**[IS216] Extra Exercises - Week 6 - Javascript JSON/Ajax**

**Objectives**

● To understand how dynamic, interactive web pages can be built using JavaScript

● To use JavaScript to build dynamic, interactive HTML web pages.

**Instructions**

● Questions with no asterisk mark are easy peasy.

● Questions marked with \* are slightly challenging.

● Questions marked with **\***\* are challenging.

● Questions marked with \*\*\* are very challenging.

**Download**

● **Resources**: Click [here](https://drive.google.com/drive/folders/1rpYGrTXTwU74wqdQeXAXKAnUAbPOakxd?usp=sharing)

* **Sample Solutions:** [here](https://drive.google.com/drive/folders/1QEl1f50Who54vY3TeKnrQwbjidpNqlQz?usp=sharing)

**NOTE:** If you spot any mistakes/errors in the questions, please contact your instructors by email and state the issues. We will try to address it as soon as possible.

[**Chat (\*)**](#_a0w781gwomy3) **2**

[**Internet Archive Search (\*)**](#_or32b7391d8s) **4**

[Part 1 - Search](#_6fsgc77aw7gw) 4

[Part 2 - Toggle shorter & longer description (\*\*)](#_fozfdrcpotx9) 7

[**Weather forecast (\*)**](#_xelk8b3uh7jl) **7**

[Part 1 - Singapore 4-days weather forecast (\*\*\*)](#_r0tod7jirrol) 7

[Part 2 - Singapore 24-hour weather forecast (\*)](#_da6dn5uu8mgi) 8

[Part 3 - Singapore 2-hourly weather forecast (\*\*\*)](#_3ihewdgxqlhk) 9

[Part 4 - Navigation bar (\*)](#_mxop6vmcyixf) 9

[**Urban Dictionary Search (\*)**](#_ur5v58nu0w56) **10**

[Search & display results](#_6ji6uqmxugmu) 10

# Chat (\*)

*#ajax #bootstrap*

**Resource:** Folder ‘chat’

1. Create a folder ‘chat’ in your WAMP web root folder and
   1. Copy the given resource into it.
   2. Create chat.html file

If done correctly, the folder structure should look like this:

<web root>

+- chat\

|

+- server\ (given resource)

