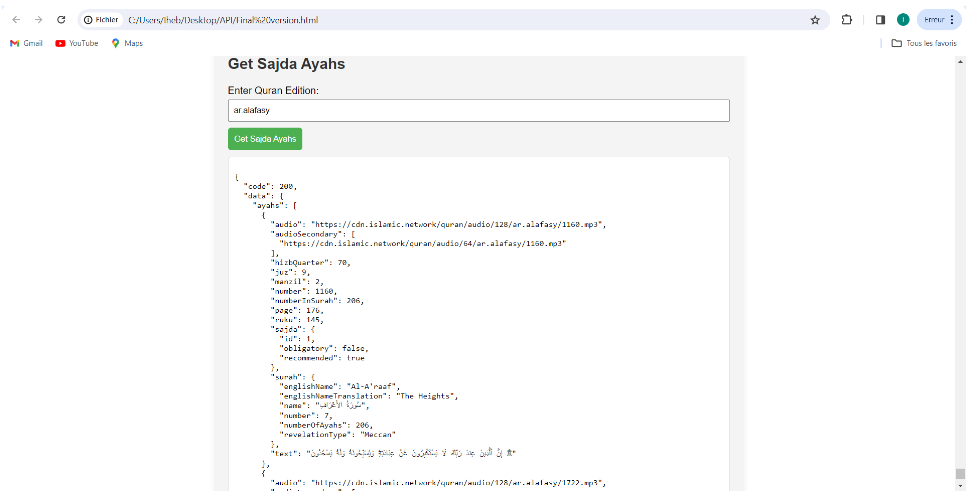
Enter Quran Page Number: 10

Enter Quran Edition: ar.alafasy

Get Quran Page

```
{
  "code": 200,
  "data": {
    "ayahs": [
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/69.mp3",
        "audioSecondary": [
          "https://cdn.islamic.network/quran/audio/64/ar.alafasy/69.mp3"
        ],
        "hizbQuarter": 4,
        "juz": 1,
        "manzil": 1,
        "number": 69,
        "numberInSurah": 62,
        "page": 10,
        "ruku": 9,
        "sajda": false,
        "surah": {
          "englishName": "Al-Baqara",
          "englishNameTranslation": "The Cow",
          "name": "سُورَةُ ٱلْبَقَرَةِ",
          "number": 2,
          "numberOfAyahs": 286,
          "revelationType": "Medinan"
        },
        "text": "ٱلَّذِينَ ءَامَنُوا۟ وَٱلَّذِينَ هَادُوا۟ وَٱلنَّصَٰرَىٰ وَٱلصَّٰبِـِٔينَ مَنْ ءَامَنَ بِٱللَّهِ وَٱلْيَوْمِ ٱلْءَاخِرِ وَعَمِلَ صَٰلِحًا فَلَهُمْ أَجْرُهُمْ عِندَ رَبِّهِمْ وَلَا خَوْفٌ عَلَيْهِمْ وَلَا هُمْ يَحْزَنُونَ"
      },
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/70.mp3",
```

Figure 16: Output of Quran Page :



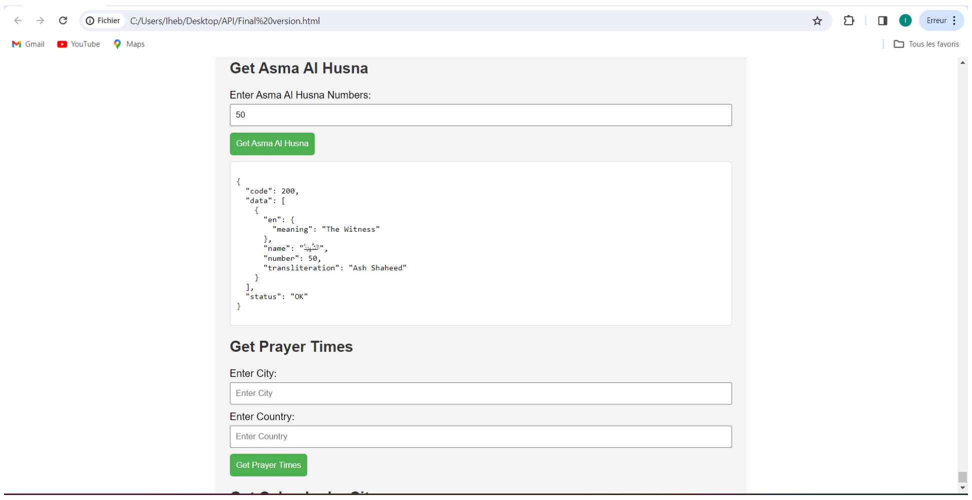
Get Sajda Ayahs

Enter Quran Edition: ar.alafasy

Get Sajda Ayahs

```
{
  "code": 200,
  "data": {
    "ayahs": [
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/1160.mp3",
        "audioSecondary": [
          "https://cdn.islamic.network/quran/audio/64/ar.alafasy/1160.mp3"
        ],
        "hizbQuarter": 70,
        "juz": 9,
        "manzil": 2,
        "number": 1160,
        "numberInSurah": 206,
        "page": 176,
        "ruku": 145,
        "sajda": {
          "id": 1,
          "obligatory": false,
          "recommended": true
        },
        "surah": {
          "englishName": "Al-A'raaf",
          "englishNameTranslation": "The Heights",
          "name": "سُورَةُ ٱلْأَعْرَافِ",
          "number": 7,
          "numberOfAyahs": 206,
          "revelationType": "Meccan"
        },
        "text": "إِنَّ ٱلَّذِينَ عِندَ رَبِّكَ لَا يَسْتَكْبِرُونَ عَنْ عِبَادَتِهِۦ وَيُسَبِّحُونَهُۥ وَلَهُۥ يَسْجُدُونَ ۩"
      },
      {
        "audio": "https://cdn.islamic.network/quran/audio/128/ar.alafasy/1722.mp3",
```

