Introduction to Web API & Database

Kevin Zhengxu Xia
UCLA Computer Science
zxxia@g.ucla.edu
07/18/2018 – 07/19/2018

Variability Expedition

UCLA

Overview of Lecture

Part-1: Introduction to Web API

- Web API: Simple Example
- Web API vs Website
- Web API in more detail
- Project -1

Part 1 Summary: Web API's are part of larger ecosystem of web development. We will try to try to touch its basics and introduce the Web API's.

Part-2: Introduction to Databases

- Data
- Databases
- Databases: types
- MySql
- MongoDB
- Exercises

Part 2 Summary: Introduce data and ways to store and query it using databases.

Part-3: Project using Web API and MongoDB.

Web API : Simple Example

Application programming interface (API): Defines method of communication between various software components.

Simple terms: *Web API* provides ways to use computing (Query data, Store data, perform calculations etc) facilities over the internet.

Web API is an evolution of web service.

Hi, this is Mr. Sandy. Can you tell me nearby hotels ?

Hello, Mr. Sandy. The nearby hotels are Waldorf Astoria Beverly Hills, Royal Palace Westwood Hotel, Holiday Inn Express West Los Angeles.

Example: Web API to search nearby hotels

Web API Example

Simple terms: *Web API* provides ways to use computing (Query data, Store data, perform calculations etc) facilities over the internet.

Hi, this is Mr. XXX. Can you tell me nearby hotels ?

Hello, Mr. XXX. The nearby hotels are Waldorf Astoria Beverly Hills, Royal Palace Westwood Hotel, Holiday Inn Express West Los Angeles.

What are different components of web api in this example ?

Hint:
- How to Communicate/Contact to remote server?
- How to understand what we are sending and receiving?
- What is my query?

Few other things:
- How to know we are serving right person ?

Web API Example

Simple terms: *Web API* provides ways to use computing (Query data, Store data, perform calculations etc) facilities over the internet.

Hi, this is Mr. XXX. Can you tell me nearby hotels ?

Hello, Mr. XXX. The nearby hotels are Waldorf Astoria Beverly Hills, Royal Palace Westwood Hotel, Holiday Inn Express West Los Angeles.

What are different components of web api in this example ?

Hint:

How to Communicate/Contact to Web service? Communication protocols (HTTP)

How to understand what we send and receive? Formatted request & response (Generally: Json)

What is my query? In request: Hotels near (how much near ?) me (what is my location ?)

Few other things:

