

Resources / Assignments (/COMP9321/19T1/resources/22490)
 / Assignment 2: Data Service for World Bank Economic Indicators

Assignment 2: Data Service for World Bank Economic Indicators

Data Service for World Bank Economic Indicators

Assignment template : <https://www.cse.unsw.edu.au/~cs9321/19T1/assn/assn2.tar.gz>
 (<https://www.cse.unsw.edu.au/~cs9321/19T1/assn/assn2.tar.gz>)

In this assignment, you are asked to develop a **Flask-Restplus** data service that allows a client to read some publicly available economic indicator data for countries around the world, and allow the consumers to access the data through a REST API.

IMPORTANT : For this assignment , you will be asked to access the given Web content programmatically. Some Web hosts do not allow their web pages to be accessed programmatically, some hosts may block your IP if you access their pages too many times. During the implementation, download a few test pages and access the content locally - try not to perform too many programmatically.

- The source URL: <http://api.worldbank.org/v2/> (<http://api.worldbank.org/v2/>)
- (<http://api.worldbank.org/v2/>) Documentations on API Call Structure:
 (<https://datahelpdesk.worldbank.org/knowledgebase/articles/898581-api-basic-call-structure>)
<https://datahelpdesk.worldbank.org/knowledgebase/articles/898581-api-basic-call-structure>
 (<https://datahelpdesk.worldbank.org/knowledgebase/articles/898581-api-basic-call-structure>)

(<https://datahelpdesk.worldbank.org/knowledgebase/articles/898581-api-basic-call-structure>)

The World Bank indicators API provides 50 year of data over more than 1500 indicators for countries around the world.

- List of indicators can be found at: <http://api.worldbank.org/v2/indicators>
 (<http://api.worldbank.org/v2/indicators>)
- List of countries can be found at: (<http://api.worldbank.org/v2/countries>)
<http://api.worldbank.org/v2/countries> (<http://api.worldbank.org/v2/countries>)

As the documentation shows , you can construct URLs specifying different parameters to acquire necessary data. Use a custom URL to get information about a specific indicator. For example, the data on the GDP of all countries from 2013 to 2018 can be acquired via this URL:

(<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2012:2017>)

<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2013:2018&format=json>

(<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2012:2017&format=json>)

For this assignment we are interested in annual values of a user specified indicator from 2013 to 2018 for one or all countries. The content of the file is quite straightforward. The data service specification will ask you to 'import' this data into a 'data storage'. You should inspect the content of the file carefully and decide on a data model and storage method. You will find that a JSON format is suitable to accessing data and publishing it.

The database we used in this assignment is **SQLite** . You need to determine the way about how to store the data. **We don't care about your output format as long as your data is correct and reasonable arranged.** **Please download the python code template a2.py. Don't change the function signature "create_db"**

which you need to use it to create the database. All the other function, it's flexible. Also, no auto-test will be provided.

Note: You may need some additional key in the record for storage purpose.

Don't send too much requests to the source API as it will ban your IP address, be careful. Please make sure you have a swaggerUI page on you API.

How to create database in SQLite: <http://www.sqlitetutorial.net/sqlite-python/creating-database/>
(<http://www.sqlitetutorial.net/sqlite-python/creating-database/>)

[Specification](#)[Make Submission](#)[Check Submission](#)[Collect Submission](#)

Assignment Specification

The data service should use JSON format to publish its data and implement following operations.

1- Import a collection from the data service

This operation can be considered as an on-demand 'import' operation. The service will download the JSON data for all countries (<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2012:2017&format=json>) respective to the year 2013 to 2018 and identified by the indicator id given by the user and process the content into an internal data format.

Parameters should be given to the endpoint (in the payload) by the user:

- **indicator_id** : an indicator (<http://api.worldbank.org/v2/indicators>) <http://api.worldbank.org/v2/indicators> (<http://api.worldbank.org/v2/indicators>)

After importing the collection, the service should return a response containing at least the following information:

- **location** : the URL with which the imported collection can be retrieved
- **collection_id** : a unique identifier automatically generated
- **creation_time** : the time the collection stored in the database

Example:

HTTP operation: POST /<collections>

Input Payload:

```
{ "indicator_id" : "NY.GDP.MKTP.CD" }
```

Returns: 201 Created

```
{
  "location" : "/<collections>/<collection_id>",
  "collection_id" : "<collection_id>",
  "creation_time": "2019-03-09T12:06:11Z",
  "indicator" : "<indicator>"
}
```

- The return response contains the location of the data entry created as the result of the processing the import. What is shown above is just a suggestion. You can decide what else is appropriate. You do not have to return the actual imported content with POST. Just return the location.
- You should return appropriate responses in case of invalid indicators or any invalid attempts to use the endpoint (e.g. If the input indicator id doesn't exist in the data source, return error 404)
- If an input contains a n indicator that already has been imported before, you should still return the location of the data entry - but with status code 200 OK (instead of 201 Created).
- A POINT TO PONDER ABOUT: An `asynchronous POST`?? If a POST takes too long, you may not want the client to wait. What you would do? You do not need to address this in the assignment.
- The source API has pagination; in order to get all of data you need to send many request to import a single collection; however, you are required to get only first **two pages** instead of all:
<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2013:2018&format=json&page=2>
 (<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2012:2017&format=json&page=2>)
- The data entries inside the collection must be converted as described below:

