



[스파르타코딩클럽] 웹개발 플러스 - 2주차



매 주차 강의자료 시작에 PDF파일을 올려두었어요!

▼ PDF 파일

▼ 단축키 모음

▼ 새로고침

- `F5`

▼ 저장

- Windows: `Ctrl` + `S`
- macOS: `command` + `S`

▼ 전체선택

- Windows: `Ctrl` + `A`
- macOS: `command` + `A`

▼ 잘라내기

- Windows: `Ctrl` + `X`
- macOS: `command` + `X`

▼ 콘솔창 줄바꿈

- `shift` + `enter`

▼ 코드정렬

- Windows: `Ctrl` + `Alt` + `L`
- macOS: `option` + `command` + `L`

▼ 들여쓰기

- `Tab`
- 들여쓰기 취소 : `Shift` + `Tab`

▼ 주석

- Windows: `Ctrl` + `/`
- macOS: `command` + `/`

[수업 목표]

1. Flask의 더 많은 기능을 배운다.
2. API에 보안 키를 전달할 수 있다.
3. 멀티페이지 사이트를 만들 수 있다!

[목차]

01. 2주차 이번주에 배울 것

- 02. 플라스크로 멀티페이지 사이트 만들기 - 기초
- 03. 플라스크로 멀티페이지 사이트 만들기 - 응용
- 04. 사전 API 사용하기
- 05. 프로젝트 2: 나만의 단어장

06. 프로젝트 세팅
07. 상세 페이지 전체 모습 만들기
08. 상세 페이지 - Ajax로 단어 뜻 가져오기
09. 상세 페이지 - jinja2로 뜻 가져오기
10. 상세 페이지 - 새 단어/기존 단어 구분하기
11. 상세 페이지 - 저장 & 삭제 기능 만들기
12. 목록 페이지 - 전체 모습 만들기
13. 목록 페이지 - 단어 목록 가져오기
14. 목록 페이지 - 검색 기능 만들기
15. 목록 페이지 - 사전에 없는 단어일 때
16. 2주차 끝 & 숙제 설명
17. 2주차 숙제 답안 코드



모든 토글을 열고 닫는 단축키

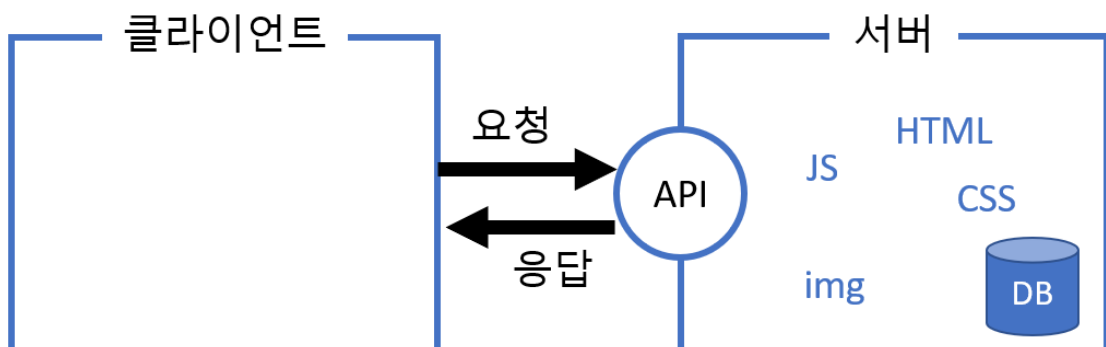
Windows : **Ctrl** + **alt** + **t**

Mac : **⌘** + **⌥** + **t**

01. 2주차 이번주에 배울 것

▼ 동적 웹페이지와 템플릿 언어

▼ 정적 웹페이지 vs. 동적 웹페이지



- 정적 웹페이지(static web page)는 서버에 저장되어있는 HTML+CSS 파일 그대로 보여주는 것!
- 반면 동적 웹페이지(dynamic web page)는 상황에 따라 서버에 저장되어있는 HTML에 데이터 추가/가공을 해서 보여주는 방법입니다 😊
- 정적 웹페이지는 추가적인 통신&계산이 필요 없기 때문에 속도가 빠르고 서버에 부담이 적은 반면, 추가/수정/삭제 등 내용 변경이 필요할 때 HTML 자체를 수정해야 하기 때문에 번거롭다는 단점이 있습니다.
- 동적 웹페이지는 한 페이지에서 상황/시간/사용자요청에 따라 다른 모습을 보여줄 수 있다는 장점이 있지만 상대적으로 보안에 취약하고 모습이 계속 변하기 때문에 (많은 경우 주소도 같이 변하죠!) 검색 엔진 최적화(search engine optimization, SEO)가 어렵습니다.

	정적 웹페이지	동적 웹페이지
특징	서버에 저장되어 있는 그대로 HTML 전송	요청 정보에 따라 HTML을 처리하여 전송
장점	<ul style="list-style-type: none"> • 속도가 빠르다 • 서버 부담이 적다 	<ul style="list-style-type: none"> • 상황에 맞게 변하는 모습 • 관리가 쉽다
단점	<ul style="list-style-type: none"> • 서비스가 한정적이다 • 내용 변경이 어렵다 	<ul style="list-style-type: none"> • 보안에 취약하다 • 검색엔진최적화가 어렵다
예시	회사소개, 음식메뉴, 포트폴리오 등	블로그, 게시판, 날씨 정보 등

▼ 동적 웹페이지의 종류

- Client-side rendering (CSR)

자바스크립트에 데이터를 포함해서 보낸 후, 클라이언트 쪽에서 HTML을 완성하는 방법

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Title</title>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

    <script>
      let people = ["홍길동", "이순신", "임꺽정"]

      $(document).ready(function () {
        for (let i = 0; i < people.length; i++) {
          let html_temp = `<li>${people[i]}</li>`
          $("#name-list").append(html_temp)
        }
      })
    </script>
  </head>
  <body>
    <h2>사람 이름</h2>
    <ul id="name-list"></ul>
  </body>
</html>
```

사람 이름

- 홍길동
- 이순신
- 임꺽정

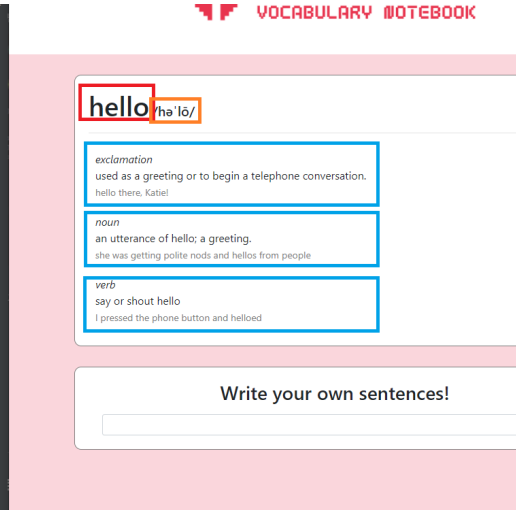
- Server-side rendering (SSR)

서버 쪽에서 템플릿 HTML에 데이터를 끼워넣어 완성된 형태의 HTML을 보내주는 방법

```

<div class="container">
  <div class="d-flex justify-content-between align-items-end">
    <div>
      <h1 id="word" style="{{ result.word }}" />
      {% if result.pronunciation %}
      <h5 id="pronunciation" style="{{ result.pronunciation }}" />
      {% endif %}
    </div>
    <button id="btn-save" class="btn btn-outline-sparta btn-lg" onclick="save_w
    class="fa fa-floppy-o"
    aria-hidden="true"></button>
    <button id="btn-delete" class="btn btn-sparta btn-lg" onclick="delete_word(
    class="fa fa-trash-o"
    aria-hidden="true"></button>
  </div>
  <hr>
  <div id="definitions">
    {% for definition in result.definitions %}
    <div style="{{ definition.type }}">
      <i>{{ definition.type }}</i>
      <br>{{ definition.definition }}<br>
      {% if definition.example %}
      <span class="example">{{ definition.example|safe }}</span>
      {% endif %}
    </div>
    {% endfor %}
  </div>
</div>

```



• 복합적인 방법

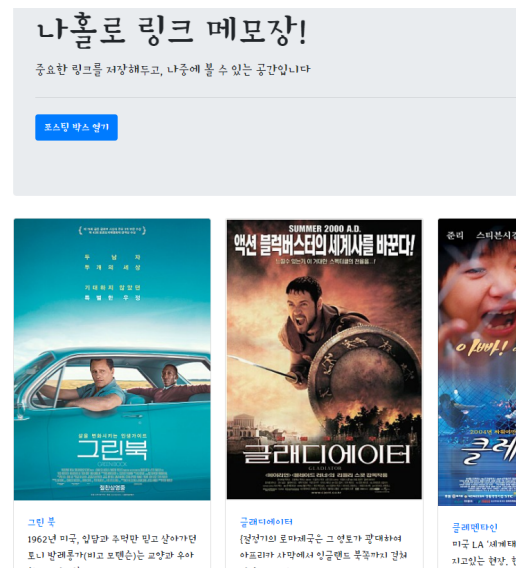
클라이언트 쪽에서 Ajax 요청을 보내서 서버에서 데이터를 받아와 HTML을 완성하는 방법