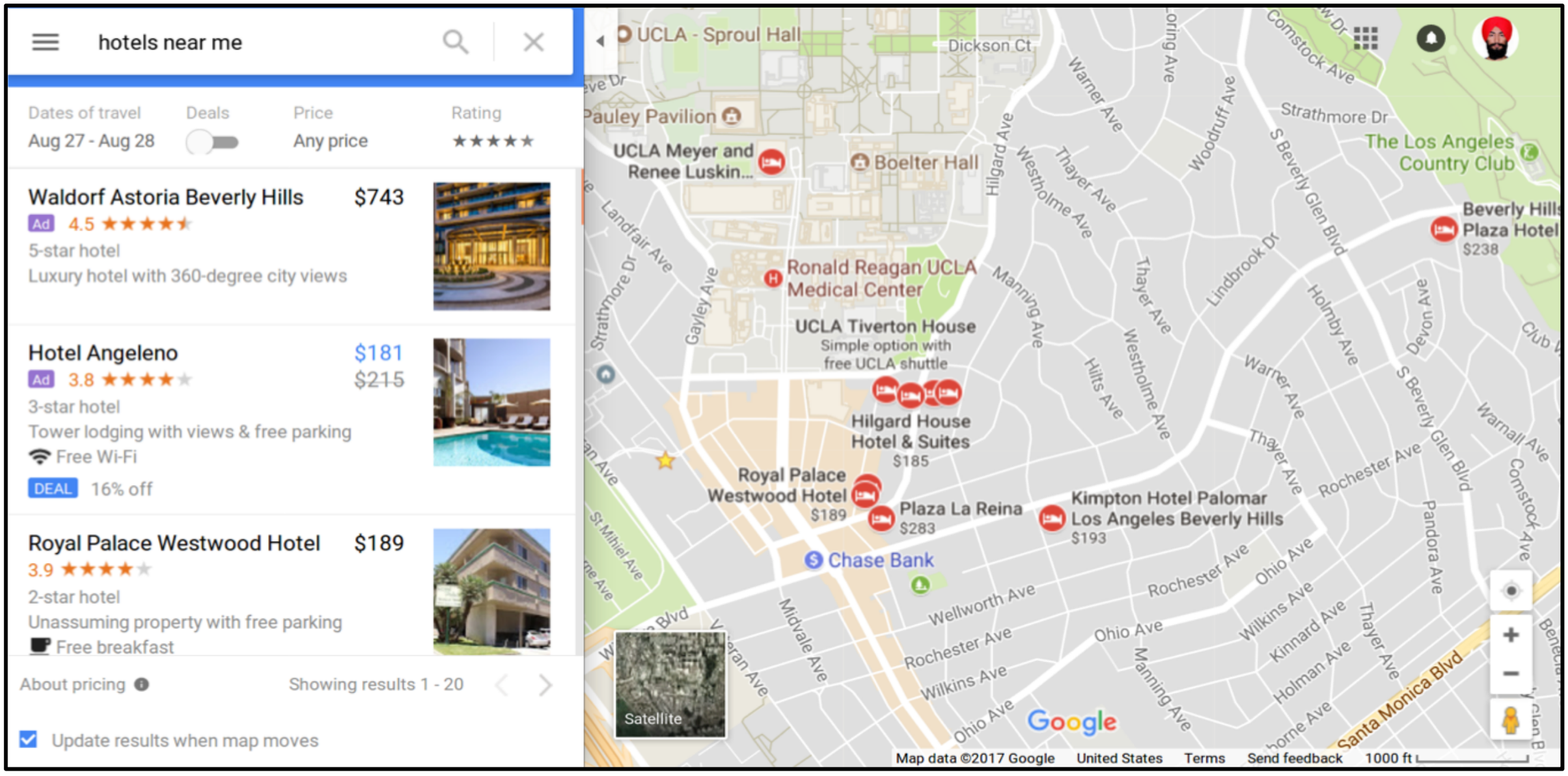
How to know we are serving right person ? Authentication

Verifying that the service requester is actually Mr. Sandy.

Web API vs Website

Website is collection of similar web pages. Web page is a document displayed by web browser.

Website may or may not use web api to do query and get data from multiple servers. *Website take care of data presentation to user but web api doesn't do that.*

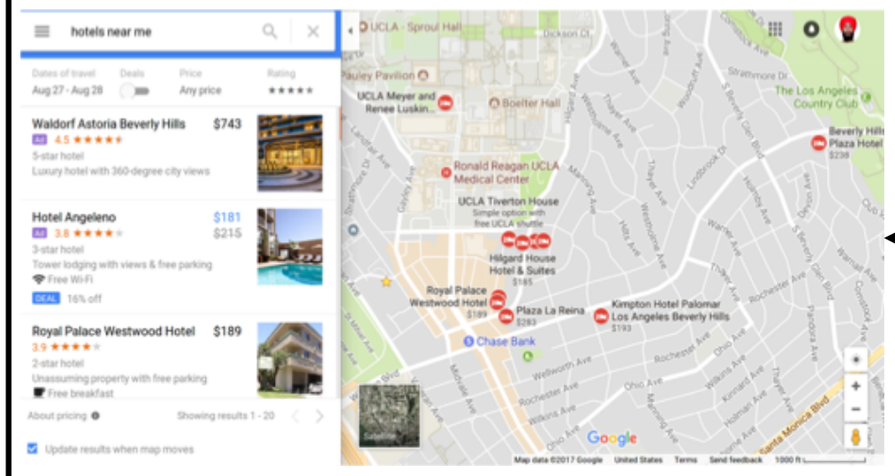


Web API vs Website

Website take care of data presentation to user but web api doesn't do that.