Data entry conversion:

Here is an example of source data entry as it is in the source API

(<http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?date=2013:2018&format=json>) :

```
{
  "indicator": {
    "id": "NY.GDP.MKTP.CD",
    "value": "GDP (current US$)"
  },
  "country": {
    "id": "1A",
    "value": "Arab World"
  },
  "countryiso3code": "",
  "date": "2016",
  "value": 2500164034395.78,
  "unit": "",
  "obs_status": "",
  "decimal": 0
}
```

However, you do not need to store all of its attributes; instead convert it to a JSON format as below:

```
{
  "country": "Arab World",
  "date": "2016",
  "value": 2500164034395.78
}
```

And as a result a collection should be formatted and stored in the database as follow:

```
{
  "collection_id" : "<collection_id>",
  "indicator": "NY.GDP.MKTP.CD",
  "indicator_value": "GDP (current US$)",
  "creation_time" : "<creation_time>"
  "entries" : [
    { "country": "Arab World", "date": "2016",  "value": 2500164034395.78 },
    { "country": "Australia",  "date": "2013",  "value": 780016444034.00 },
    ...
  ]
}
```

2- Deleting a collection with the data service

This operation deletes an existing collection from the database. The interface should look like as below:

```
HTTP operation: DELETE /<collections>/<collection_id>
```

Returns: 200 OK

```
{
  "message" : "Collection = <collection_id> is removed from the database!"
}
```

3 - Retrieve the list of available collections

This operation retrieves all available collections. The interface should look like as like below:

```
HTTP operation: GET /<collections>
```

Returns: 200 OK

```
[
  {
    "location" : "/<collections>/<collection_id_1>",
    "collection_id" : "collection_id_1",
    "creation_time": "<time>",
    "indicator" : "<indicator>"
  },
  {
    "location" : "/<collections>/<collection_id_2>",
    "collection_id" : "collection_id_2",
    "creation_time": "<time>",
    "indicator" : "<indicator>"
  },
  ...
]
```

4 - Retrieve a collection

This operation retrieves a collection by its ID . The response of this operation will show the imported content from world bank API for all 6 years. That is, the data model that you have designed is visible here inside a JSON entry's content part.

The interface should look like as like below:

```
HTTP operation: GET /<collections>/<collection_id>
```

Returns: 200 OK

```
{
  "collection_id" : "<collection_id>",
  "indicator": "NY.GDP.MKTP.CD",
  "indicator_value": "GDP (current US$)",
  "creation_time" : "<creation_time>"
  "entries" : [
    { "country": "Arab World", "date": "2016", "value": 2500164034395.78 },
    { "country": "Australia", "date": "2016", "value": 780016444034.00 },
    ...
  ]
}
```

5 - Retrieve economic indicator value for given country and a year

The interface should look like as like below:

```
HTTP operation: GET /<collections>/<collection_id>/<year>/<country>
```

Returns: 200 OK

```
{
  "collection_id": <collection_id>,
  "indicator" : "<indicator_id>",
  "country": "<country>",
  "year": "<year>",
  "value": <indicator_value_for_the_country>
}
```

6 - Retrieve top/bottom economic indicator values for a given year

The interface should look like as like below:

```
HTTP operation: GET /<collections>/<collection_id>/<year>?q=<query>
```

Returns: 200 OK

```
{
  "indicator": "NY.GDP.MKTP.CD",
  "indicator_value": "GDP (current US$)",
  "entries" : [
    {
      "country": "Arab World",
      "date": "2016",
      "value": 2500164034395.78
    },
    ...
  ]
}
```

The <query> is an optional parameter which can be either of following:

- **top<N>** : Return top N countries sorted by indicator value
- **bottom<N>** : Return bottom N countries sorted by indicator value

where N can be an integer value between 1 and 100. For example, a request like " GET /<collections>/<id>/2015?query=top10 " should returns top 10 countries according to the collection_id.

Notes:

- **REASON** : the source API will ban your (and tutors') IP addresses if you send too many requests
The source API has pagination; in order to get all of data you need to send many request to import a single collection; however, you are **required** to get only first **two pages** instead of all:
http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?
date=2013:2018&format=json&page=1
(http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?
date=2012:2017&format=json&page=1)
http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?
date=2013:2018&format=json&page=2
(http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?
date=2012:2017&format=json&page=2) or to get all data use :
http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?
date=2013:2018&format=json&per_page=2000
(http://api.worldbank.org/v2/countries/all/indicators/NY.GDP.MKTP.CD?
date=2012:2017&format=json&per_page=2000)
- In Q6: top1 should return the highest numeric value
- Your code must implemented in flask-restplus and provide swagger doc for testing the endpoints.

Submission:


give cs9321 assn2 a2.py

Plagiarism:

this will result in a final 0 mark for COMP9321 2019 Term 1.

Resource created 2 days ago (Wednesday 06 March 2019, 02:40:15 PM), last modified about 3 hours ago (Friday 08 March 2019, 12:16:38 PM).

Comments

 (/COMP9321/19T1/forums/search?forum_choice=resource/24864)

 (/COMP9321/19T1/forums/resource/24864)

 Add a comment

There are no comments yet.