| +- chat.php

| +- chat.json

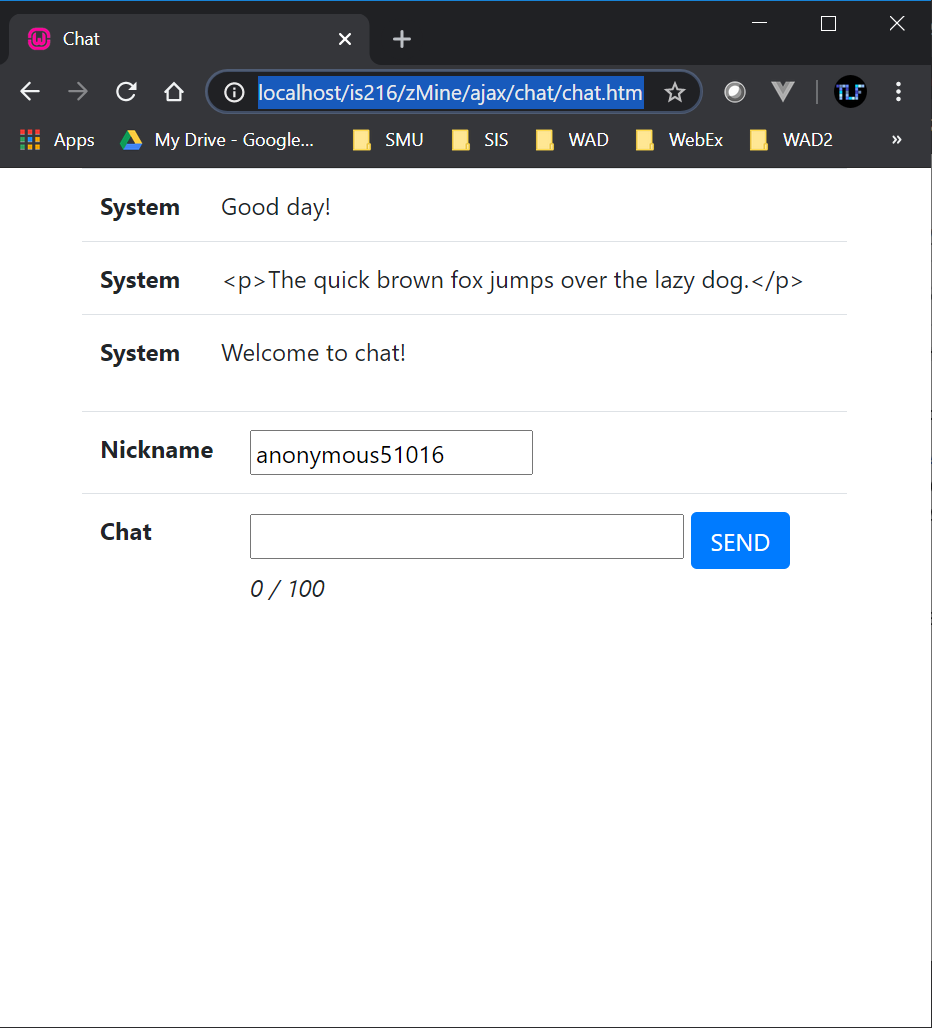
|

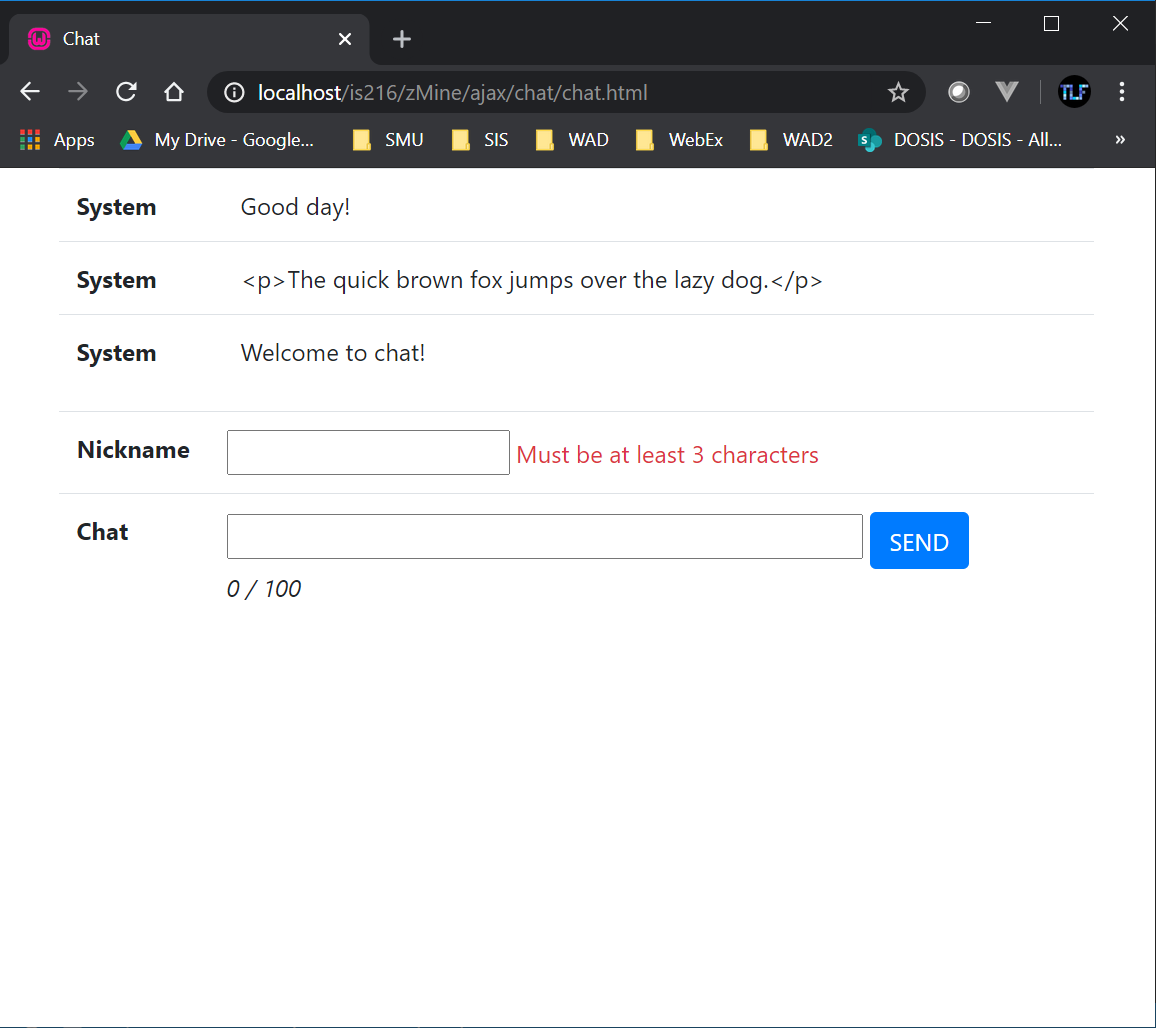
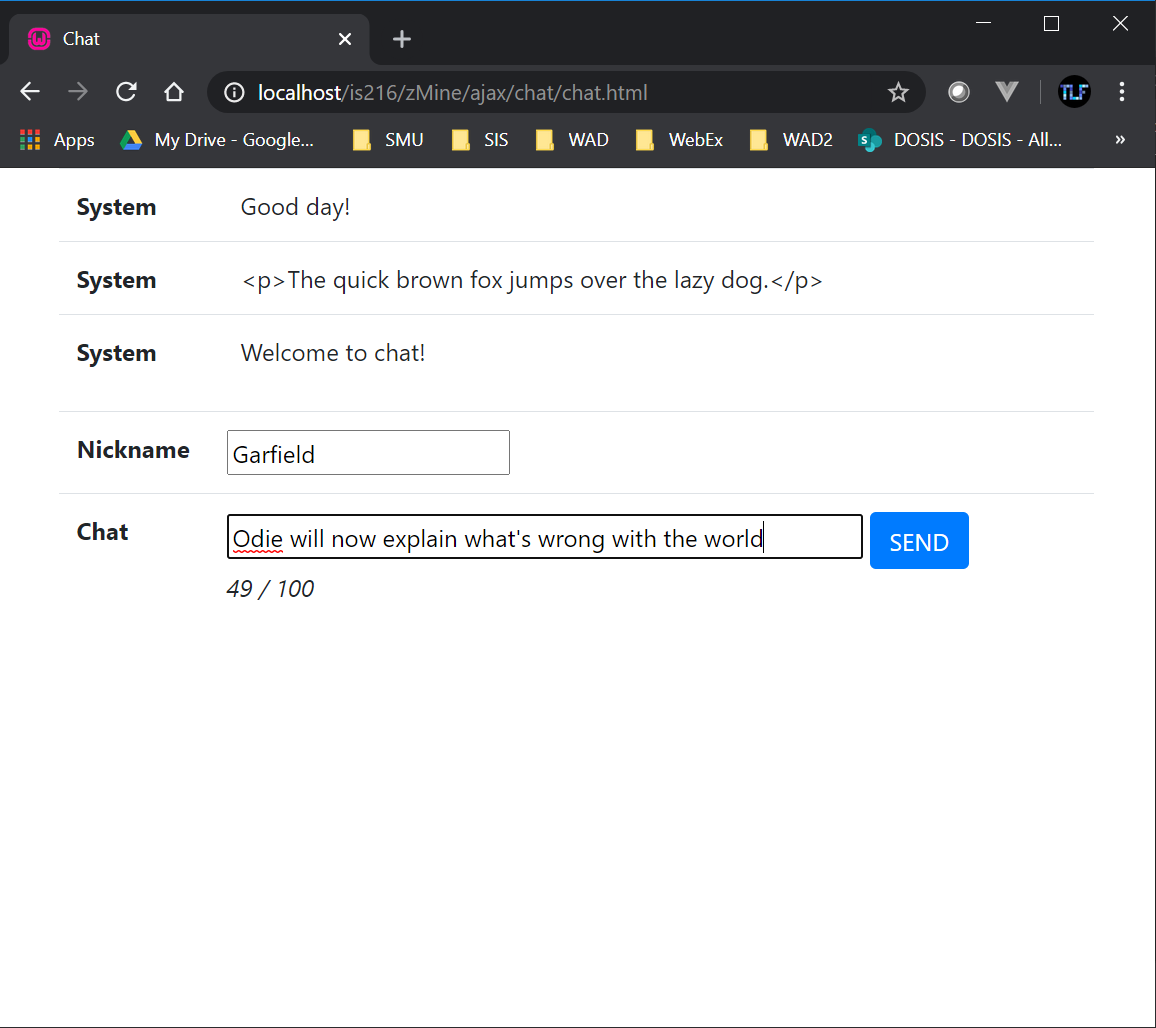
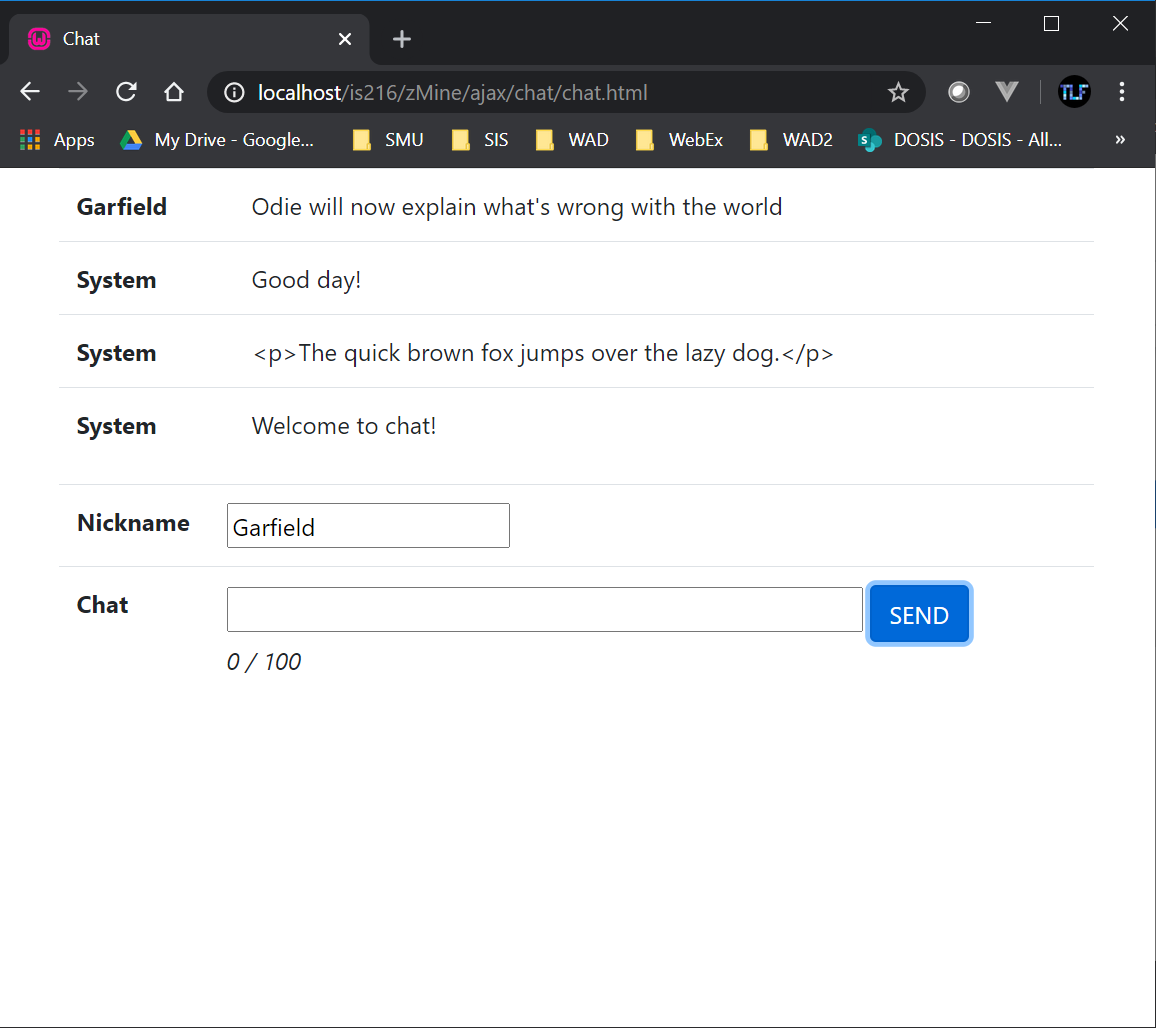
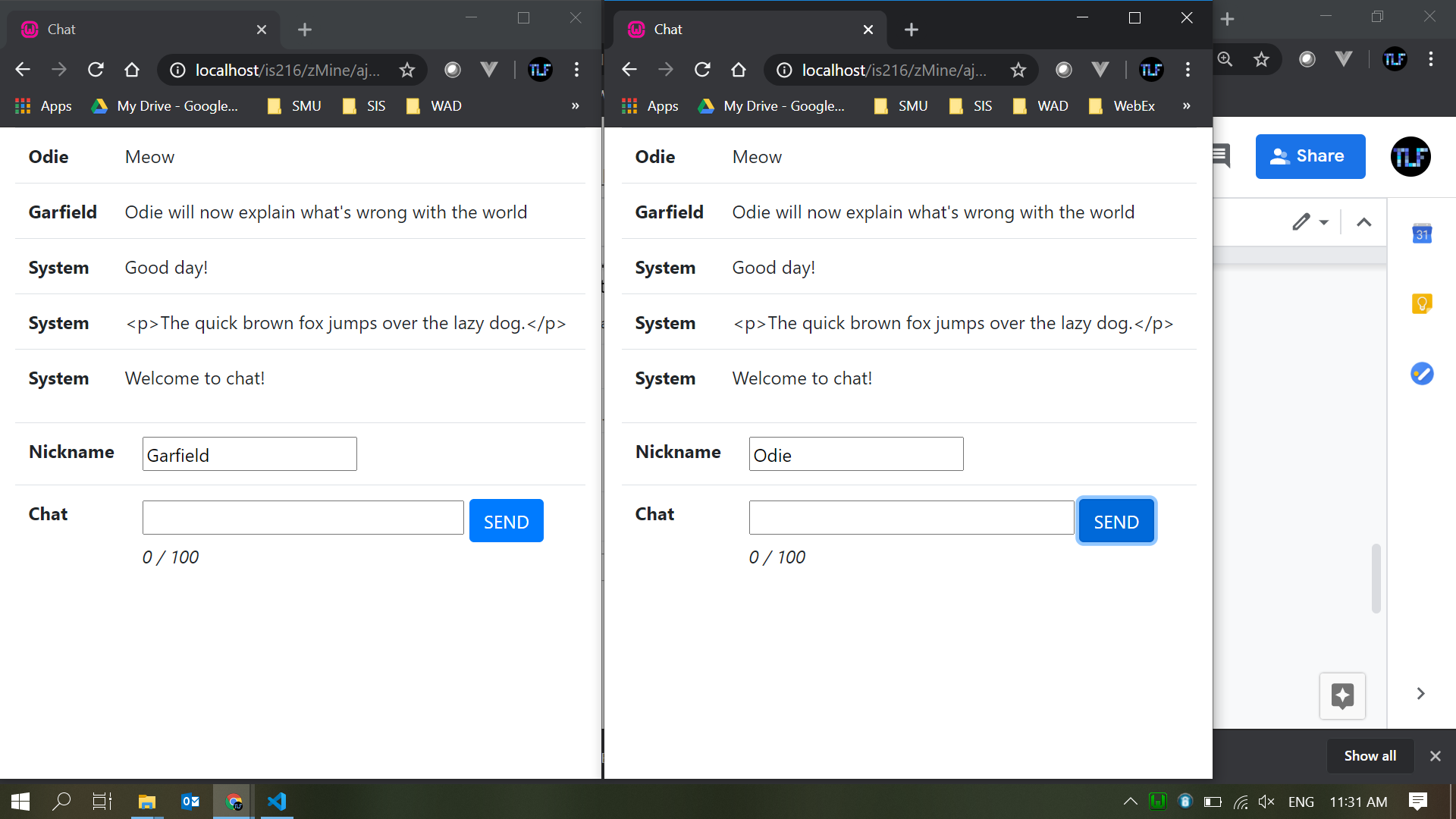
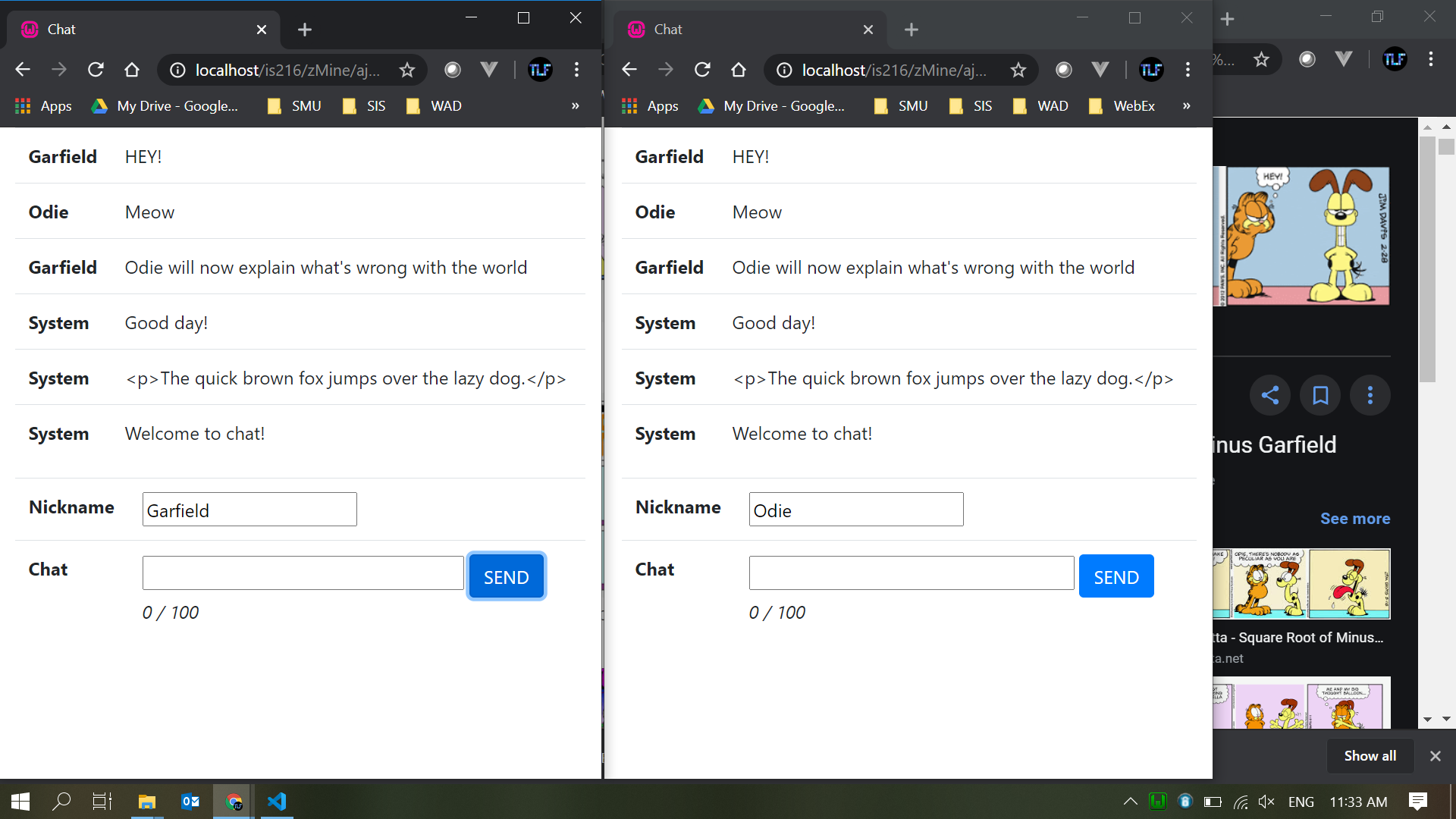
+- chat.html (to do)