WebPage: https://www.google.com/maps/search/hotels+near+me/@34.0635363,-118.4455592,15z

Query: Hotels near me, what is my location (Where to find hotels) & my zoom level (In how much area to search).



Web Page displays data in web browser

https://maps.googleapis.com/maps/api/place/nearbysearch/json

Query: Hotels near 34.0635, -118.445 and Range R and API Key

Results: Hotels are UCLA Tiverton House, ...

Google Places Web API

Google Server

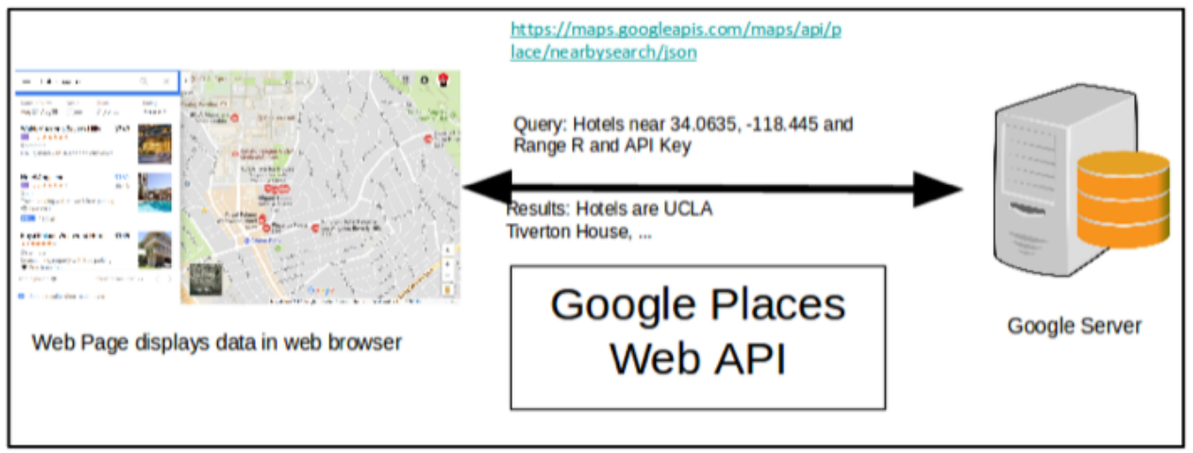
Web API vs Website

Website take care of data presentation to user but web service doesn't do that.

WebPage: https://www.google.com/maps/search/hotels+near+me/@34.0635363,-118.4455592,15z

To See Web API Data:

https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=34.0635363,-118.4455592&radius=1000&type=hotels&keyword=stay&key=%20AIzaSyCA7Ju4jwAoUxDu4GZbCZcwahHdz7OGQfc



