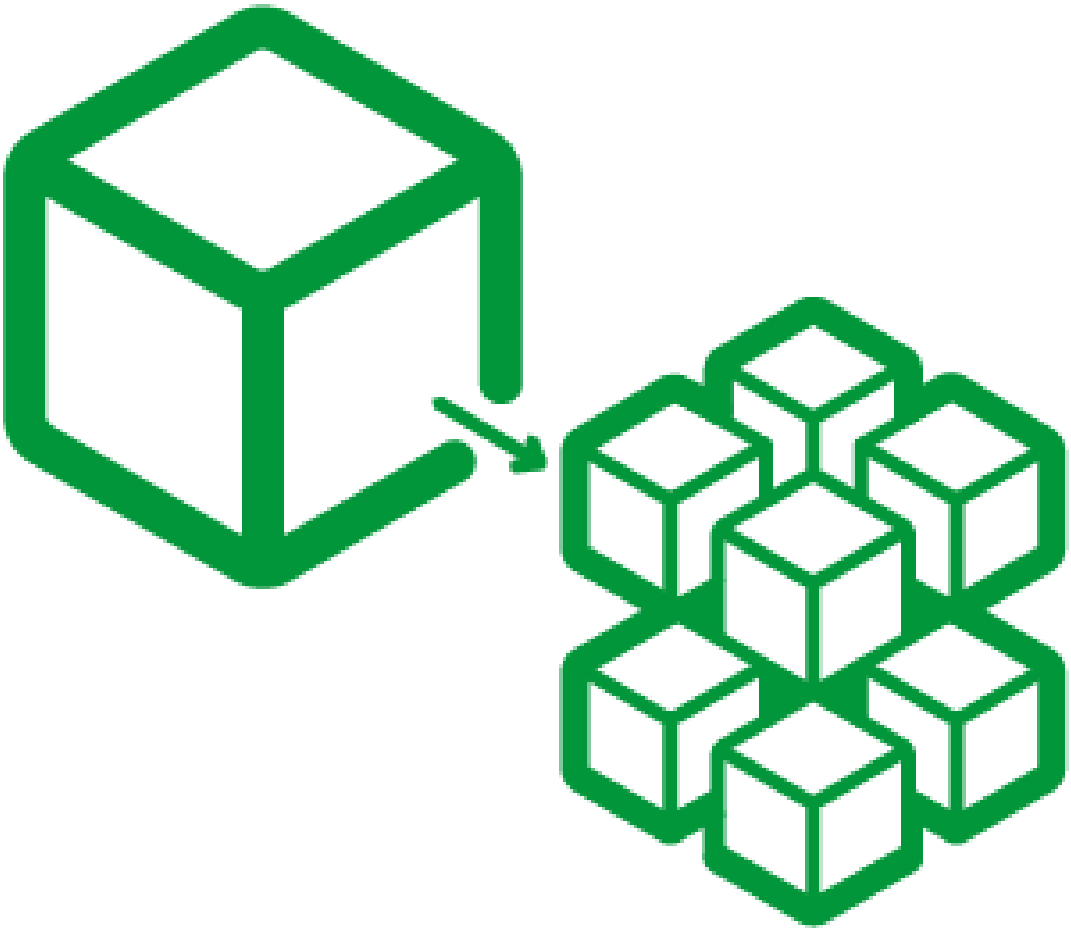


M7024E Laboratory 4:

PROGRAMMING CLOUD SERVICES - RESTFUL APIS

Ahmed Afif Monrat, Syed Asif Iqbal | M7024E: Cloud Services | 1/20/2018

Question 1: What are microservices? Describe in detail the pros and cons of microservices architecture by giving examples.

Answer: The microservice structural style is a way to deal with building up a single application as a suite of small services, each running in its own procedure and communicating with lightweight components, frequently a HTTP resource API. These services are worked around business capacities and freely deployable by completely automated deployment machines. Microservices came to help understand the disappointments of developers were having with large applications that require change cycles to be tied together. In a monolithic application, any little change or refresh required building and deploying a completely new application. This unavoidably implies any application advancement involves a specific measure of arranging, readiness, time and, conceivably, cash.

Microservices, again, is an approach of breaking big software projects into smaller, independent, and loosely coupled modules. It needs little centralized management. The applications are independently deployable and scalable. They enable business capabilities with less planning and production than monoliths.

This modular architectural style is particularly well suited to cloud-based environments and its popularity seems to be rising. Individual modules are responsible for highly defined and discrete tasks and communicate with other modules through simple, universally accessible APIs.

- As opposed to more monolithic design architecture, microservices-
- Large applications can remain largely unaffected by the failure of a single module.
- Dependency concerns is lighter than with monolithic designs, and rolling back changes much easier.
- Makes it easier for a new developer to understand the functionality of a service.