```

function showArticles() {
  $.ajax({
    type: "GET",
    url: "/post",
    data: {},
    success: function (response) {
      let articles = response["articles"];
      for (let i = 0; i < articles.length; i++) {
        makeCard(articles[i]["image"], articles[i]["url"], arti
      }
    }
  })
}

function makeCard(image, url, title, desc, comment) {
  let tempHtml = `<div class="card">
    
      <a href="${url}" target="_blank" class="card-title">${title
      <p class="card-text">${desc}</p>
      <p class="card-text comment">${comment}</p>
    </div>
  `;
  $("#cards-box").append(tempHtml);
}
</script>
</head>

```



▼ Jinja2 템플릿 언어



Flask 프레임워크에서 사용하는 템플릿 언어
'템플릿'이 되는 HTML 문서에 데이터가 들어갈 곳을 표시해놓는 역할을 합니다!

▼ API 키

- Open API라도 너무 많은 요청을 보내는 등의 악용을 방지하기 위해, API 키를 배부 받아 요청 시에 같이 보내줘야하는 경우가 있습니다. 오늘 쓸 Owlibot 사전 API처럼요!
- 이 API 키를 1) 플라스크 서버에서 API로 요청을 보낼 때, 2) 클라이언트에서 API로 요청을 보낼 때 어떻게 같이 보내줘야하는지 배워보겠습니다.

▼ 이번주에 만들 웹서비스 구경하기: 나만의 단어장

▼ [코드스니펫] - 나만의 단어장 보러가기

<http://spartacodingclub.shop/wp/vocab>

02. 플라스크로 멀티페이지 사이트 만들기 - 기초

▼ 1) 페이지 간 이동하기



메인 페이지에서 링크를 클릭하면 상세 페이지로 가고, 상세 페이지에서 다시 메인으로 갈 수 있게 하려면 어떻게 해야 할까요?

▼ 우선 파이참으로 이번주의 프로젝트를 준비해야겠죠!

- File > New Project...에 가서 project02 폴더 열기
- Project Interpreter에서 가상환경에 필요한 패키지 설치하기(flask , requests , pymongo)
- project02 폴더 안에 templates, static 폴더 만들기
- app.py, index.html, detail.html 파일 만들기

▼ [코드스니펫] - app.py 시작코드

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def main():
    return render_template("index.html")

@app.route('/detail')
def detail():
    return render_template("detail.html")

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

▼ 페이지 연결하기

- 상세 페이지로 가는 하이퍼링크는 이렇게 만듭니다.

```
<a href="/detail">상세 페이지로 가기</a>
```

- 메인 페이지로 돌아가는 버튼은 이렇게 만들 수 있겠죠!

```
// script 태그 안에 정의하기
function to_main() {
    window.location.href = "/"
}
```

```
<!-- 버튼에 함수 연결하기 -->
<button onclick="to_main()">메인으로 돌아가기</button>
```

- 짧은 코드는 onclick에 바로 넣을 수 있습니다.

```
<button onclick='window.location.href = "/">메인으로 돌아가기</button>
```

▼ 2) Jinja2 템플릿 언어 이용하기

- 서버에서 name이라는 이름으로 값을 보내줍니다.

```
@app.route('/')
def main():
    myname = "sparta"
    return render_template("index.html", name=myname)
```

- html 파일에서 이 값이 들어갈 자리를 표시해줍니다.

```
<h3>Hello, {{ name }}!</h3>
```



파이참 Settings(맥은 Preferences) > Languages & Frameworks > Template Languages에서 템플릿 언어를 Jinja2로 설정해주면 자동완성과 하이라이팅 기능을 사용할 수 있어요!

03. 플라스크로 멀티페이지 사이트 만들기 - 응용

▼ 3) Jinja2 템플릿 언어 이용하기 응용편

▼ Jinja를 본격적으로 쓰기에 앞서, 원래 ajax 요청을 보내 html을 완성할 때는 어떻게 했었는지 살펴봅시다. 서울시 Open API에서 정보를 받아와 현재 미세먼지 수치가 50 이상인 구만 페이지에 나타내볼까요?

▼ [코드스니펫] - ajax 요청 준비

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
    $(document).ready(function () {
        get_list()
    })

    function get_list() {
        $.ajax({
            type: "GET",
            url: "http://openapi.seoul.go.kr:8088/6d4d776b466c656533356a4b4b5872/json/RealtimeCityAir/1/99",
            data: {},
            success: function (response) {
                let rows = response["RealtimeCityAir"]["row"];
                console.log(rows)
            }
        })
    }
</script>
```

- 미세먼지 정보가 들어갈 ul 태그 만들기

```
<ul id="gu-list">
</ul>
```

- 각 구에 대해서 구 이름과 미세먼지 수치를 변수에 저장하기

```
for (let i=0;i<rows.length;i++) {
    let gu_name = rows[i]["MSRSTE_NM"]
    let gu_mise = rows[i]["IDEX_MVL"]
    console.log(gu_name, gu_mise)
}
```

- 미세먼지 수치가 50 이상일 때만 태그 추가하기

```
if (gu_mise >= 50) {
    let html_temp = `<li>${gu_name}: ${gu_mise}</li>`
    $("#gu-list").append(html_temp)
}
```

▼ 이번에는 같은 일을 jinja2로 해보겠습니다.

▼ [코드스니펫] - requests 요청 보내기

```
r = requests.get('http://openapi.seoul.go.kr:8088/6d4d776b466c65653356a4b4b5872/json/RealtimeCityAir/1/99')
response = r.json()
rows = response['RealtimeCityAir']['row']
```

- 렌더링할 html에 미세먼지 정보 보내기

```
return render_template("index.html", name=name, rows=rows)
```

- 첫번째 구의 정보를 태그로 만들기

```
<li>{{ rows[0].MSRSTE_NM }}: {{ rows[0].IDEX_MVL }}</li>
```

- 변수에 저장하기

```
{% set gu_name = rows[0].MSRSTE_NM %}
{% set gu_mise = rows[0].IDEX_MVL %}
<li>{{ gu_name }}: {{ gu_mise }}</li>
```

- 모든 구에 대해서 태그 만들기

```
{% for row in rows %}
    {% set gu_name = row.MSRSTE_NM %}
    {% set gu_mise = row.IDEX_MVL %}
    <li>{{ gu_name }}: {{ gu_mise }}</li>
{% endfor %}
```

- 미세먼지 수치가 50 이상일 때만 태그 만들기

```
{% if gu_mise >= 50 %}
    <li>{{ gu_name }}: {{ gu_mise }}</li>
{% endif %}
```

- 그 외에도, head 태그 안의 내용(title 태그 등)을 바꿀 때도 쓸 수 있고, 다른 html 문서를 통째로 가져와 템플릿으로 사용할 수도 있습니다.

▼ 4) URL의 일부를 변수로 받기

- ▼ 브라우저에 HTML을 띄우는 것은 GET 요청이기 때문에, 주소 뒤에 ? 를 붙여 파라미터를 넘겨줄 수 있습니다.