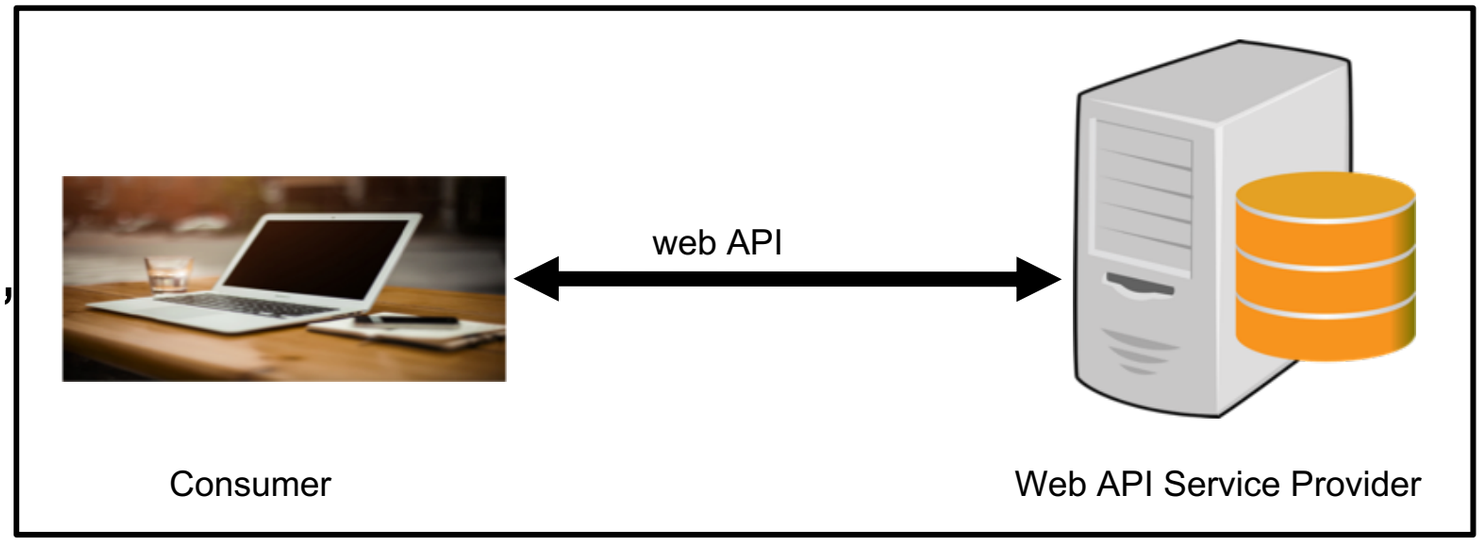

```
▾ results:
  ▾ 0:
    ▾ geometry:
      ▾ location:
          lat:                34.063341
          lng:                -118.4423284
      ▾ viewport:
        ▾ northeast:
            lat:              34.06470823029151
            lng:              -118.4410860197085
        ▾ southwest:
            lat:              34.06201026970851
            lng:              -118.4437839802915
    ▾ icon:                   "https://maps.gstatic.com/mapfiles/place_api/icons/lodging-71.png"
      id:                     "3ce6d9687a73ce07b7f12dd3b348938e20358a00"
      name:                   "UCLA Tiverton House"
    ▾ opening_hours:
        open_now:             true
        weekday_text:
    ▾ photos:
      ▾ 0:
          height:             1520
        ▾ html_attributions:
          ▾ 0:                "<a href=\"https://maps.google.com/maps/contrib/108099629524485009937/photos\">John Damiano</a>"
        ▾ photo_reference:    "CmRaAAAAYfwiSI4ZqjvY7NWMpEGNHl3myOJE_n5aDQ3N_JOlpgYY_vnXT9MljUQ7WBJz5LFFoEmpjnZCPpRl3xUCStP7LbwtA2qNOORkRYgt0IZh7J4d10fUtWc_MP6ZSKhtB
                              hhdtiGw1S4YfBg"
          width:              2688
      place_id:               "ChIJQZfkYIG8woAR6OptJYSDTJg"
```

Defining Web API

Simple terms: *Web API* provides ways to use computing (Query data, Store data, perform calculations etc) facilities over the internet. A Computer/Cluster/Cloud away from user, may provide different functionalities to user by offering a *Web API.* It is a concept not a technology.

This functionality may be:

- Saving data.
- Running computations (eg. query, transformations, calculations).
- Returning results (data).
- or multiple of above.