But microservices have cons as well:

- Developing distributed systems can be complex. As everything is now an independent service, developers need to carefully handle requests traveling between different modules. There can be a scenario where one of the services may not be responding, forcing to write extra code specifically to avoid disruption. Therefore, things can get more complicated when latency is experienced.
- Multiple databases and transaction management can be difficult to manage.
- Testing a microservices-based application can be cumbersome. Using the monolithic approach, developers just need to launch the WAR on an application server and ensure its connectivity with the underlying database. But now, each dependent service needs to be confirmed before you can start testing.

- Deploying microservices can be complex. They may need coordination among multiple services, which may not be as straightforward as deploying a WAR in a container.

This architectural style is especially appropriate to cloud-based situations and its ubiquity is by all accounts rising. Individual modules are responsible for highly defined and discrete tasks and communicate with other modules through simple, all around open APIs.

Instead of monolithic design, if microservices can be used then-

- Large applications can remain unaffected by the failure of a single module.
- Dependency concerns is lighter than with monolithic design, and moving back changes substantially is less demanding.
- Makes it less demanding for another engineer to comprehend the usefulness of a service.

But, microservices have cons too:

- Developing distributed systems can be difficult. As everything is currently an independent service, engineers need to precisely deal with demands going between various modules. There can be situations one of the services may not react, driving to compose additional code particularly to stay away from disturbance. In this way, things can get more confusing when latency is experienced.
- Multiple databases and exchange administration can be hard to oversee.
- Testing a microservices-based application can be cumbersome. Utilizing the monolithic approach, engineers simply need to dispatch the WAR on an application server and guarantee its availability with the hidden database. But, in microservices, every service should be affirmed before one can begin testing.
- Deploying microservices can be difficult. They may require coordination among various services, which may not be as direct as deploying a WAR in a container.

Exercise 3(a). Create a RESTful API to implement your service.

Introduction of the Service:

Here I will describe a RESTful webservice application that my team (Asif, Tawseef, Farniba) has developed for Lab 4 in cloud computing course. We developed an API related to most attractive tourist places in a City. Our API will only provide three services:

(i)For a given city anywhere in the world, it will list the name, address and type of popular places in the that city.

(ii)It will recommend the tourist some famous places to visit based on the date they are planning to travel and the weather condition for example, whether its snow, rainy or sunny.

(iii)Finally, it will also give some suggestions for accommodation and find out the city names based on country using third party API.

Framework:

We have used PHP as our language and CodeIgniter framework to build a three-tier service.

Database:

MySQL was choice of our database, that we had previous experience in building application. MySQL is very secure in terms of data security and has firewall to protect SQL injection. The user-friendly IDE is easy to install, and the authorization and privacy management is impressive.

The Database consists of two tables, one with the information of city and countries, other with the information of places in each city.

1286	Maroua	Cameroon
1287	Yaounde	Cameroon
1288	Selkirk	Canada
1289	Berens River	Canada
1290	Pukatawagan	Canada

placeId	placeName	placeAddress	cityId	visitorMark
1	Selkirk Golf & Country Club	100 Sutherland Ave, Selkirk, MB R1A 0L9, Canada	1288	5
2	The Marine Museum of Manitoba	490 Eveline St, Selkirk, MB R1A 2B1, Canada	1288	10
3	Kountry Kids Jam Center Inc.	1E5, 511 Robinson Ave, Selkirk, MB R1A, Canada	1288	*NULL*
4	Sir Walter Scott's Courtroom	26 Market Pl, Selkirk TD7, United Kingdom	1288	*NULL*

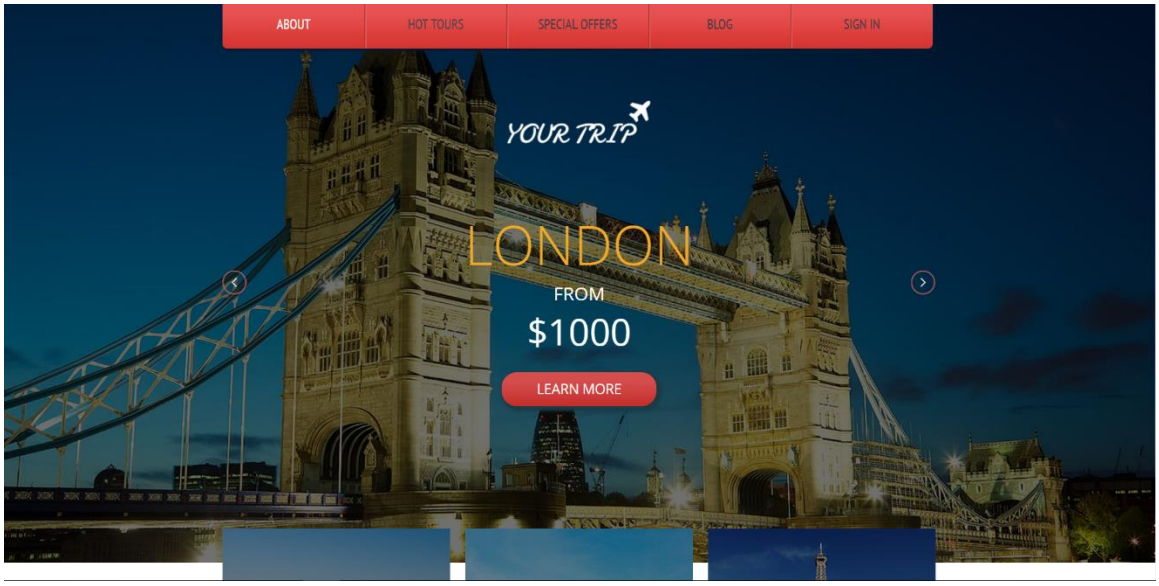
Figure: Database Table structure.