- 브라우저에 해당 API로 요청 보내기

```
http://localhost:5000/detail?word_give=hello
```

- 서버에서 파라미터 값을 받아 HTML로 넘겨주기

```
@app.route('/detail')
def detail():
    word_receive = request.args.get("word_give")
    return render_template("detail.html", word=word_receive)
```

- HTML에서 word 라는 변수에 저장된 값 나타내기

```
받은 단어는 {{ word }}
```

- 플라스크 프레임워크에서는 URL의 일부를 변수로 받는 방법도 사용할 수 있습니다.

```
@app.route('/detail/<keyword>')
def detail(keyword):
    return render_template("detail.html", word=keyword)
```

04. 사전 API 사용하기

▼ 5) 사전 API 구경하기

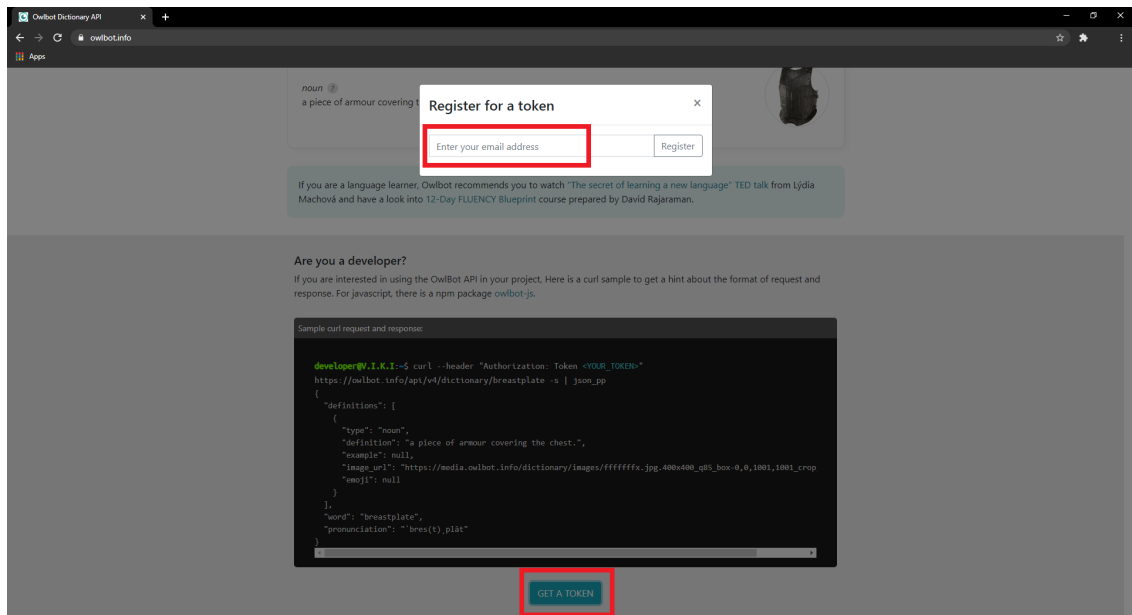
- ▼ 이번주 프로젝트에서는 Owlbot이라는 Open API에서 영단어의 발음, 뜻, 예문을 받아올 거예요. 한 번 살펴볼까요?

▼ [코드스니펫] - Owlbot 웹사이트

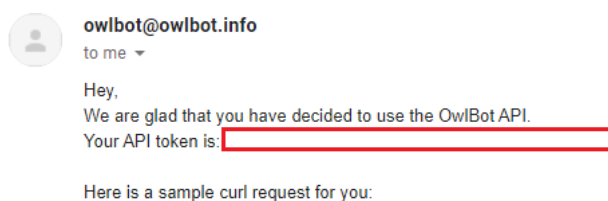
```
https://owlbot.info/
```

▼ 6) API 키 신청하기

- 웹사이트에서 토큰 신청하기



- 이메일로 토큰 받기



▼ 7) 파이썬으로 API에 요청 보내기

▼ [코드스니펫] - Owlbot API 요청 파이썬

```
r = requests.get("https://owlbot.info/api/v4/dictionary/owl", headers={"Authorization": "Token [내토큰]"})
result = r.json()
print(result)
```

▼ 8) Ajax로 요청 보내기

▼ [코드스니펫] - Owlbot API 요청 Ajax

```
$.ajax({
  type: "GET",
  url: "https://owlbot.info/api/v4/dictionary/owl",
  beforeSend: function (xhr) {
```



```

    xhr.setRequestHeader("Authorization", "Token [내토큰]");
  },
  data: {},
  error: function (xhr, status, error) {
    alert("에러 발생!");
  },
  success: function (response) {
    console.log(response)
  }
}
})

```

05. 프로젝트 2: 나만의 단어장

▼ 9) 문제 분석 - 완성작부터 보기: 나만의 단어장

▼ [코드스니펫] - 나만의 단어장 보러가기

<http://spartacodingclub.shop/wp/vocab>

▼ 10) API 설계하기



필요한 기능들을 생각해봅시다. 각 페이지에서는 어떤 일이 일어나야 하나요?

▼ 메인 페이지

1. 단어 검색

단어가 단어장에 이미 있는 단어인지 검색 → 있으면 하이라이트, 없으면 상세 페이지로 이동

2. 단어장에 있는 단어를 클릭했을 때 상세 페이지로 이동

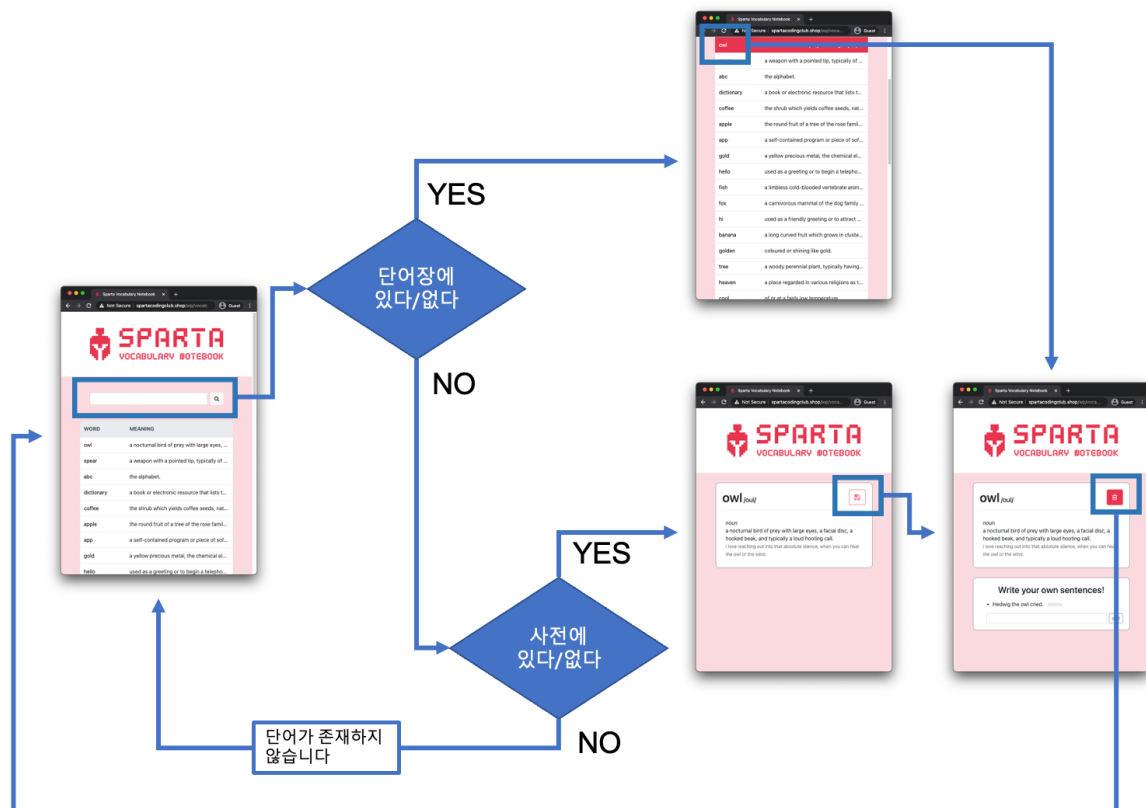
▼ 상세 페이지

1. 단어 저장 또는 삭제

- 단어가 이미 존재하면 삭제 버튼, 아니면 저장 버튼 노출
- 저장 버튼을 누르면 DB에 저장하고 삭제 버튼으로 바뀜
- 삭제 버튼을 누르면 DB에서 삭제하고 메인 페이지로 이동

2. 예문 저장과 삭제

- 저장된 단어의 경우 예문 칸이 보여지게 하기
- 예문을 저장하면 목록 맨 아래에 추가
- 예문에 단어가 포함되지 않으면 얼럿 띄우기
- 예문을 선택해서 삭제할 수 있음



06. 프로젝트 세팅

▼ 11) 프로젝트 준비 - app.py 준비하기

▼ [코드스니펫] - 나만의 단어장 app.py 시작코드

```

from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient
import requests

app = Flask(__name__)

client = MongoClient('내AWS아이피', 27017, username="아이디", password="비밀번호")
db = client.dbsparta_plus_week2

@app.route('/')
def main():
    # DB에서 저장된 단어 찾아서 HTML에 나타내기
    return render_template("index.html")

@app.route('/detail/<keyword>')
def detail(keyword):
    # API에서 단어 뜻 찾아서 결과 보내기
    return render_template("detail.html", word=keyword)

@app.route('/api/save_word', methods=['POST'])
def save_word():
    # 단어 저장하기
    return jsonify({'result': 'success', 'msg': '단어 저장'})

@app.route('/api/delete_word', methods=['POST'])
def delete_word():