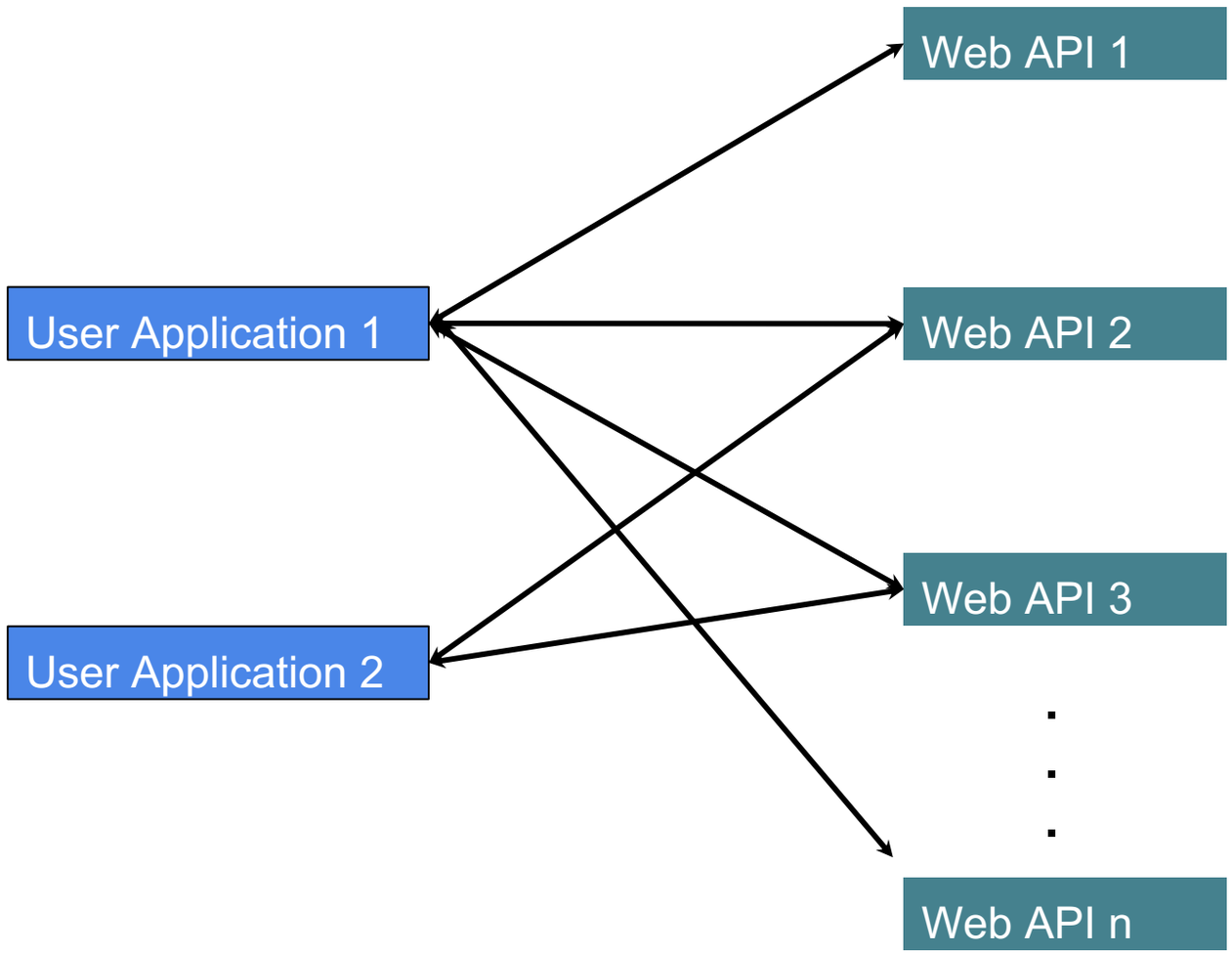


Definition:

Web API is an application programming interface for a web server. Web API doesn't include web server implementation details.

Benefits of Web API

- Loosely Coupled
- Ease of Integration
- Service Reuse



Web API History

Web API evolved from traditionally web services . Web Services are complex and have below components with similar functionalities:

Web Services Description Language (WSDL): Describes how to call web service, what are input parameters and what data structure it returns.

Simple Object Access Protocol (SOAP): Used to exchange information in web services.

Web Service Registry: Web service providers publish services to it, and consumers use it to locate them.

Earlier information exchange, data and response was mostly in extensible markup language (XML) format. Today most of the web services functionality is provided using web API which is implemented by the remote server.

Web API is easy to use, design and implement than Web Services

Components of Web API

In order to get functionality we should know:

Required:

1. From where to get service?
   Service url of the API.
   Eg. https://maps.googleapis.com/maps/api/place/nearbysearch/json
2. What is the format of the request?
   Specifying query and input data etc.|
   Eg. location=34.0635363,-118.4455592&radius=1000&type=hotels
3. What is the response format?
   Response can be string, xml or Json
   Eg. Google places API returns result in Json format.

