SAD-A3 Report

Title: KalolLie

https://github.com/akraradets/SAD-A3

By these developers
st120224 Akraradet Sinsamersuk (Microservices)
st120534 Rangtiwa Fordswngnoen (UX developer)
st120695 Thanakit Phanthawit (Front-end + Database)

About KaloLie

Do you have a difficulty remembering all the calorie of your favorite plates?
Introducing KaloLie, the platform that helps you keep all your favorite plates, its ingredients and its calorie in one place. Use it now and have fun!!. (*≧▽≦) ≧(´▽｀)≦

The services

1. Our Microservices

We implement 1 microservice with 2 functions. The first function is to calculate the calorie and the other is acting like a proxy for calling google map API.

The reason why we make this as a microservice is this calculation could be power hungry algorithm, so we separate this function away from the other service. This results in more scalability and more robustness of our system.

```
Services = "https://sad-assignment3.appspot.com/"
```

2. 3rd party Microservices

We use google map API and call to 2 functions. The first function is finding a place from a plain text, but this function did not return phone number, so we need to use another function for query phone number.

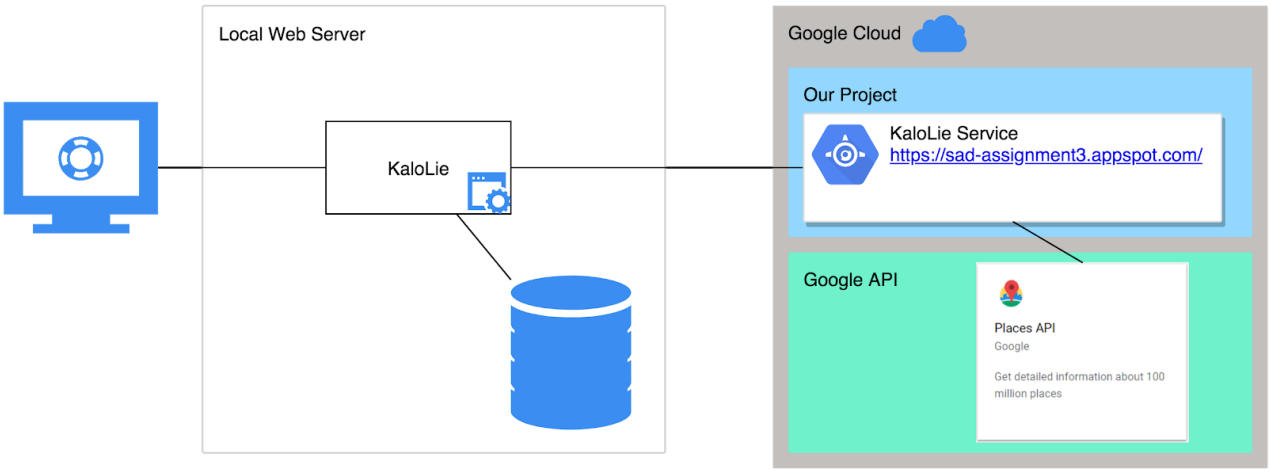
We cannot build our own map and we should not, so we use google map API to provide search feature for our client. (They might feel hungry when they use our application)

```
Findplacefromtext = "https://maps.googleapis.com/maps/api/place/findplacefromtext/json";
Details = "https://maps.googleapis.com/maps/api/place/details/json";
```

Serverless

We use **google cloud/App Engine/services** as our host for our microservice so that we can deploy our microservice without worrying what kind of server our service is running on.

App Architecture



Our Implementation

We use Java and MySQL for our front-end web application.
On each plate, we keep its ingredient list and when we want to know the calorie of the plate, we send the list of ingredient to the cloud (microservice) and it will answer the total calories.