1. The **chat.php** simulates mockup JSON Web API for sending and receiving text messages.

Look at the given code to understand the required HTTP protocol and parameters, and the expected response.

1. **NOTE:** The chat.php is hosted on your local WAMP server.
   1. Make your AJAX call to <http://localhost/chat/server/chat.php>. If you have placed the resources in another folder, remember to modify the URL accordingly.
   2. The solution’s chat.html accesses the web API using relative URL. If you wish to run the solution, you can either
      1. Modify the URL accordingly, or
      2. Access chat.html via your WAMP server; i.e. [http://localhost/chat/chat.html](http://localhost/mart/mart.html)
2. You are to create chat.html that looks like this



1. Upon page load, call chat.php to get the past messages.
2. **Notice** that the “<p> … </p>” is displayed safely. Your code should handle the HTML special characters safely. **(\*\*)**
3. Generate a random nickname for the current user. E.g. prefix “anonymous” and append with a random integer value.
4. If the user enters a nickname shorter than 3 characters, display an error message immediately.  
   
5. As the user types his chat message, the number of characters (below the input field) is updated in real time.  
   
6. When the user presses “ENTER” or click the “SEND” button, call the web API to send the message. The new message is displayed at the top.  
   
7. The page updates the messages every 1 second so that the user can see the latest messages from other users.
   1. Odie replies. Garfield receives it without any additional action  
      
   2. Garfield replies. Similarly, Odie receives it without any additional action  
      

# Internet Archive Search (\*)

*#ajax #bootstrap*

**Resource:** nil

## Part 1 - Search

[Internet Archive](https://archive.org/) is a non-profit digital library offering free universal access to books, movies & music, as well as 439 billion archived web pages. It provides a search API to sieve through its content.