```

```
# 단어 삭제하기
return jsonify({'result': 'success', 'msg': '단어 삭제'})

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



로컬에서 개발할 때도 AWS 서버에 있는 MongoDB를 바로 연결해서 코딩하면 나중에 배포할 때 DB를 복사해 옮길 필요가 없어 편합니다 😊

▼ 12) 프로젝트 준비 - index.html, detail.html 준비하기

▼ [코드스니펫] - 나만의 단어장 index.html, detail.html 시작코드

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <title>Sparta Vocabulary Notebook</title>

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
          integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
          crossorigin="anonymous">

    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
          integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
          crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
          integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
          crossorigin="anonymous"></script>

  </head>
  <body>
  </body>
</html>
```

▼ 13) 프로젝트 준비 - 배너 이미지 준비하기

▼ [코드스니펫] - 나만의 단어장 배너 이미지

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/be2028d1-26d3-4bed-8bc6-70b414612a49/logo_red.png

07. 상세 페이지 전체 모습 만들기

▼ 14) 배경과 배너 넣기

▼ [코드스니펫] - 배경 & 배너 HTML

```
<div class="wrap">
  <div class="banner" onclick="window.location.href = '/'">
  </div>
</div>
```

▼ [코드스니펫] - 배경 & 배너 CSS

```
.wrap {
  background-color: RGBA(232, 52, 78, 0.2);
  min-height: 100vh;
  padding-bottom: 50px;
}

.banner {
  width: 100%;
  height: 200px;

  background-color: white;
  background-image: url('{{ url_for("static", filename="logo_red.png") }}');

  background-position: center;
  background-size: contain;
  background-repeat: no-repeat;

  cursor: pointer;
}
```

▼ 15) 단어 뜻 박스 만들기

▼ [코드스니펫] - 단어 뜻 HTML

```
<div class="container">
  <div class="d-flex justify-content-between align-items-end">
    <div>
      <h1 id="word" style="display: inline;">owl</h1>
      <h5 id="pronunciation" style="display: inline;">/oul</h5>
    </div>
    <button id="btn-save" class="btn btn-outline-sparta btn-lg">save</button>
    <button id="btn-delete" class="btn btn-sparta btn-lg">delete</button>
  </div>
  <hr>
  <div id="definitions">
    <div style="padding:10px">
      <i>noun</i>
      <br>a nocturnal bird of prey with large eyes, a facial disc, a hooked beak, and typically a loud
      hooting call.<br>
      <span class="example">I love reaching out into that absolute silence, when you can hear the owl or the wind.</span>
    </div>
  </div>
</div>
```

▼ [코드스니펫] - 단어 뜻 CSS

```
.container {
  width: 80%;
  max-width: 800px;
  margin: 30px auto;
  padding: 20px;
  background-color: white;

  border: solid 1px gray;
  border-radius: 10px;
}

span.example {
  color: gray;
  font-size: 14px;
}

.btn-sparta {
  color: #fff;
  background-color: #e8344e;
  border-color: #e8344e;
}

.btn-outline-sparta {
  color: #e8344e;
  background-color: transparent;
  background-image: none;
  border-color: #e8344e;
}
```

▼ 16) 아이콘 삽입하기

▼ 아이콘은 [Font Awesome](#)이라는 곳에서 가져와서 쓰겠습니다.

▼ [코드스니펫] - Font Awesome 웹사이트

```
https://fontawesome.com/v4.7.0/
```

▼ Font Awesome의 아이콘을 쓰기 위해서는 head에 링크를 삽입해주어야합니다.

▼ [코드스니펫] - Font Awesome 링크

```
<link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">
```

• HTML에서는 클래스의 형식으로 들어갑니다.

```
<i class="fa fa-cog"></i>
```

08. 상세 페이지 - Ajax로 단어 뜻 가져오기

▼ 17) html에서 단어 뜻 보여주기 - ajax 이용

▼ 우선 ajax를 이용해서 단어 뜻을 가져와 넣어봅시다.

▼ [코드스니펫] - Owlbot API 요청 Ajax

```
let word = '{{ word }}'
$(document).ready(function () {
  get_definitions()
})

function get_definitions() {
  $.ajax({
    type: "GET",
    url: `https://owlbot.info/api/v4/dictionary/${word}`,
    beforeSend: function (xhr) {
      xhr.setRequestHeader("Authorization", "Token [내토큰]");
    },
    data: {},
    error: function (xhr, status, error) {
      alert("에러 발생!");
    },
    success: function (response) {
      console.log(response)
    }
  })
}
```

• API에서 받은 값들을 해당하는 태그에 넣어줍니다.

```
$("#word").text(response["word"])
$("#pronunciation").text(`${response["pronunciation"]}/`)
$("#definitions").empty()
let definitions = response["definitions"]
for (let i=0;i<definitions.length;i++) {
  let definition = definitions[i]
  let html_temp = `<div style="padding:10px">
    <i>${definition["type"]}</i>
    <br>${definition["definition"]}<br>
    <span class="example">${definition["example"]}</span>
  </div>`
  $("#definitions").append(html_temp)
}
```

• 발음이 null 일 경우 빈 텍스트를 넣어줍니다.

```
if (response["pronunciation"]==null) {
  $("#pronunciation").text("")
} else {
```

```

    $("#pronunciation").text(`${response["pronunciation"]}`)
}

```

- 예문이 `null` 일 경우 예외처리를 해줍니다.

```

let html_temp = ``
if (definition["example"]!=null) {
    html_temp = `<div style="padding:10px">
        <i>${definition["type"]}</i>
        <br>${definition["definition"]}<br>
        <span class="example">${definition["example"]}</span>
    </div>`
} else {
    html_temp = `<div style="padding:10px">
        <i>${definition["type"]}</i>
        <br>${definition["definition"]}<br>
    </div>`
}

$("#definitions").append(html_temp)

```

09. 상세 페이지 - jinja2로 뜻 가져오기

▼ 18) html에서 단어 뜻 보여주기 - jinja2 이용

- 우선 app.py에서 API에 요청을 보내 받은 응답을 보내줍니다.

▼ [코드스니펫] - Owlbot API 요청 Flask

```

r = requests.get(f"https://owlbot.info/api/v4/dictionary/{keyword}", headers={"Authorization": "Token [네토큰]"})
result = r.json()
print(result)

```

- HTML에서 결과가 들어가야할 부분을 표시해줍니다.

```

<div class="container">
    <div class="d-flex justify-content-between align-items-end">
        <div>
            <h1 id="word" style="display: inline;">{{ word }}</h1>
            <h5 id="pronunciation" style="display: inline;">{{ result.pronunciation }}</h5>
        </div>
        <button id="btn-save" class="btn btn-outline-sparta btn-lg">save</button>
        <button id="btn-delete" class="btn btn-sparta btn-lg" style="display:none;">delete</button>
    </div>
    <hr>
    <div id="definitions">
        {% for definition in result.definitions %}
            <div style="padding:10px">
                <i>{{ definition.type }}</i>
                <br>{{ definition.definition }}<br>
                <span class="example">{{ definition.example }}</span>
            </div>
        {% endfor %}
    </div>
</div>

```

- 발음이 있는 경우에만 보여주도록 예외처리를 해줍니다.

```

{% if result.pronunciation %}
    <h5 id="pronunciation" style="display: inline;">{{ result.pronunciation }}</h5>
{% endif %}

```

- 예문이 있는 경우에만 보여주도록 예외처리를 해줍니다.

```

{% if definition.example %}
    <span class="example">{{ definition.example }}</span>
{% endif %}

```

- 예문에 HTML 태그 쓰는 것을 허용해줍니다.

```
<span class="example">{{ definition.example|safe }}</span>
```

- 정의와 예문에서 깨진 글자를 없앱니다.

```
<br>{{ definition.definition.encode('ascii', 'ignore').decode('utf-8') }}<br>
{% if definition.example %}
  <span class="example">{{ definition.example.encode('ascii', 'ignore').decode('utf-8') }}</span>
{% endif %}
```

10. 상세 페이지 - 새 단어/기존 단어 구분하기

▼ 19) 단어에 따라 저장/삭제 버튼 숨기기

- 그럴 일은 없지만, `status_give` 라는 파라미터를 주지 않을 경우를 대비해, 기본값을 `"new"` 로 주도록 하겠습니다.

```
status_receive = request.args.get("status_give", "new")
```

11. 상세 페이지 - 저장 & 삭제 기능 만들기

▼ 20) 단어 저장 기능 만들기

- 목록 페이지에서는 단어 당 뜻을 하나만 보여줄 것이기 때문에 단어와 첫 번째 정의만 POST 요청으로 보내고, 서버에서 단어와 뜻을 받아 `words` 컬렉션에 저장합니다.

```
@app.route('/api/save_word', methods=['POST'])
def save_word():
    # 단어 저장하기
    word_receive = request.form['word_give']
    definition_receive = request.form['definition_give']
    doc = {"word": word_receive, "definition": definition_receive}
    db.words.insert_one(doc)
    return jsonify({'result': 'success', 'msg': f'word "{word_receive}" saved'})
```

- 클라이언트에서는 단어와 첫번째 정의만 POST 요청으로 보내줍니다. 단어 저장에 성공하면 얼럿을 띄운 후, `status=old` 로 바뀐 페이지를 띄워줍니다. 저장 버튼에 `onclick=save_word()` 로 연결해줍니다.