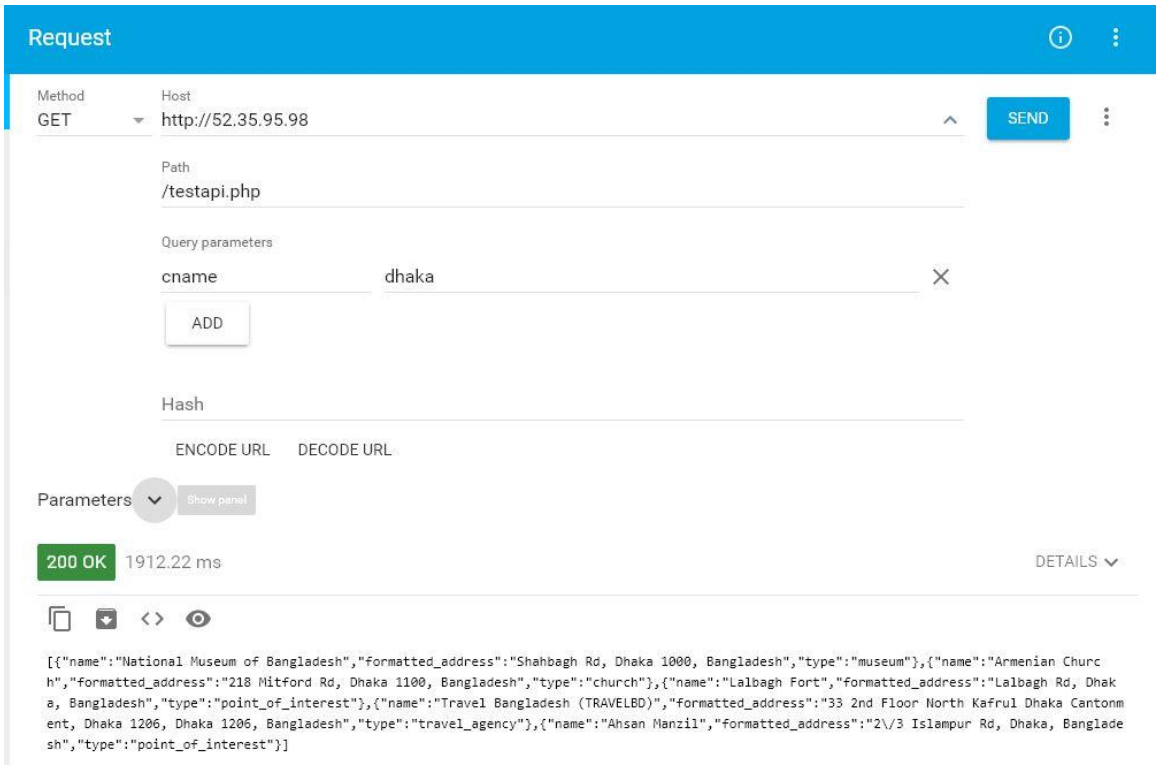
Structure:

The service has basic three structures. One file to GET the data from the database according to request parameter. Second file is to POST data, to add to new popular places to a specific city. The third file is the logic file which manages the connection between the POST and GET method with the database.

Result:

In the following picture, we can see that our service has successfully replied to the get request from an Advance RESTful Client, a google chrome extension. The get parameter is the cityname, 'Dhaka' and the reply is the most visited places in Dhaka in json format. This can be easily used by any other application.





References:

[1]"What is microservices? - Definition from WhatIs.com", SearchMicroservices, 2017. [Online]. Available: http://searchmicroservices.techtarget.com/definition/microservices. [Accessed: 02- Dec- 2017].

[2]V. Badola, "Microservices architecture: advantages and drawbacks", Cloud Academy Blog, 2017. [Online]. Available: https://cloudacademy.com/blog/microservices-architecture-challenge-advantage-drawback/. [Accessed: 02- Dec- 2017].