The following sample URL looks for content with “funny cat” in its title or description and returns the results as JSON.

https://archive.org/advancedsearch.php?fl[]=identifier&fl[]=title&fl[]=description&fl[]=mediatype&fl[]=year&output=json&q=**description:"funny cat" OR title:"funny cat"**

Sample results

|  |
| --- |
| {  "responseHeader": {  "status": 0,  "QTime": 62,  "params": {  "query": "description:\"funny cat\" OR title:\"funny cat\"",  "qin": "description:\"funny cat\" OR title:\"funny cat\"",  "fields": "identifier,title,description,mediatype,year",  "wt": "json",  "rows": "5",  "start": 0  }  },  "response": {  "numFound": 49,  "start": 0,  "docs": [{  "description": "MANGA: The Salary Man & Tofu the Funny Cat",  "identifier": "manga\_The\_Salary\_Man\_Tofu\_the\_Funny\_Cat",  "mediatype": "texts",  "title": "MANGA: The Salary Man & Tofu the Funny Cat"  }, {  "description": "Funny Cat",  "identifier": "pisica",  "mediatype": "movies",  "title": "pisica",  "year": "2019"  }, {  "description": "funny cat",  "identifier": "BeautifulKittenMeowingHermosoGatitoMaullandoAdorableCute",  "mediatype": "movies",  "title": "Beautiful Kitten Meowing Hermoso Gatito Maullando Adorable Cute"  }, {  "description": "Funny cat",  "identifier": "IMG0254",  "mediatype": "movies",  "title": "IMG 0254",  "year": "2013"  }, {  **"description": ["24 unnumbered pages : 23 cm", "This rebus book is about Otto, a funny cat", "\"Ages 3-5\"--Cover", "Includes 24 flash cards", "\"Ages 3-5\"--Cover", "This rebus book is about Otto, a funny cat"]**,  "identifier": "ottocat0000herm",  "mediatype": "texts",  "title": "Otto the cat",  "year": "1995"  }, *...*  ]  }  } |

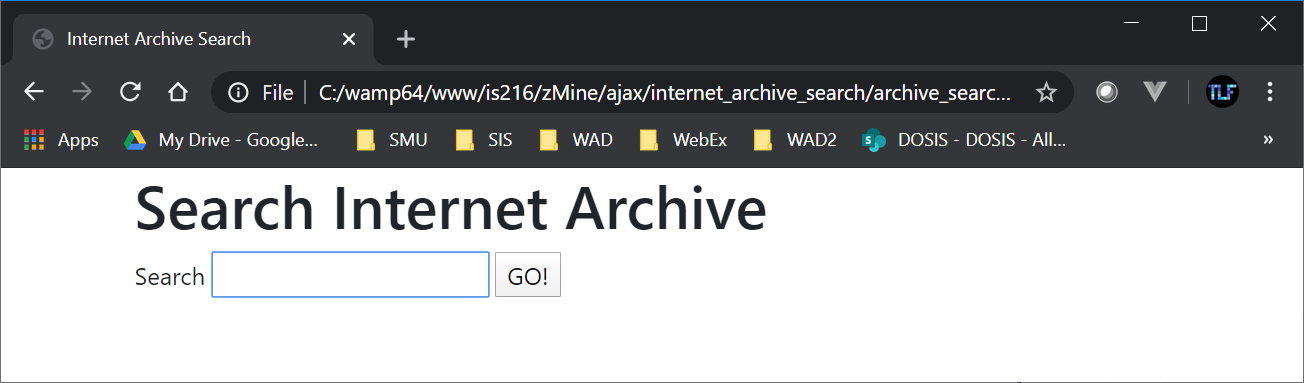
Refer to [Internet Archive: Search Engine](https://archive.org/advancedsearch.php) for more details about the API.

Note the following for each element in the property “response.docs”

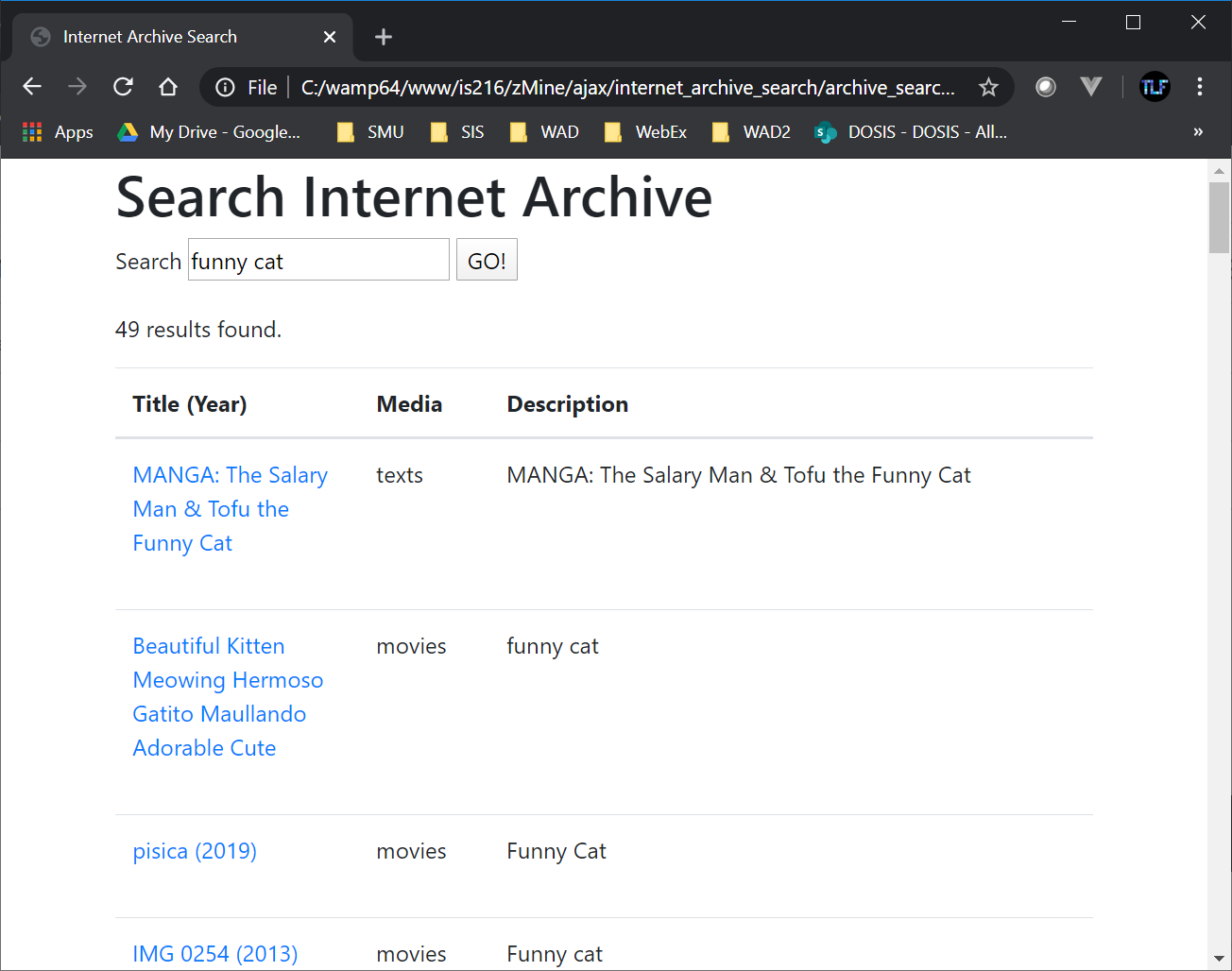
1. Property “description” can be either a string or an array
   1. E.g. The last element’s description is an array
   2. <https://www.w3schools.com/jsref/jsref_isarray.asp>
2. Property “year” may not be present
   1. E.g. The first element does not have a year.
   2. Hint: obj.hasOwnProperty( “propName” ) returns true if obj has a property “propName”. See <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/hasOwnProperty>
3. To view a particular result, link to https://archive.org/details/<identifier> where <identifier> is to be replaced by the value of property “identifier”
   1. E.g. The last element’s identifier is “ottocat0000herm”, thus, link to <https://archive.org/details/ottocat0000herm>

Create a search page that allows users to enter a search phrase and retrieve up to 50 results.