▼ [코드스니펫] - save_word() 시작코드

```
function save_word() {
    $.ajax({
        type: "POST",
        url: `/api/save_word`,
        data: {},
        success: function (response) {
            alert(response["msg"])
        }
    });
}
```

```
function save_word() {
    $.ajax({
        type: "POST",
        url: `/api/save_word`,
        data: {
            word_give: "{{ word }}",
            definition_give: "{{ result.definitions[0].definition }}"
        },
        success: function (response) {
            alert(response["msg"])
            window.location.href = "/detail/{{ word }}?status_give=old"
        }
    })
}
```

```
});
}
```

▼ 21) 단어 삭제 기능 만들기

- 단어를 삭제할 때는 단어만 있으면 되므로 POST 요청으로 단어를 보내주고, 서버에서는 해당 단어를 찾아 삭제해줍니다.

```
@app.route('/api/delete_word', methods=['POST'])
def delete_word():
    # 단어 삭제하기
    word_receive = request.form['word_give']
    db.words.delete_one({"word":word_receive})
    return jsonify({'result': 'success', 'msg': f'word "{word_receive}" deleted'})
```

- 클라이언트에서는 단어를 보내주고, 단어 삭제에 성공하면 더이상 보여줄 정보가 없으므로 얼럿을 띄운 후 메인 페이지로 이동합니다.

▼ [코드스니펫] - delete_word() 시작코드

```
function delete_word() {
    $.ajax({
        type: "POST",
        url: '/api/delete_word',
        data: {},
        success: function (response) {
            alert(response["msg"])
        }
    });
}
```

```
function delete_word() {
    $.ajax({
        type: "POST",
        url: '/api/delete_word',
        data: {
            word_give: '{{ word }}',
        },
        success: function (response) {
            alert(response["msg"])
            window.location.href = "/"
        }
    });
}
```

12. 목록 페이지 - 전체 모습 만들기

▼ 22) CSS 파일 분리하기

- 단어 목록 페이지와 단어 상세 페이지는 배경과 배너 등 디자인 요소가 겹치므로 CSS 파일을 분리하여 링크로 넣어주면 같은 CSS를 적용해줄 수 있습니다.
- static 폴더에 mystyle.css 파일을 만들고 공통 요소에 대한 CSS를 잘라내어 붙여넣습니다.
- 배너 이미지는 같은 폴더 안에 있으므로 아래처럼 바꿔줍니다.

```
background-image: url('logo_red.png');
```

- 두 html에 아래와 같이 링크를 걸어줍니다.


▼ [코드스니펫] - mystyle.css 링크

```
<link href='{{ url_for("static", filename="mystyle.css") }}' rel="stylesheet">
```

```
<link href='{{ url_for("static", filename="mystyle.css") }}' rel="stylesheet">
```


- index.html에 배경과 배너를 넣어 CSS가 잘 적용됨을 확인합니다.

```
<div class="wrap">
  <div class="banner" onclick="window.location.href = '/'">
  </div>
</div>
```

 만약 잘 적용되지 않는다면? 캐시 문제일 수 있으니 강제 새로고침(**Ctrl/Cmd + Shift + R**)을 한 번 해보세요!

▼ 23) 검색창과 테이블 만들기

- 우선 검색창을 만들어줍니다.

▼ [코드스니펫] - 검색창 HTML

```
<div class="search-box d-flex justify-content-center">
  <input id="input-word" class="form-control" style="margin-right: 0.5rem">
  <button class="btn btn-light" onclick="find_word()"><i class="fa fa-search"></i></button>
</div>
```

▼ [코드스니펫] - 검색창 CSS

```
.search-box {
  width: 70%;
  margin: 50px auto;
  max-width: 700px;
}
```

- 테이블도 만들어줍니다.

▼ [코드스니펫] - 테이블 HTML

```
<table class="table">
  <thead class="thead-light">
    <tr>
      <th scope="col" style="width:30%">WORD</th>
      <th scope="col">MEANING</th>
    </tr>
  </thead>
  <tbody id="tbody-box">
    <tr id="word-word">
      <td><a href="#">word</a></td>
      <td>a single distinct meaningful element of speech or writing, used with others (or sometimes alone) to form a sentence and typically shown with a space on either side when written or printed.
      </td>
    </tr>
    <tr id="word-dictionary">
      <td><a href="#">dictionary</a></td>
      <td>a book or electronic resource that lists the words of a language (typically in alphabetical order) and gives their meaning, or gives the equivalent words in a different language, often also providing information about pronunciation, origin, and
      </td>
    </tr>
    <tr id="word-name">
      <td><a href="#">name</a></td>
      <td>a word or set of words by which a person or thing is known, addressed, or referred to.
      </td>
    </tr>
  </tbody>
</table>
```

▼ [코드스니펫] - 테이블 CSS

```

.table {
  width: 80%;
  max-width: 800px;
  margin: auto;
  table-layout: fixed;
}

.table th {
  border-top-style: none;
}

td {
  background-color: white;
  text-overflow: ellipsis;
  overflow: hidden;
  white-space: nowrap;
}

td > a, a:visited, a:hover, a:active {
  color: black;
}

thead:first-child tr:first-child th:first-child {
  border-radius: 10px 0 0 0;
}

thead:first-child tr:first-child th:last-child {
  border-radius: 0 10px 0 0;
}

tbody:last-child tr:last-child td:first-child {
  border-radius: 0 0 0 10px;
}

tbody:last-child tr:last-child td:last-child {
  border-radius: 0 0 10px 0;
}

```

13. 목록 페이지 - 단어 목록 가져오기

▼ 24) jinja2로 단어 목록 테이블 채우기

- 이제 실제로 DB에서 단어 목록을 가져와 테이블을 채워봅시다.
- app.py에서는 words 컬렉션의 단어들을 가져와 넘겨줍니다.

```

@app.route('/')
def main():
    # DB에서 저장된 단어 찾아서 HTML에 나타내기
    words = list(db.words.find({}, {"_id": False}))
    return render_template("index.html", words=words)

```

- index.html에서는 각 단어마다 테이블의 한 줄이 되도록 넣어줍니다.

```

<tbody id="tbody-box">
  {% for word in words %}
  <tr id="word-{{ word.word }}">
    <td><a href="/detail/{{ word.word }}"?status_give=old">{{ word.word }}</a></td>
    <td>{{ word.definition|safe }}
    </td>
  </tr>
  {% endfor %}
</tbody>

```

14. 목록 페이지 - 검색 기능 만들기

▼ 25) 단어 검색 기능 만들기

- 단어를 검색했을 때 이미 저장된 단어인지 알기 위해서 있는 단어 리스트를 만듭니다.

```
let words = [{ words|tojson }];
let word_list = [];
for (let i = 0; i < words.length; i++) {
  word_list.push(words[i]["word"])
}
```

- 단어를 검색했을 때 단어 리스트에 있는 경우에는 해당 행을 하이라이트하고, 없는 단어일 때는 단어 상세페이지로 넘어가는 기능을 만듭니다.

```
function find_word() {
  let word = $("#input-word").val().toLowerCase();
  if (word == "") {
    // 빈 문자열이면 얼럿
    alert("please write something first :)")
    return
  }
  if (word_list.includes(word)) {
    // 리스트에 있으면 하이라이트
    $('#word-${word}`).addClass('highlight').siblings().removeClass('highlight');
    $('#word-${word}')[0].scrollIntoView();
  } else {
    // 리스트에 없으면 상세 페이지로
    window.location.href = `/detail/${word}?status_give=new`
  }
}
```

- 하이라이트된 행은 다음과 같은 CSS로 나타내줍니다.

```
tr.highlight > td {
  background-color: #e8344e;
  color: white;
}

tr.highlight a {
  color: white;
}
```