Optional:

1. How to differentiate between different users or restrict service access?
   Allowed users vs not allowed. Authentication. Eg: key
2. Error codes in the request?
   Inform user if anything is missing in the request. Error codes.

Project - 1

Goal: Use the weather api of ***openweathermap.org*** to get the current weather details of UCLA campus.

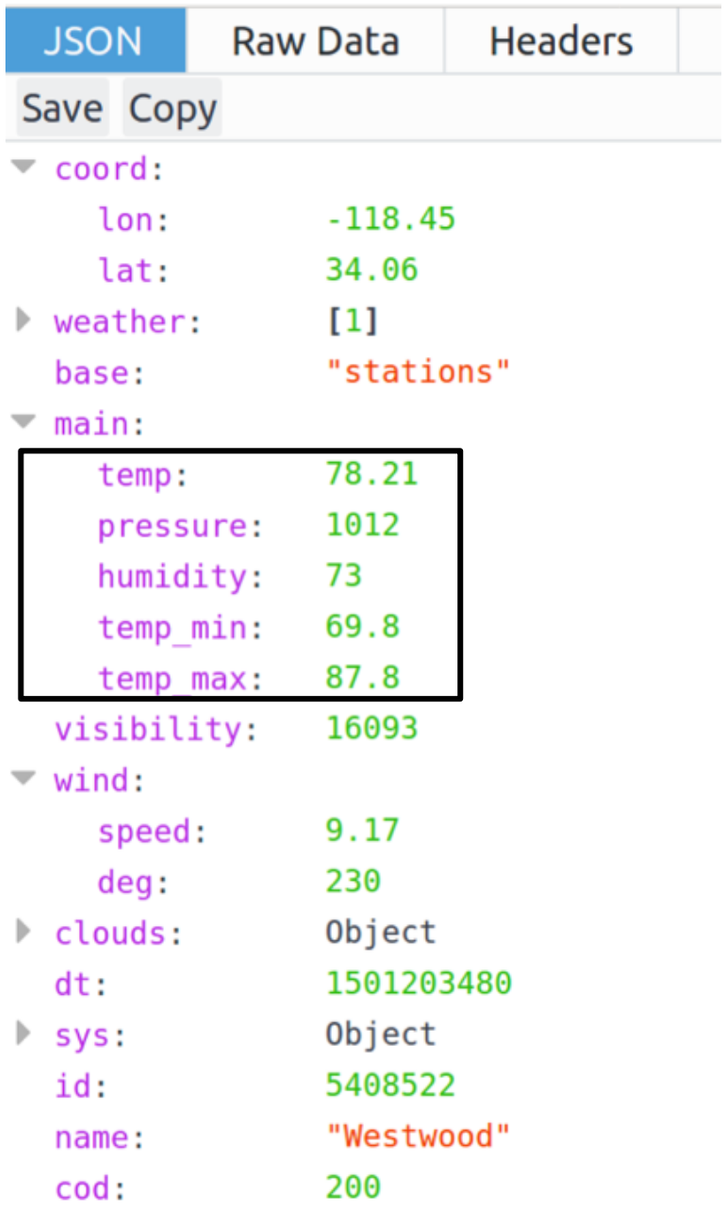
Procedure: Put query url in correct format in the browser and get the results in it.

Steps:

1. Query url: http://api.openweathermap.org/data/2.5/weather
2. Sample query: http://api.openweathermap.org/data/2.5/weather?lat=xxx&lon=xxx&units=Imperial&appid=xxx
3. Put latitude and longitude of a place within UCLA campus.
4. Put appid. (used for authentication)
   use: appid=f3ed13c42e7c876a64fd7841e7da9838

Project 1 Sample output:

Query URL to use:
http://api.openweathermap.org/data/2.5/weather?lat=34.0635363&lon=-118.4455592&units=Imperial&appid=f3ed13c42e7c876a64fd7841e7da9838



Exercise: Using Web API in Python

API Query and receiving data:

You may need to install requests package:

1) Use: sudo pip install requests in OSX/Linux
2) Use easy install in windows.
   Path\easy_install.exe requests
3) Installing easy install in windows:
   http://setuptools.readthedocs.io/en/latest/easy_install.html#installing-easy-install

```
import requests
url="http://api.openweathermap.org/data/2.5/weather?lat=34.0635363&lon=-118.4455592&units=Imperial&appid=f3ed13c42e7c876a64fd7841e7da9838"
response = requests.get(url)
print(response.text)
```

{"coord":{"lon":-118.45,"lat":34.06},"weather":[{"id":801,"main":"Clouds","description":"few clouds","icon":"02d"}],"base":"stations","main":{"temp":79.25,"pressure":1014,"humidity":73,"temp_min":73.4,"temp_max":89.6},"visibility":12874,"wind":{"speed":10.29,"deg":230},"clouds":{"all":20},"dt":1501271880,"sys":{"type":1,"id":490,"message":0.0055,"country":"US","sunrise":1501246989,"sunset":1501297017},"id":5408522,"name":"Westwood","cod":200}

Overview of Lecture

Part-1: Introduction to Web API

- Web API: Simple Example
- Web API vs Website
- Web API in more detail
- Project -1

Part 1 Summary: Web API's are part of larger ecosystem of web development. We will try to try to touch its basics and introduce the Web API's.

Part-2: Introduction to Databases

- Data
- Databases
- Databases: types
- MySql
- MongoDB
- Exercises

Part 2 Summary: Introduce data and ways to store and query it using databases.

Part-3: Project using Web API and MongoDB.

What is Data?

Data: A piece of information.
Anything on which operations can be performed by computer, can be stored and transmitted.

Data is least abstract. Information is next and knowledge is most abstract.

Data is Future? World is driven by data..?

Types of Data :

At higher level data is of following three types depending on its structure:

1. Structured data: Expressed using Tables.
2. Semi-structured data: Expressed using XML or Json.
3. Un-structured data: Expressed as plain text

Other types: quantitative data vs quantitative data

Which type of data is easy to use ?

which type of data exist in abundance ?

```
<dataitem>
    <city>"Los Angeles"</city>
    <name>"Sandeep"</name>
    <id>20</id>
</dataitem>
```

```
{
            "City": "Los
Angeles",
            "Name":
"Sandeep",
```

5	23	2	0.913	0.0588	0.8049	1
8	21	2	0.8261	0.079	0.6848	0.996
9	19	1	0.7826	0.086	0.631	0.971
12	18	1	0.7391	0.0916	0.5798	0.942
13	17	1	0.6957	0.0959	0.5309	0.912
18	14	1	0.646	0.1011	0.4753	0.878
23	13	2	0.5466	0.1073	0.3721	0.803
27	11	1	0.4969	0.1084	0.324	0.762
30	9	1	0.4417	0.1095	0.2717	0.718
31	8	1	0.3865	0.1089	0.2225	0.671
33	7	1	0.3313	0.1064	0.1765	0.622
34	6	1	0.2761	0.102	0.1338	0.569
43	5	1	0.2208	0.0954	0.0947	0.515
45	4	1	0.1656	0.086	0.0598	0.458
48	2	1	0.0828	0.0727	0.0148	0.462

Exercise: Parsing Json

JSON (JavaScript Object Notation) is a lightweight data-interchange format.

Example Json Data:

```
{
        "City": "Los
Angeles",
        "Name":
"Sandeep",
```

In python: using json library

```
>>> import json
>>> print(json.dumps({'Name': 'Sandeep', 'City': "Los Angeles",'id':20}, sort_keys=True, indent=4))
{
    "City": "Los Angeles",
    "Name": "Sandeep",
    "id": 20
}
```

Parsing Json Data:

```
import json
data = '{"Name" : "Sandeep", "City" : "Los Angeles", "id" : 20}'
j = json.loads(data)
print(j['Name'])
Sandeep
```

Database

A Database is structured collection of data.

Example: Telephone directory, Dictionary and many more.

Databases can be stored in computer and analyzed by program. Programs are often called *database management systems* or in short *databases*. We call the programs which help us to analyze data as databases too.