Upon initial loading,



After clicking ‘Go!’ button,



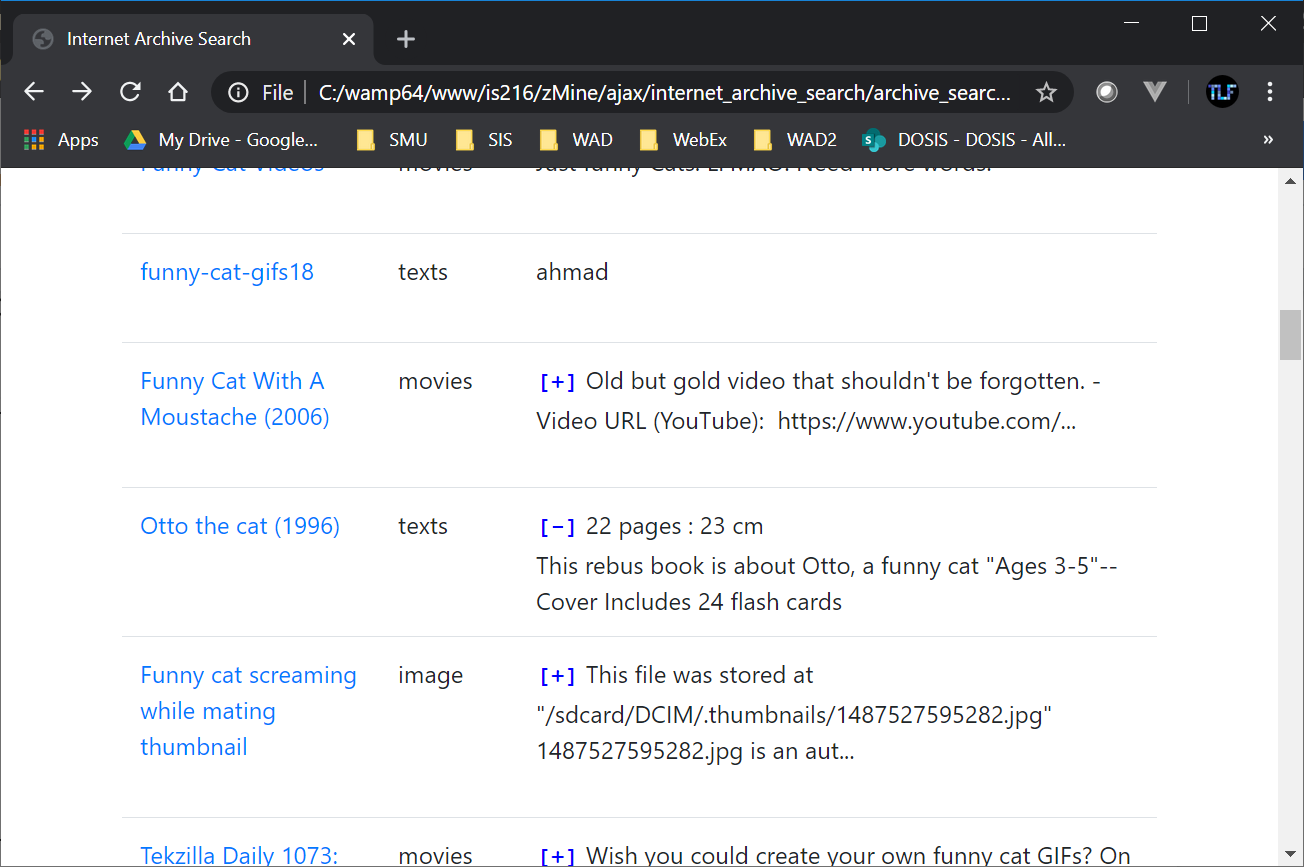
* Title (Year) takes up 25% of the width and is linked to the result; i.e. https://archive.org/details/<identifier>
* If description is an array, each element of the description is on a separate line. See the 3rd result in the screenshot below

## Part 2 - Toggle shorter & longer description (\*\*)

If the description is more than 100 characters,

1. Display the first 97 characters and append with “...”. See the 2nd and 4th results in the screenshot below.
2. Upon clicking the “[+]”, display the full description. See the 3rd result below.
3. Upon clicking the “[-]”, display the shorter description again.

If the description is 100 characters or less, just display it. See the 1st result below.



References:

* <https://archive.org/>
* <http://blog.archive.org/developers/>
* <https://archive.org/advancedsearch.php#raw>

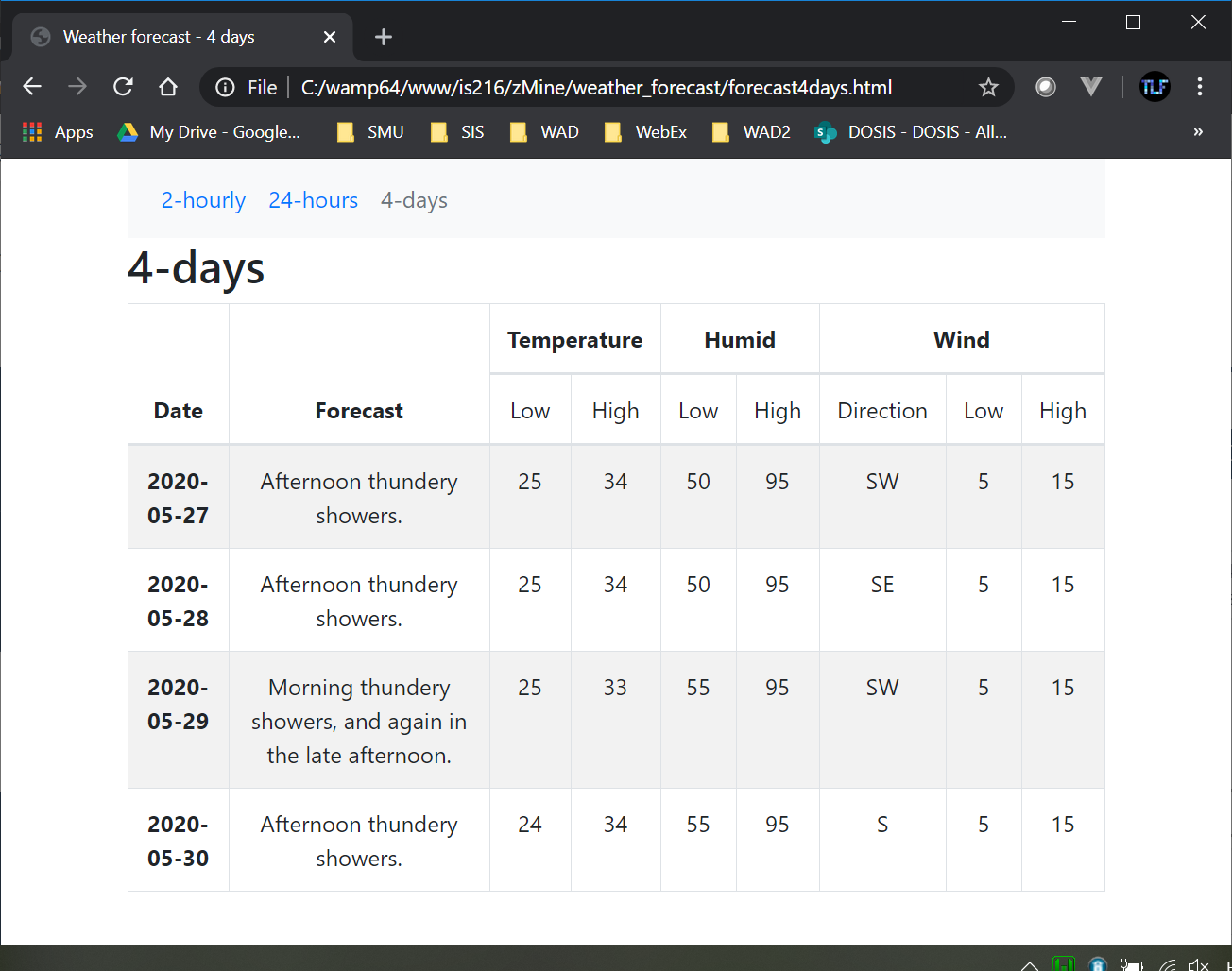
# Weather forecast (\*/\*\*\*)