15. 목록 페이지 - 사전에 없는 단어일 때

▼ 26) 단어가 존재하지 않을 때 기능 만들기

- 주소에 단어가 아닌 것을 넣었을 때, 사전 API에서 단어를 찾을 수 없기 때문에 예러가 납니다. 이 때 예러를 보여주지 않고 단어 목록 페이지로 리다이렉팅시켜봅시다.
- 값을 잘 받아왔을 때 상태 코드가 200이므로 200이 아닐 때 `main`으로 리다이렉팅 시킵니다.

```
if r.status_code != 200:
  return redirect(url_for("main"))
```

- 단어 찾기 실패 얼럿을 띄우려면 `redirect()`에 메시지를 같이 전달합니다.

```
url_for("main", msg="Word not found in dictionary; Try another word")
```

- `main()`에서 메시지를 받아 템플릿에 같이 보내줍니다.

```
@app.route('/')
def main():
  # DB에서 저장된 단어 찾아서 HTML에 나타내기
  msg = request.args.get("msg")
  return render_template("index.html", words=words, msg=msg)
```

- index.html에서 `msg`가 있을 때 해당 메시지로 얼럿을 띄웁니다.

```
{% if msg %}
    alert("{{ msg }}")
{% endif %}
```

▼ 27) og태그, favicon 넣기

- 서비스의 완성도를 위해 Open Graph 태그와 favicon을 넣어주는 게 좋습니다. ogimg는 배너 이미지를 사용하고 favicon은 아래 파일을 다운 받아 static 폴더에 넣어줍니다.

▼ [코드스니펫] - favicon

```
https://s3-us-west-2.amazonaws.com/secure.notion-static.com/8ba29765-3ffb-42d4-bff1-aa1e68ae60e9/favicon.ico
```

- index.html과 detail.html 위에 링크를 첨부합니다.

▼ [코드스니펫] - og태그와 favicon 링크

```
<meta property="og:title" content="Sparta Vocabulary Notebook"/>
<meta property="og:description" content="mini project for Web Plus"/>
<meta property="og:image" content="{{ url_for('static', filename='logo_red.png') }}" />
<link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
<link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
```

▼ 28) 전체 완성 코드

▼ app.py

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient
import requests

app = Flask(__name__)

client = MongoClient('mongodb://아이디:비밀번호@내 아이피', 27017)
db = client.dbsparta_plus_week2

@app.route('/')
def main():
    msg = request.args.get("msg")

    # DB에서 저장된 단어 찾아서 HTML에 나타내기
    words = list(db.words.find({}, {"_id": False}))
    return render_template("index.html", words=words, msg=msg)

@app.route('/detail/<keyword>')
def detail(keyword):
    # API에서 단어 뜻 찾아서 결과 보내기
    status_receive = request.args.get("status_give", "old")
    r = requests.get(f"https://owlbot.info/api/v4/dictionary/{keyword}",
                    headers={"Authorization": "Token [내 토큰]"})
    if r.status_code != 200:
        return redirect(url_for("main", msg="Word not found in dictionary; Try another word"))
    result = r.json()
    print(result)
    return render_template("detail.html", word=keyword, result=result, status=status_receive)

@app.route('/api/save_word', methods=['POST'])
def save_word():
    # 단어 저장하기
    word_receive = request.form['word_give']
    definition_receive = request.form['definition_give']
    doc = {"word": word_receive, "definition": definition_receive}
    db.words.insert_one(doc)
    return jsonify({'result': 'success', 'msg': f'word "{word_receive}" saved'})

@app.route('/api/delete_word', methods=['POST'])
def delete_word():
    # 단어 삭제하기
    word_receive = request.form['word_give']
```

```

db.words.delete_one({"word": word_receive})
return jsonify({'result': 'success', 'msg': f'word "{word_receive}" deleted'})

if __name__ == '__main__':
    app.run('0.0.0.0', port=5000, debug=True)

```

▼ index.html

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <title>Sparta Vocabulary Notebook</title>
    <meta property="og:title" content="Sparta Vocabulary Notebook"/>
    <meta property="og:description" content="mini project for Web Plus"/>
    <meta property="og:image" content="{{ url_for('static', filename='logo_red.png') }}" />
    <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
    <link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
      integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
      crossorigin="anonymous">
    <link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">

    <link href="{{ url_for("static", filename="mystyle.css") }}" rel="stylesheet">
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
      integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
      crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
      integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"></script>

    <style>
      .search-box {
        width: 70%;
        margin: 50px auto;
        max-width: 700px;
      }

      .table {
        width: 80%;
        max-width: 800px;
        margin: auto;
        table-layout: fixed;
      }

      .table th {
        border-top-style: none;
      }

      td {
        background-color: white;
        text-overflow: ellipsis;
        overflow: hidden;
        white-space: nowrap;
      }

      td > a, a:visited, a:hover, a:active {
        color: black;
      }

      tr.highlight > td {
        background-color: #e8344e;
        color: white;
      }

      tr.highlight a {
        color: white;
      }

      thead:first-child tr:first-child th:first-child {
        border-radius: 10px 0 0 0;
      }

      thead:first-child tr:first-child th:last-child {
        border-radius: 0 10px 0 0;
      }

      tbody:last-child tr:last-child td:first-child {
        border-radius: 0 0 0 10px;
      }
    </style>

```

```

    }

    tbody:last-child tr:last-child td:last-child {
        border-radius: 0 0 10px 0;
    }
</style>
<script>
    {% if msg %}
        alert('{{ msg }}')
    {% endif %}
    let words = {{ words|tojson }};
    let word_list = [];
    for (let i = 0; i < words.length; i++) {
        word_list.push(words[i]["word"])
    }

    function find_word() {
        let word = $("#input-word").val().toLowerCase();
        if (word == "") {
            alert("please write something first :)")
            return
        }
        if (word_list.includes(word)) {
            $('#word-{{word}}').addClass('highlight').siblings().removeClass('highlight');
            $('#word-{{word}}').get(0).scrollIntoView();
        } else {
            window.location.href = `/detail/${word}?status_give=new`
        }
    }
</script>
</head>
<body>
    <div class="wrap">
        <div class="banner" onclick="window.location.href = '/'">
        </div>
        <div class="search-box d-flex justify-content-center">
            <input id="input-word" class="form-control" style="margin-right: 0.5rem">
            <button class="btn btn-light" onclick="find_word()"><i class="fa fa-search"></i></button>
        </div>
        <table class="table">
            <thead class="thead-light">
                <tr>
                    <th scope="col" style="width:30%">WORD</th>
                    <th scope="col">MEANING</th>
                </tr>
            </thead>
            <tbody id="tbody-box">
                {% for word in words %}
                    <tr id="word-{{ word.word }}">
                        <td><a href="/detail/{{ word.word }}?status_give=old">{{ word.word }}</a></td>
                        <td>{{ word.definition|safe }}</td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</body>
</html>

```

▼ detail.html

```

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
        <title>Sparta Vocabulary Notebook</title>
        <meta property="og:title" content="Sparta Vocabulary Notebook"/>
        <meta property="og:description" content="mini project for Web Plus"/>
        <meta property="og:image" content="{{ url_for('static', filename='logo_red.png') }}" />
        <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
        <link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
            integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
            crossorigin="anonymous">
        <link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">
        <link href="{{ url_for('static', filename='mystyle.css') }}" rel="stylesheet">
        <!-- Optional JavaScript -->
        <!-- jQuery first, then Popper.js, then Bootstrap JS -->
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

```

```

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
  integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j77fakFPskvXusvfa0b4Q"
  crossorigin="anonymous"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
  integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVMcYl"
  crossorigin="anonymous"></script>
<style>

  span.example {
    color: gray;
    font-size: 14px;
  }

  .btn-sparta {
    color: #fff;
    background-color: #e8344e;
    border-color: #e8344e;
  }

  .btn-outline-sparta {
    color: #e8344e;
    background-color: transparent;
    background-image: none;
    border-color: #e8344e;
  }
</style>
<script>
  let word = "{{ word }}"
  $(document).ready(function () {
    })

    function save_word() {
      $.ajax({
        type: "POST",
        url: '/api/save_word',
        data: {
          word_give: "{{ word }}",
          definition_give: "{{ result.definitions[0].definition }}"
        },
        success: function (response) {
          alert(response["msg"])
          window.location.href = "/detail/{{ word }}?status=old"
        }
      });
    }

    function delete_word() {
      $.ajax({
        type: "POST",
        url: '/api/delete_word',
        data: {
          word_give: '{{ word }}',
        },
        success: function (response) {
          alert(response["msg"])
          window.location.href = "/"
        }
      });
    }
  })
</script>
</head>
<body>
<div class="wrap">
  <div class="banner" onclick="window.location.href = '/'">
  </div>
  <div class="container">
    <div class="d-flex justify-content-between align-items-end">
      <div>
        <h1 id="word" style="display: inline;">{{ result.word }}</h1>
        {% if result.pronunciation %}
          <h5 id="pronunciation" style="display: inline;">{{ result.pronunciation }}</h5>
        {% endif %}
      </div>
      {% if status=="new" %}
        <button id="btn-save" class="btn btn-outline-sparta btn-lg" onclick="save_word()"><i
          class="fa fa-floppy-o"
          aria-hidden="true"></i></button>
      {% else %}
        <button id="btn-delete" class="btn btn-sparta btn-lg" onclick="delete_word()"><i
          class="fa fa-trash-o"
          aria-hidden="true"></i></button>
      {% endif %}
    </div>
    <hr>
    <div id="definitions">
      {% set definitions = result.definitions %}
      {% for definition in definitions %}