Different types of databases to store different type of data.

Normally we have two types of datastores:

1. **Relational database**. (SQL)
2. **Non-Relational database.** (NoSQL)

Relational Databases

Databases whose organization is based on relations. We have tables and data items are inserted in the form of rows.

Also called SQL databases.

Various relational databases: MySQL, Oracle, Microsoft SQL, IBM DB2.

SQL: **Structured query language.** Used to insert, query and update data items.

Most of early data stored in tables. eg. Data of big banks in Oracle database.

SQL: Queries

Creating Table:

```
CREATE TABLE table_name (
            column1 datatype,
            column2 datatype,
            column3 datatype,
   ....
);
```

Inserting Data:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

Query Data:

```
SELECT column1, column2, … FROM
table_name WHERE condition;
```

Database language, similar to python, but designed to work with database. Can do many complex things, which we left for simplicity.

NoSQL Databases: MongoDB

NoSQL: Used to Store data which is not in tabular form and is semi-structured.

Eg: Json data.

Also called not only SQL.

Today, used increasingly to store big data in web applications.

MongoDB: MongoDB is a **document database** with the scalability and flexibility that you want with the querying and indexing that you need.

Json data items are stored in documents. Documents are similar to tables. Every Json data item is equivalent to row.

Exercise 2: Using MongoDB in Python

1. Start MongoDB service on your machine. (First install MongoDB)
2. Python: Connect to MongoDB and insert few data items.
3. Query

Demo 1:

```
import pymongo
from pymongo import MongoClient
import json


client = MongoClient()
db = client.test_database
collection = db.test_collection


data = '{"Name" : "Sandeep", "City" : "Los Angeles", "id" : 20}'
j = json.loads(data)
data_id = collection.insert_one(j).inserted_id
collection.find_one()
```

Exercise 2: Using MongoDB in Python

Demo 2:

```
#get data
import requests
url="https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=34.0635363,-118.4455592&radius=2000&type=hotels&keyword=stay&key=%20AIzaSyCA7Ju4jwAoUxDu4GZbCZcwahHdz7OGQfc"
response = requests.get(url)
print(response.text)

#parse json
import json
rawdata=response.text
rawjson=json.loads(rawdata)
data=rawjson['results']

#insert into MongoDB
import pymongo
from pymongo import MongoClient
client = MongoClient()
db = client.test_database
collection = db.test_collection
data_id = collection.insert_many(data).insert

#Print All data
cursor=collection.find()
for doc in cursor:
 print(doc)
```

Exercise 2: Using MongoDB in Python

Demo 3:

```
#Print only location from data
cursor=collection.find()
for doc in cursor:
 print(doc['geometry'])

#Creating Index and Query
 collection.ensure_index([('rating',pymongo.ASCENDING)] )
collection.ensure_index([('rating',pymongo.DESCENDING)] )
cursor=collection.find().sort([("rating",-1)]).limit(1)
for doc in cursor:
 print(doc['name'])
```

Overview of Lecture

Part-1: Introduction to Web API

- Web API: Simple Example
- Web API vs Website
- Web API in more detail
- Project -1

Part 1 Summary: Web API's are part of larger ecosystem of web development. We will try to try to touch its basics and introduce the Web API's.

Part-2: Introduction to Databases

- Data
- Databases
- Databases: types
- MySql
- MongoDB
- Exercises

Part 2 Summary: Introduce data and ways to store and query it using databases.

Part-3: Project using Web API and MongoDB.

Part-3: Project using Web API and MongoDB.

Goal: Given a location in NY as input, find the hotels within 10 KM range along with their weather conditions. Store the data in a MongoDB database.

a) How many hotels did you get within 10 KM range.
b) Query the database to give the hotel with best rating.
c) Which hotel/hotels have the highest temperature.
d) Which hotel/hotels have the minimum temperature.

Procedure: Query the hotels using Google Places API. Find the location of Hotel. Use the location of each hotel to get the weather data using weather API. Store the Data of hotel and its weather in MongoDB. Query the database to report the results.

Steps: Will be explained in lecture.