*#ajax #bootstrap*

**Resource:** nil

## Part 1 - Singapore 4-days weather forecast (\*\*\*)

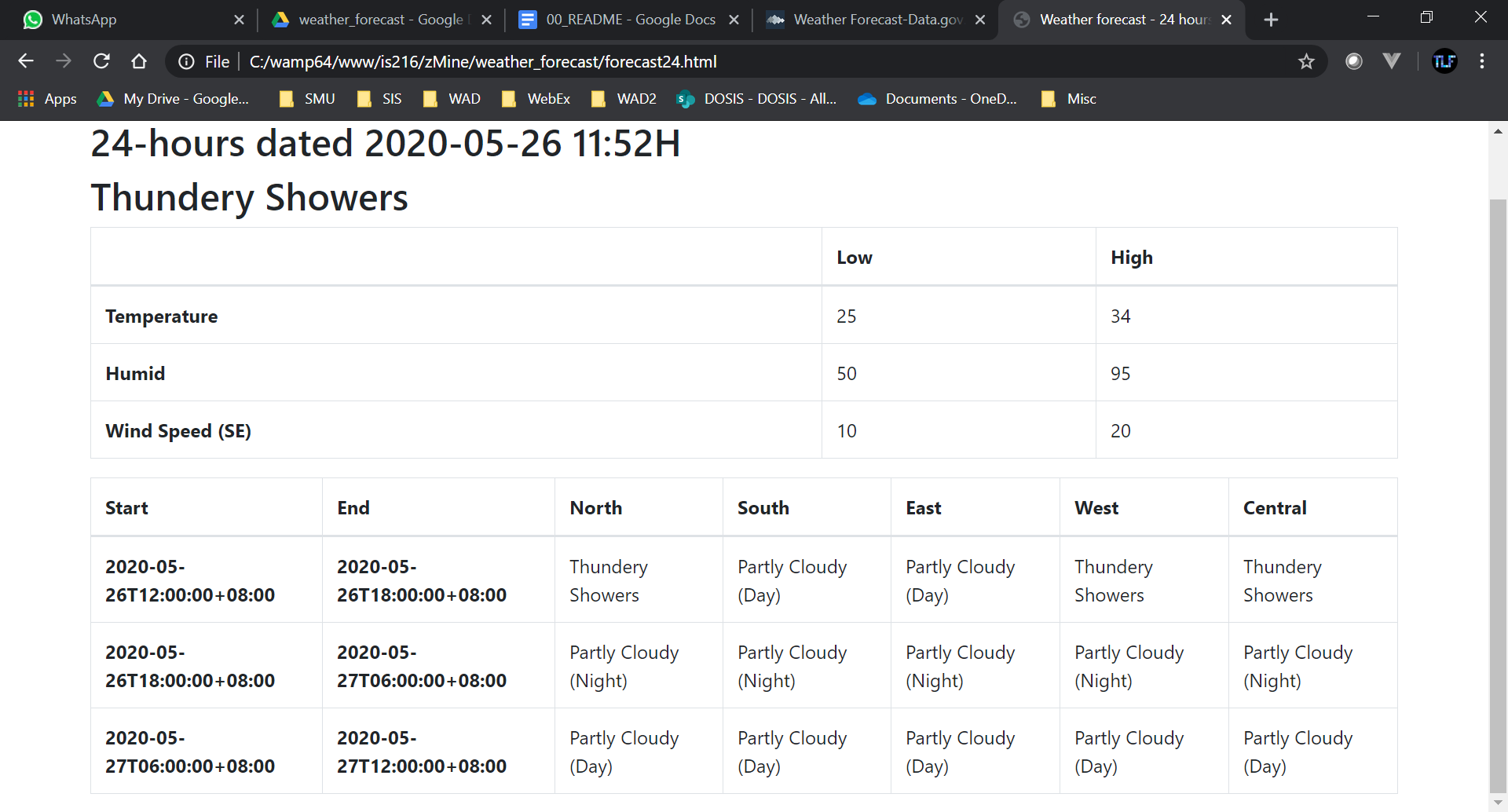
Create the following web page using<https://data.gov.sg/dataset/weather-forecast?resource_id=9a8bd97e-0e38-46b7-bc39-9a2cb4a53a62> API to display the 4-days weather forecast.



Note that the JSON response has multiple forecasts with different update times. Display the **last** item aka latest update time.

## Part 2 - Singapore 24-hour weather forecast (\*)

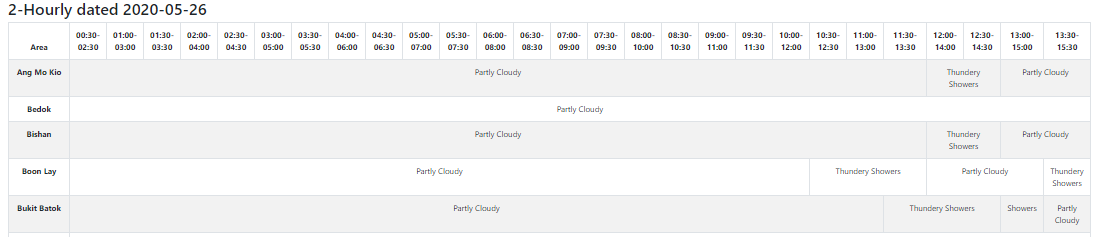
Create the following web page using <https://data.gov.sg/dataset/weather-forecast?resource_id=9a8bd97e-0e38-46b7-bc39-9a2cb4a53a62> API to display the 24-hour weather forecast.



Similarly, the JSON response has multiple forecasts with different update times. Display the **last** item aka latest update time.

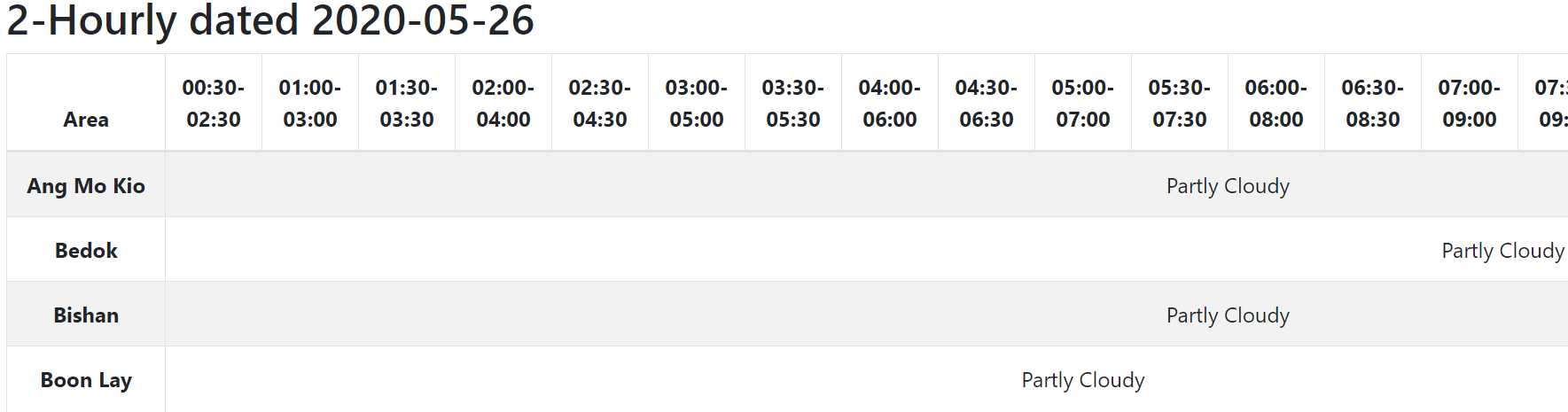
## Part 3 - Singapore 2-hourly weather forecast (\*\*\*)

Create the following web page using <https://data.gov.sg/dataset/weather-forecast?resource_id=9a8bd97e-0e38-46b7-bc39-9a2cb4a53a62> API to display the 2-hourly weather forecast.



There are more rows below that are not shown.