```

```

        <div style="padding:10px">
            <i>{{ definition.type }}</i>
            <br>{{ definition.definition.encode('ascii', 'ignore').decode('utf-8') }}<br>
            {% if definition.example %}
                <span class="example">{{ definition.example.encode('ascii', 'ignore').decode('utf-8') }}</span>
            {% endif %}
        </div>
    {% endfor %}
</div>
</div>
</div>
</body>
</html>

```

▼ mystyle.css

```

.wrap {
    background-color: RGBA(232, 52, 78, 0.2);
    min-height: 100vh;
    padding-bottom: 50px;
}

.banner {
    width: 100%;
    height: 200px;

    background-color: white;
    background-image: url('logo_red.png');
    background-position: center;
    background-size: contain;
    background-repeat: no-repeat;


    cursor: pointer;
}

.container {
    width: 80%;
    max-width: 800px;
    margin: 30px auto;
    padding: 20px;
    background-color: white;

    border: solid 1px gray;
    border-radius: 10px;
}

```

16. 2주차 끝 & 숙제 설명

 각 단어에 내가 만든 예문을 저장/삭제하는 기능을 만들어봅시다!

- 예문을 저장/삭제하고 보여주는 기능을 만든 후 AWS 서버에 올리는 것까지가 숙제입니다! 1주차에 올려놓은 일기장은 끄고 올려야겠죠?
- ▼ 어디서 시작해야할지 모르겠다면? 아래 코드들을 복사해놓고 시작해보세요 😊

▼ [코드스니펫] - 숙제 시작 코드 HTML

```

<div id="examples" class="container">
    <h3 style="text-align: center;margin-bottom:1rem">Write your own sentences!</h3>
    <ul id="example-list">
        <li id="ex-0">This sentence contains the word 'word'.&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<a
            href="javascript:delete_ex(0)">delete</a></li>
        <li id="ex-1">I don't like using the MS Word program.&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<a
            href="javascript:delete_ex(1)">delete</a></li>
    </ul>
    <div class="d-flex justify-content-between" style="margin-left:20px;">
        <input id="new-example" class="form-control form-control-sm" style="margin-right: 0.5rem">
        <button class="btn btn-outline-secondary btn-sm" onclick="add_ex()">add</button>
    </div>
</div>

```

▼ [코드스니펫] - 숙제 시작 코드 자바스크립트


```

function get_examples() {
  $("#example-list").empty()
  $.ajax({
    type: "GET",
    url: `/api/get_exs?word_give=${word}`,
    data: {},
    success: function (response) {
      console.log(response)
    }
  });
}

function add_ex() {
  let new_ex = $('#new-example').val();
  console.log(new_ex)
  $.ajax({
    type: "POST",
    url: `/api/save_ex`,
    data: {
    },
    success: function (response) {
      console.log(response)
      get_examples()
    }
  });
}

function delete_ex(i) {
  console.log("deleting", i)
  $.ajax({
    type: "POST",
    url: `/api/delete_ex`,
    data: {
      word_give: word,
      number_give: i
    },
    success: function (response) {
      get_examples()
    }
  });
}

```

▼ [코드스니펫] - 속제 시작 코드 파이썬

```

@app.route('/api/get_exs', methods=['GET'])
def get_exs():
    # 예문 가져오기
    return jsonify({'result': 'success'})

@app.route('/api/save_ex', methods=['POST'])
def save_ex():
    # 예문 저장하기
    return jsonify({'result': 'success'})

@app.route('/api/delete_ex', methods=['POST'])
def delete_ex():
    # 예문 삭제하기
    return jsonify({'result': 'success'})

```

▼ 힌트

- 우선 예문이 들어갈 칸을 만들어주세요. 새 단어일 때는 보이지 않고 기존 단어일 때만 나타나야합니다 😊
- 변화가 있을 때 페이지 전체를 로딩하지 않고 예문 칸만 새로 채워넣는 방법이 효율적이기 때문에 jinja보다는 ajax를 쓰는 것이 좋습니다.
- input 칸에 예문을 넣고 저장 버튼을 누르면 1) 예문이 해당 단어를 포함하는지 확인하고 2) POST 요청을 보내 DB에 저장해주세요. examples라는 컬렉션을 따로 만들면 좋겠죠?
- DB에 저장된 예문들 중 해당 단어에 관한 것만 찾아와서 보여주는 함수를 만듭니다. 페이지가 로딩되었을 때, 새 예문을 저장했을 때, 기존 예문을 삭제했을 때 함수를 실행해서 다시 예문을 보여줍니다.
- 예문을 삭제할 때 여러 예문 중 어떤 것인지 구분을 해야하기 때문에 각 줄에 id를 부여하는 것이 좋습니다.

- 이제 예문들이 생겼으니, 단어를 삭제할 때 이 단어에 해당하는 예문들도 같이 지워주어야합니다. 컬렉션에서 조건을 만족하는 여러 문서를 지울 때는 `delete_many()` 를 사용할 수 있어요.

17. 2주차 숙제 답안 코드

▼ [코드스니펫] - 2주차 숙제 답안 코드

전체 코드

▼ app.py

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
from pymongo import MongoClient
import requests

app = Flask(__name__)

client = MongoClient('mongodb://아이디:비밀번호@내 아이피', 27017)
db = client.dbsparta_plus_week2

@app.route('/')
def main():
    msg = request.args.get("msg")

    # DB에서 저장된 단어 찾아서 HTML에 나타내기
    words = list(db.words.find({}, {"_id": False}))
    return render_template("index.html", words=words, msg=msg)

@app.route('/detail/<keyword>')
def detail(keyword):
    # API에서 단어 뜻 찾아서 결과 보내기
    status_receive = request.args.get("status_give", "old")
    r = requests.get(f"https://owlbot.info/api/v4/dictionary/{keyword}",
                    headers={"Authorization": "Token [내 토큰]"})
    if r.status_code != 200:
        return redirect(url_for("main", msg="Word not found in dictionary; Try another word"))
    result = r.json()
    print(result)
    return render_template("detail.html", word=keyword, result=result, status=status_receive)

@app.route('/api/save_word', methods=['POST'])
def save_word():
    # 단어 저장하기
    word_receive = request.form['word_give']
    definition_receive = request.form['definition_give']
    doc = {"word": word_receive, "definition": definition_receive}
    db.words.insert_one(doc)
    return jsonify({'result': 'success', 'msg': f'word "{word_receive}" saved'})

@app.route('/api/delete_word', methods=['POST'])
def delete_word():
    # 단어 삭제하기
    word_receive = request.form['word_give']
    db.words.delete_one({"word": word_receive})
    db.examples.delete_many({"word": word_receive})
    return jsonify({'result': 'success', 'msg': f'word "{word_receive}" deleted'})

@app.route('/api/get_examples', methods=['GET'])
def get_exs():
    word_receive = request.args.get("word_give")
    result = list(db.examples.find({"word": word_receive}, {"_id": 0}))
    print(word_receive, len(result))

    return jsonify({'result': 'success', 'examples': result})

@app.route('/api/save_ex', methods=['POST'])
def save_ex():
    word_receive = request.form['word_give']
    example_receive = request.form['example_give']
    doc = {"word": word_receive, "example": example_receive}
    db.examples.insert_one(doc)
    return jsonify({'result': 'success', 'msg': f'example "{example_receive}" saved'})

@app.route('/api/delete_ex', methods=['POST'])
```

```
def delete_ex():
    word_receive = request.form['word_give']
    number_receive = int(request.form["number_give"])
    example = list(db.examples.find({"word": word_receive}))[number_receive]["example"]
    print(word_receive, example)
    db.examples.delete_one({"word": word_receive, "example": example})
    return jsonify({'result': 'success', 'msg': f'example #{number_receive} of "{word_receive}" deleted'})