Figure 17: Output of Sajdah ayah:



Get Asma Al Husna

Enter Asma Al Husna Numbers: 50

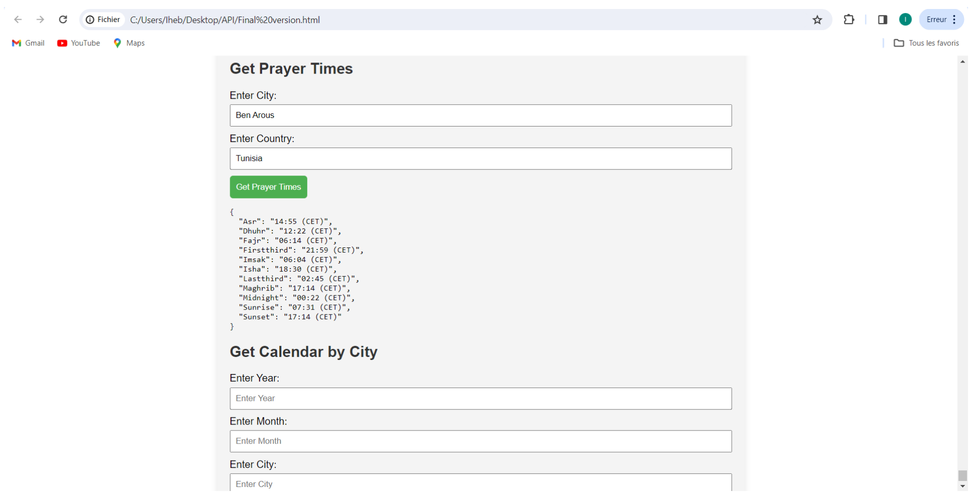
```
{
  "code": 200,
  "data": [
    {
      "en": {
        "meaning": "The Witness"
      },
      "name": "الشَّهِيدُ",
      "number": 50,
      "transliteration": "Ash Shaheed"
    }
  ],
  "status": "OK"
}
```

Get Prayer Times

Enter City: Enter City

Enter Country: Enter Country

Figure 18: Output of Asmaa al Husna:



Get Prayer Times

Enter City: Ben Arous

Enter Country: Tunisia

```
{
  "Asr": "14:55 (CET)",
  "Dhuhr": "12:22 (CET)",
  "Fajr": "06:14 (CET)",
  "Firstthird": "21:59 (CET)",
  "Imsak": "06:04 (CET)",
  "Isha": "18:30 (CET)",
  "Lastthird": "02:45 (CET)",
  "Maghrib": "17:14 (CET)",
  "Midnight": "00:22 (CET)",
  "Sunrise": "07:31 (CET)",
  "Sunset": "17:14 (CET)"
}
```

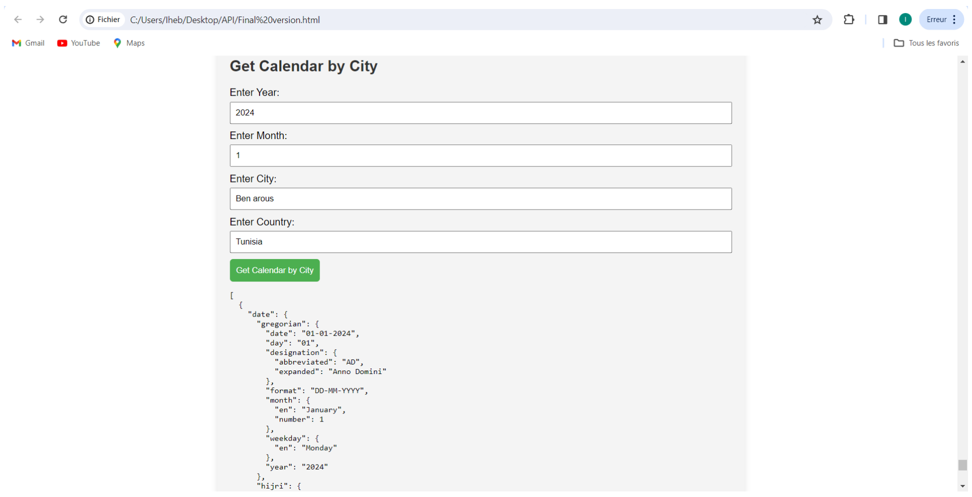
Get Calendar by City

Enter Year: Enter Year

Enter Month: Enter Month

Enter City: Enter City

Figure 19: Output of prayer times:



Get Calendar by City

Enter Year: 2024

Enter Month: 1

Enter City: Ben arous

Enter Country: Tunisia

```
[
  {
    "date": {
      "gregorian": {
        "date": "01-01-2024",
        "day": "01",
        "designation": {
          "abbreviated": "AD",
          "expanded": "Anno Domini"
        },
        "format": "DD-MM-YYYY",
        "month": {
          "en": "January",
          "number": 1
        },
        "weekday": {
          "en": "Monday"
        },
        "year": "2024"
      },
      "hijri": {
```

Figure 20: Output of calendar by city:

14 Conclusion

The conclusion summarizes the key findings and contributions of the Flask API for Islamic information. It reiterates the importance of the API and its potential impact on developers and applications. It also facilitates the process of finding data that should be updated on a daily basis. I have surely some future perspectives to work on for this API. So far, this API will be helpful for simple tasks of finding some relevant Islamic information. lastly, I am grateful to our professor Dr.Montassar Ben Messaoud who paved our learning path to lead us here, and I would like to personally thank him for his contribution and guidance.