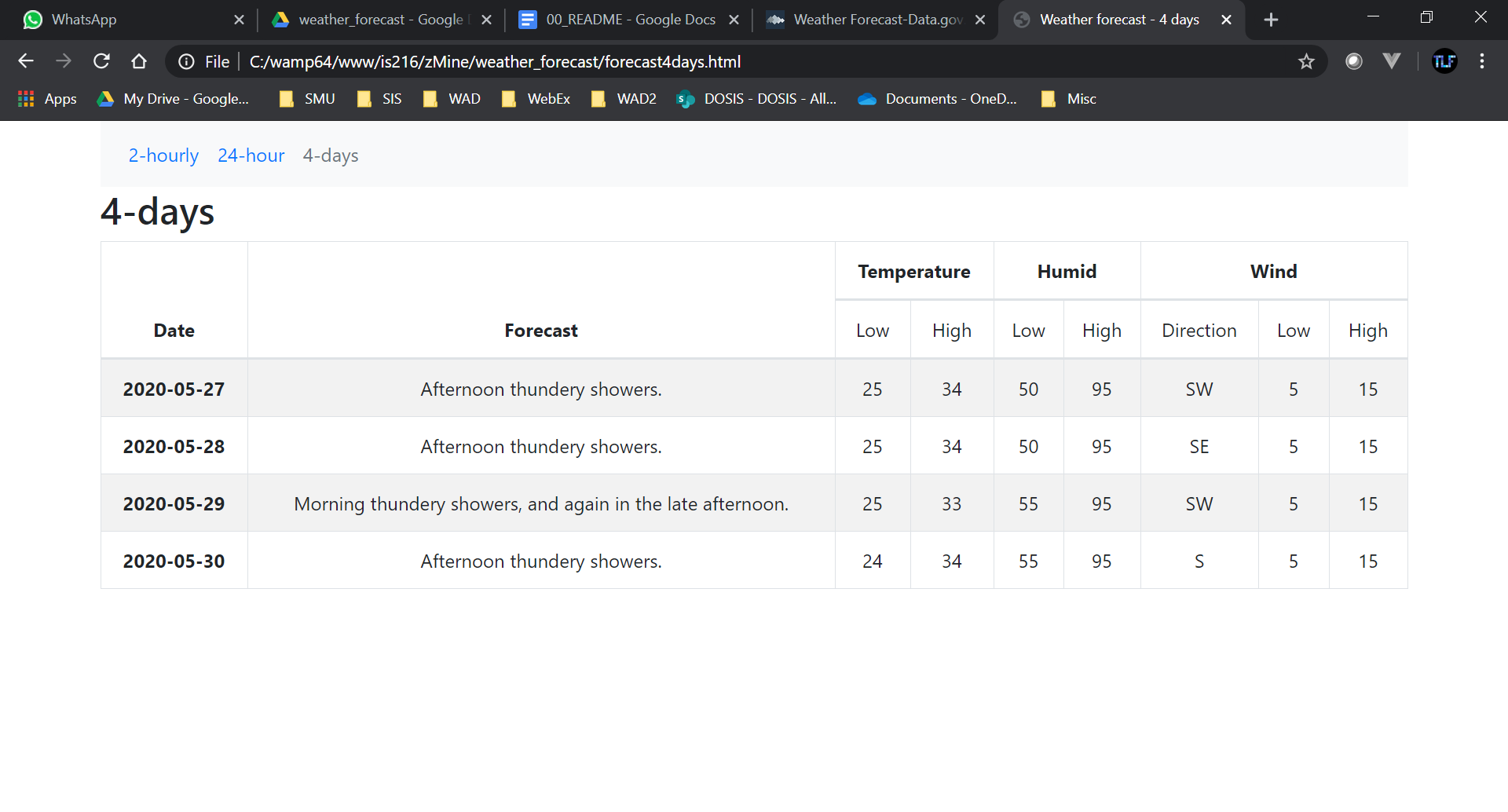
A zoom-in screenshot



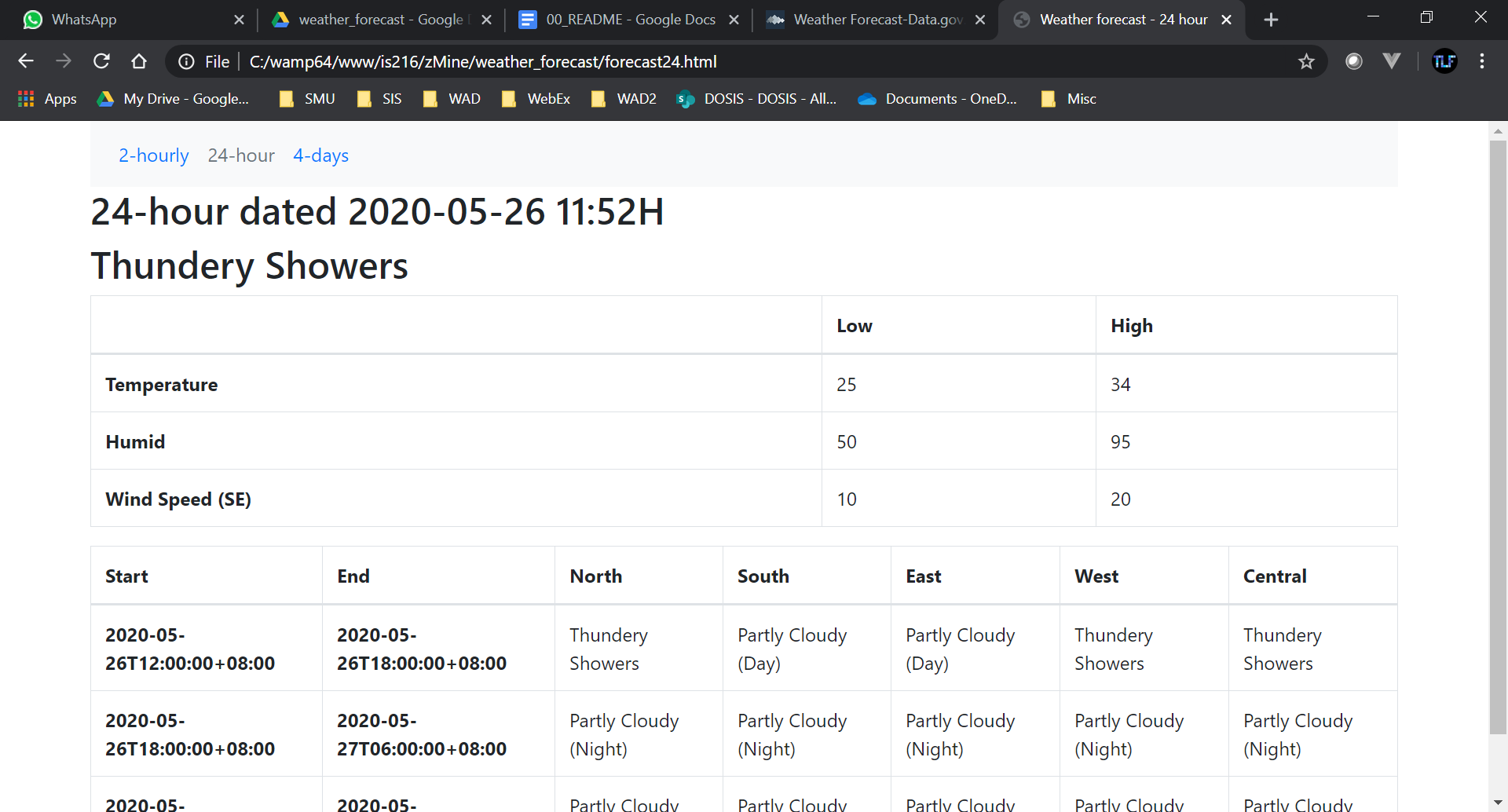
1. The JSON response may have multiple forecasts for 2-hourly time slots with different update times; i.e. there may be 2 or more elements in property “items” with “validity\_period.start” equals to 10:00:00. Display the **last** item aka latest update time.
2. There are forecasts “Partly Cloudy (Night)” and “Partly Cloudy (Day)”. Display both of them as just “Partly Cloudy”.
3. For a particular area, merge continuous cells that have the same forecast; i.e. colspan accordingly and display the forecast once.

## Part 4 - Navigation bar (\*)

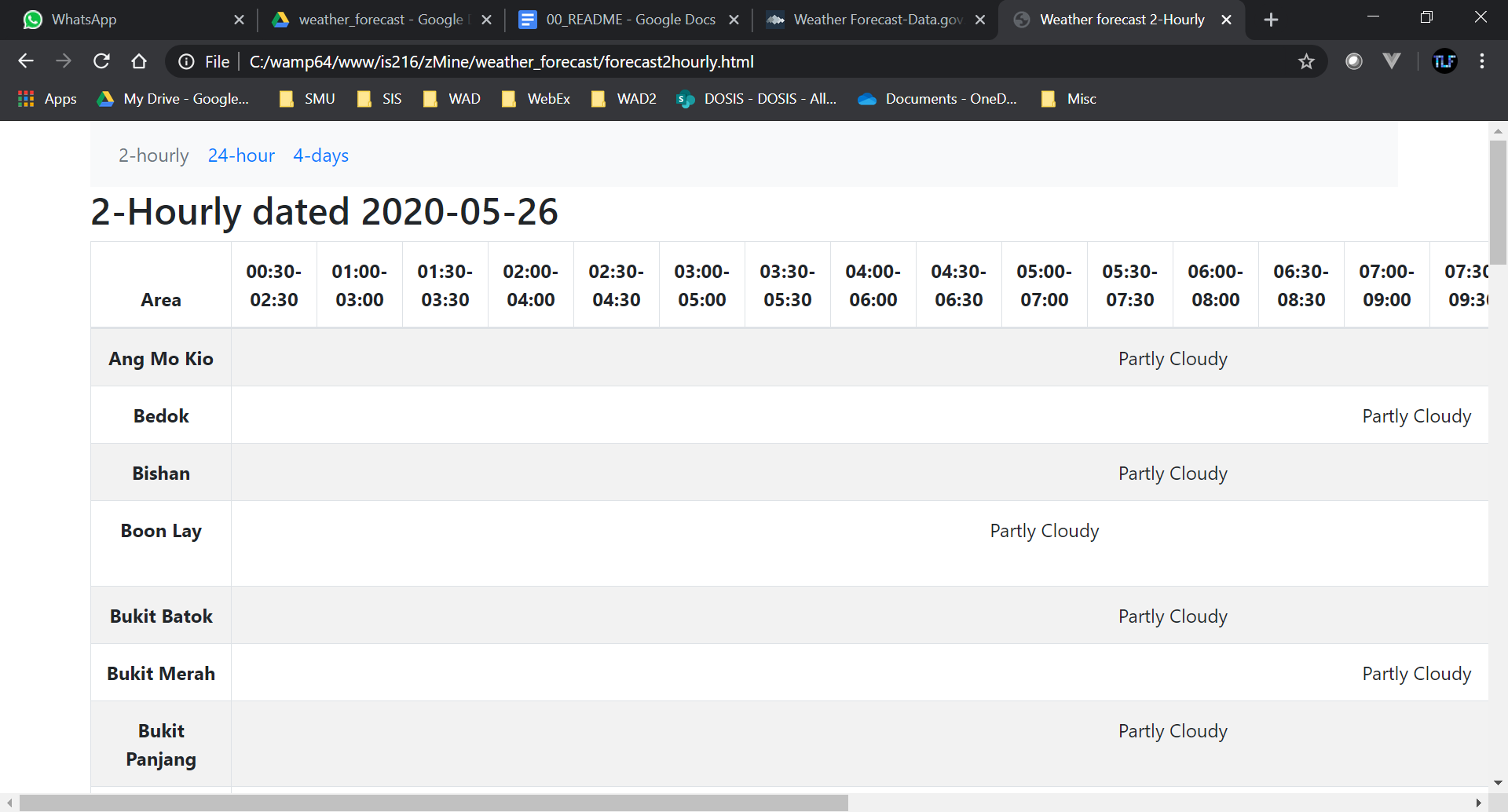
Add a bootstrap navigation bar to link up all 3 pages.