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

▼ index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <title>Sparta Vocabulary Notebook</title>
    <meta property="og:title" content="Sparta Vocabulary Notebook"/>
    <meta property="og:description" content="mini project for Web Plus"/>
    <meta property="og:image" content="{{ url_for('static', filename='logo_red.png') }}" />
    <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
    <link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
      integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
      crossorigin="anonymous">
    <link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">

    <link href="{{ url_for('static', filename='mystyle.css') }}" rel="stylesheet">
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
      integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
      crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
      integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"></script>
    <style>
      .search-box {
        width: 70%;
        margin: 50px auto;
        max-width: 700px;
      }

      .table {
        width: 80%;
        max-width: 800px;
        margin: auto;
        table-layout: fixed;
      }

      .table th {
        border-top-style: none;
      }

      td {
        background-color: white;
        text-overflow: ellipsis;
        overflow: hidden;
        white-space: nowrap;
      }

      td > a, a:visited, a:hover, a:active {
        color: black;
      }

      tr.highlight > td {
        background-color: #e8344e;
        color: white;
      }

      tr.highlight a {
        color: white;
      }

      thead:first-child tr:first-child th:first-child {
        border-radius: 10px 0 0 0;
      }

      thead:first-child tr:first-child th:last-child {

```

```

        border-radius: 0 10px 0 0;
    }

    tbody:last-child tr:last-child td:first-child {
        border-radius: 0 0 0 10px;
    }

    tbody:last-child tr:last-child td:last-child {
        border-radius: 0 0 10px 0;
    }
}
</style>
<script>
    {% if msg %}
        alert("{{ msg }}")
    {% endif %}
    let words = {{ words|tojson }};
    let word_list = [];
    for (let i = 0; i < words.length; i++) {
        word_list.push(words[i]["word"])
    }

    function find_word() {
        let word = $("#input-word").val().toLowerCase();
        if (word == "") {
            alert("please write something first :)")
            return
        }
        if (word_list.includes(word)) {
            $('#word-${word}`).addClass('highlight').siblings().removeClass('highlight');
            $('#word-${word}').get(0).scrollIntoView();
        } else {
            window.location.href = `/detail/${word}?status_give=new`
        }
    }

</script>
</head>
<body>
    <div class="wrap">
        <div class="banner" onclick="window.location.href = '/'">
        </div>
        <div class="search-box d-flex justify-content-center">
            <input id="input-word" class="form-control" style="margin-right: 0.5rem">
            <button class="btn btn-light" onclick="find_word()"><i class="fa fa-search"></i></button>
        </div>
        <table class="table">
            <thead class="thead-light">
                <tr>
                    <th scope="col" style="width:30%">WORD</th>
                    <th scope="col">MEANING</th>
                </tr>
            </thead>
            <tbody id="tbody-box">
                {% for word in words %}
                <tr id="word-{{ word.word }}">
                    <td><a href="/detail/{{ word.word }}?status_give=old">{{ word.word }}</a></td>
                    <td>{{ word.definition|safe }}</td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</body>
</html>

```

▼ detail.html

```

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
        <title>Sparta Vocabulary Notebook</title>
        <meta property="og:title" content="Sparta Vocabulary Notebook"/>
        <meta property="og:description" content="mini project for Web Plus"/>
        <meta property="og:image" content="{{ url_for('static', filename='logo_red.png') }}" />
        <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
        <link rel="icon" href="{{ url_for('static', filename='favicon.ico') }}" type="image/x-icon">
        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
            integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
            crossorigin="anonymous">
    </head>

```

```

<link href="//maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet">
<link href="{{ url_for("static", filename="mystyle.css") }}" rel="stylesheet">
<!-- Optional JavaScript -->
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
        integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
        crossorigin="anonymous"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
        integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
        crossorigin="anonymous"></script>
</style>

    span.example {
        color: gray;
        font-size: 14px;
    }

    .btn-sparta {
        color: #fff;
        background-color: #e8344e;
        border-color: #e8344e;
    }

    .btn-outline-sparta {
        color: #e8344e;
        background-color: transparent;
        background-image: none;
        border-color: #e8344e;
    }
}
</style>
<script>
    let word = "{{ word }}"
    $(document).ready(function () {
        {% if status=="old" %}
            get_examples()
        {% endif %}
    })

    function save_word() {
        $.ajax({
            type: "POST",
            url: `/api/save_word`,
            data: {
                word_give: "{{ word }}",
                definition_give: "{{ result.definitions[0].definition }}"
            },
            success: function (response) {
                alert(response["msg"])
                window.location.href = "/detail/{{ word }}?status=old"
            }
        });
    }

    function delete_word() {
        $.ajax({
            type: "POST",
            url: `/api/delete_word`,
            data: {
                word_give: '{{ word }}',
            },
            success: function (response) {
                alert(response["msg"])
                window.location.href = "/"
            }
        });
    }

    function get_examples() {
        $("#example-list").empty()
        $.ajax({
            type: "GET",
            url: `/api/get_examples?word_give={{word}}`,
            data: {},
            success: function (response) {
                let examples = response["examples"];
                for (let i = 0; i < examples.length; i++) {
                    let example = examples[i]["example"];
                    console.log(example)
                    let html_temp = `<li id="ex-${i}">${example}&nbsp;&nbsp;&nbsp;&nbsp;<a
                        href="javascript:delete_ex(${i})">delete</a></li>`
                    $("#example-list").append(html_temp)
                }
            }
        });
    }
}

```

```

function add_ex() {
    let new_ex = $('#new-example').val();
    if (!new_ex.toLowerCase().includes(word.toLowerCase())) {
        alert('the word '${word}' is not included.');
```

return;

}

console.log(new_ex)

\$.ajax({

type: "POST",

url: '/api/save_ex',

data: {

word_give: word,

example_give: new_ex

},

success: function (response) {

get_examples();

\$('#new-example').val("");

}

});

}

function delete_ex(i) {

console.log("deleting", i)

\$.ajax({

type: "POST",

url: '/api/delete_ex',

data: {

word_give: word,

number_give: i

},

success: function (response) {

get_examples()

}

});

}

</script>

</head>

<body>

<div class="wrap">

<div class="banner" onclick="window.location.href = '/'">

</div>

<div class="container">

<div class="d-flex justify-content-between align-items-end">

<div>

<h1 id="word" style="display: inline;">{{ result.word }}</h1>

{% if result.pronunciation %}

<h5 id="pronunciation" style="display: inline;">{{ result.pronunciation }}</h5>

{% endif %}

</div>

{% if status=="new" %}

<button id="btn-save" class="btn btn-outline-sparta btn-lg" onclick="save_word()"><i class="fa fa-floppy-o" aria-hidden="true"></i></button>

{% else %}

<button id="btn-delete" class="btn btn-sparta btn-lg" onclick="delete_word()"><i class="fa fa-trash-o" aria-hidden="true"></i></button>

{% endif %}

</div>

<hr>

<div id="definitions">

{% set definitions = result.definitions %}

{% for definition in definitions %}

<div style="padding:10px">

<i>{{ definition.type }}</i>

{{ definition.definition.encode('ascii', 'ignore').decode('utf-8') }}

{% if definition.example %}

{{ definition.example.encode('ascii', 'ignore').decode('utf-8') }}

{% endif %}

</div>

{% endfor %}

</div>

{% if status=="old" %}

<div id="examples" class="container">

<h3 style="text-align: center;margin-bottom:1rem">Write your own sentences!</h3>

<ul id="example-list">

<li id="ex-0">This sentence contains the word 'word'. delete

<li id="ex-1">I don't like using the MS Word program. delete

<div class="d-flex justify-content-between" style="margin-left:20px;">

<input id="new-example" class="form-control form-control-sm" style="margin-right: 0.5rem">

```

        <button class="btn btn-outline-secondary btn-sm" onclick="add_ex()">add</button>
      </div>
    </div>
  {% endif %}
</div>
</body>
</html>

```

▼ mystyle.css

```

.wrap {
  background-color: RGBA(232, 52, 78, 0.2);
  min-height: 100vh;
  padding-bottom: 50px;
}

.banner {
  width: 100%;
  height: 200px;

  background-color: white;
  background-image: url('logo_red.png');
  background-position: center;
  background-size: contain;
  background-repeat: no-repeat;

  cursor: pointer;
}

.container {
  width: 80%;
  max-width: 800px;
  margin: 30px auto;
  padding: 20px;
  background-color: white;

  border: solid 1px gray;
  border-radius: 10px;
}

```

Copyright © TeamSparta All rights reserved.