Note: 4-days link is disabled.



Note: 24-hour link is disabled.



Note: 2-hourly link is disabled.

# Urban Dictionary Search (\*)

*#ajax #bootstrap*

**Note:** No resource given

## Search & display results

[Urban Dictionary](https://rapidapi.com/community/api/urban-dictionary) is a crowdsourced online dictionary for slang words and phrases, operating under the motto "Define Your World." The API provided is obtained through RapidAPI. You will need to register to access this. Simply use your SMU google account on the registration page.

The following sample URL looks for the term “programmer” and returns 10 results.

https://mashape-community-urban-dictionary.p.rapidapi.com/define?term=programmer

Sample results (there are 10 but showing only 3 with the last 7 removed for easier reading)

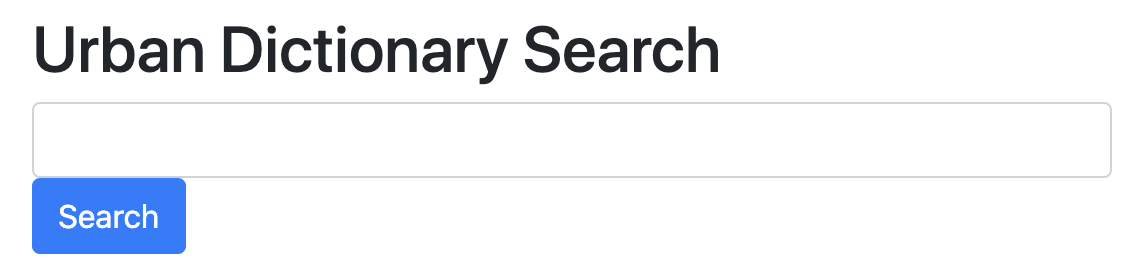
|  |
| --- |
| {  "**list**": [  {  "definition": "A person who is [paid] to professionally [scream] at a [computer].",  "permalink": "http://programmer.urbanup.com/13009271",  "thumbs\_up": 62,  "sound\_urls": [],  "author": "MidnightWasTwoHoursAgo",  "word": "Programmer",  "defid": 13009271,  "current\_vote": "",  "written\_on": "2018-06-18T00:00:00.000Z",  "example": "Programmer: \"AAAAAAAAAHHHHHHHHHHHHHHHHHHHHH-oh, it [works].\"",  "thumbs\_down": 4  },  {  "definition": "pro·[gram]·[mer] (n) An [organism] capable of converting caffeine into code.",  "permalink": "http://programmer.urbanup.com/1546278",  "thumbs\_up": 715,  "sound\_urls": [],  "author": "its\_allgood",  "word": "Programmer",  "defid": 1546278,  "current\_vote": "",  "written\_on": "2005-12-08T00:00:00.000Z",  "example": "Programmer A \"You started on that program yet? When's it due anyway?\"\r\n\r\nProgrammer B \"Nah, not yet, and its due tomorrow morning at eight!\"\r\n\r\nProgrammer A \"Damn, you [gonna get it] done in time?\"\r\n\r\nProgrammer B \"Yeah, I've got a bottle of [Stacker] [II's] and a pot of coffee, I should be fine.\"",  "thumbs\_down": 118  },  {  "definition": "One who can [read] [the following] [example]",  "permalink": "http://programmer.urbanup.com/2616331",  "thumbs\_up": 359,  "sound\_urls": [],  "author": "John Standish",  "word": "programmer",  "defid": 2616331,  "current\_vote": "",  "written\_on": "2007-10-04T00:00:00.000Z",  "example": "if( ![User].IsProgrammer ){\r\n [return] \"Everyone [Else]\";\r\n}",  "thumbs\_down": 66  }  ]  } |

Note the following for the returned data

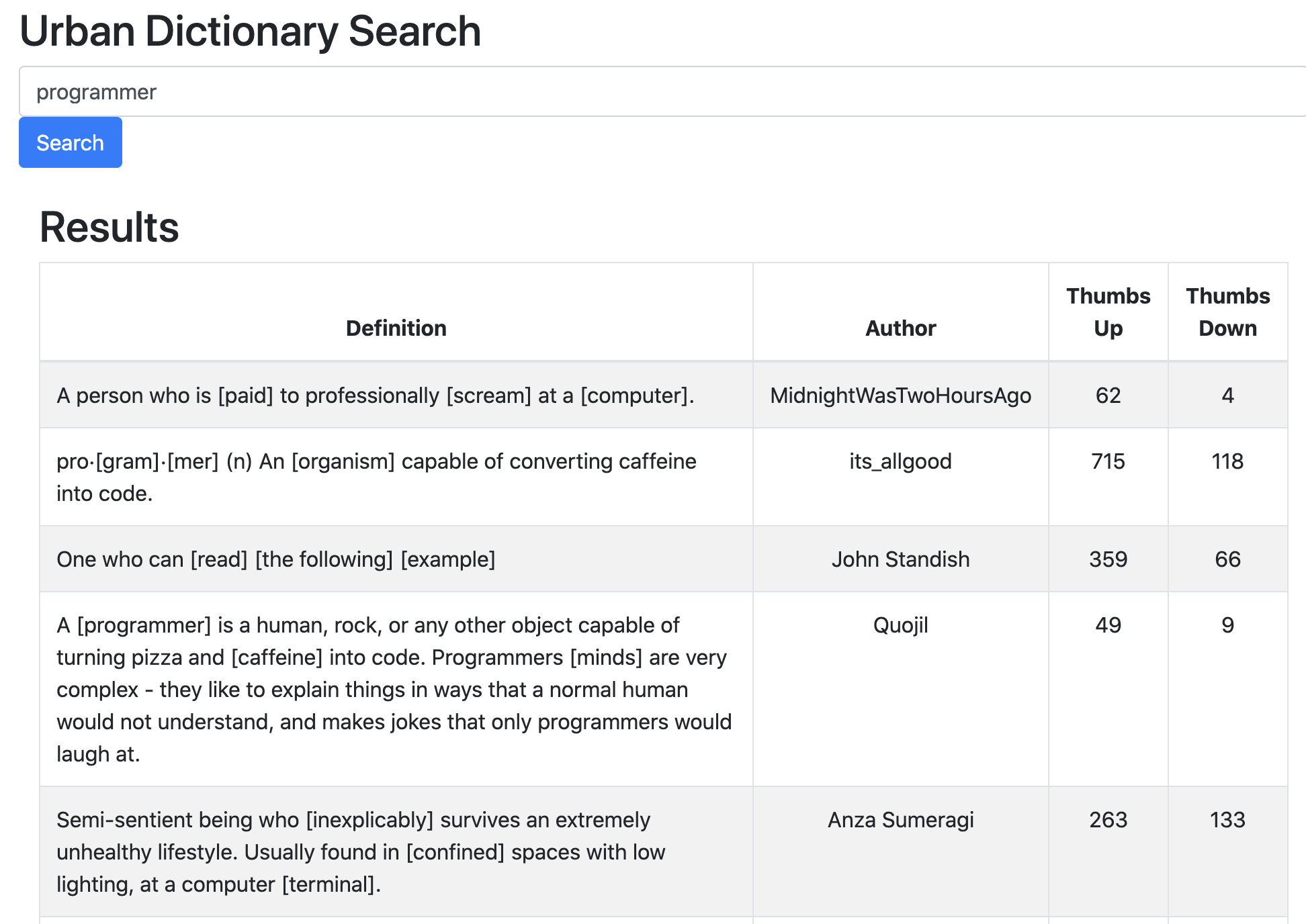
1. It is returned as an object with the key “list”, the value is an array of objects
2. When there are no results, the list will still be returned with an empty array.

Create a search page that allows users to enter a search phrase.

Upon initial loading,



After clicking ‘Search’ button,



* Display the list of results for the following fields:
  + definition
  + author
  + thumbs\_up
  + thumbs\_down
* There are more fields than the chosen field. You can look at the returned JSON data to see all the fields and use more of the fields if you like.

If there are no results,