Java

```
//JAVA code for calling MicroService API
HttpPost request = new HttpPost("https://sad-assignment3.appspot.com/");
StringEntity params = new StringEntity(jsArray.toString());
request.addHeader("content-type", "application/json");
request.setEntity(params);
HttpResponse response = httpClient.execute(request);
HttpEntity responseEntity = response.getEntity();
String str = "";
if (responseEntity != null) {
    str = EntityUtils.toString(responseEntity);
}
Gson gson = new Gson();
JsonParser parser = new JsonParser();
JsonObject object = (JsonObject) parser.parse(str);
ResponseObject responseObject = gson.fromJson(object, ResponseObject.class);
model.addAttribute("cal", responseObject.getCal());
httpClient.close();
```

API Request

POST: https://sad-assignment3.appspot.com/
[{"code":"001","unit":"3"},{"code":"009","unit":"4"}]

API Response

{ "code": "0","msg": "","cal": 3900}

Our service is implemented in PHP+Lumen Framework.

PHP+Lumen | calculate Calorie function

```php
// PHP cal microservices
public function cal(Request $request)
{
    $returnArray = array(
        "code" => "1",
        "msg" => "",
        "cal" => ""
    );
    $inputArray = $request->all();
    $cal = 0;
    foreach ($inputArray as $ingred) {
        if(empty($ingred["code"])){
            $returnArray["msg"] = "<code> is empty";
            return $returnArray;
        }
        if(empty($ingred["unit"])){
            $returnArray["msg"] = "<unit> is empty";
            return $returnArray;
        }
        $cal = $cal + (strrev($ingred["code"]) * $ingred["unit"]);
    }
    $returnArray["code"] = "0";
    $returnArray["cal"] = $cal;
    return $returnArray;
}
```

The other service, we use **javascript/Ajax** to call from the browser to our service.

```javascript
function search(){
      // Use modal for fancy
      $("#modal_search").modal('show');
      var input_search = $("#input_search").val();
      var url = "https://sad-assignment3.appspot.com/getPlace";
      // Ajax GET method
      $.get(url, {input: input_search})
      .done(function (res) {
        console.log(res);
        display(res);
      })
      .fail(function (res) {
        console.log(res);
        alert("error");
      });
    }
```

Then our microservice will act as a proxy to query the needed data from **google map API**

```php
    public function getPlace(Request $request){
        $returnArray = array(
            "code" => "1",
            "msg" => "",
            "data" => array()
        );
```

```
        $data = array(
            "id" => "",
            "name" => "",
            "address" => "",
            "url" => "",
            "phoneNo" => ""
        );
        $input = $request->input("input");
        // Get place information from plaintext
        $url = "https://maps.googleapis.com/maps/api/place/findplacefromtext/json";
        $inputArray = array(
            "input"=>$input,
            "inputtype"=>"textquery",
            "fields"=>"place_id,formatted_address,name",
            "key"=>"AIzaSyDzFp4-2Tooi0kXbG2ap0UnkELyRK2DeoQ"
        );
        $result = json_decode($this->callAPI("GET",$url,$inputArray),true);
        if($result["status"] == "ZERO_RESULTS" ||  count($result["candidates"]) == 0 ){
            $returnArray["msg"] = "Place not found";
            return $returnArray;
        }
        $candidate = $result["candidates"][0];
        $data["id"] = $candidate["place_id"];
        $data["name"] = $candidate["name"];
        $data["address"] = $candidate["formatted_address"];
        // Get phone number using placeId
        $url = "https://maps.googleapis.com/maps/api/place/details/json";
        $inputArray = array(
            "placeid" => $data["id"],
            "fields"=>"url,formatted_phone_number",
            "key"=>"AIzaSyDzFp4-2Tooi0kXbG2ap0UnkELyRK2DeoQ"
        );
        $result = json_decode($this->callAPI("GET",$url,$inputArray),true);
        if($result["status"] == "OK"){
            $data["url"] = $result["result"]["url"];
            $data["phoneNo"] = $result["result"]["formatted_phone_number"];
        }
        $returnArray["code"] = 0;
        $returnArray["data"] = $data;
        return $returnArray;
    }
```

Discuss the pros/cons of microservice and serverless technology in your own words

Microservices

Pro :

- Separate the entire backend into different services which will enable the followings
    - Flexibility in term of languages use to implement.
    - No single point of failure (but only if the down service is not important).
- Scaling. If one of the service is really popular, we can scale that particular one, not the entire backend system.

Con :

- If we do not design error handling welled, we will have a bad time find out where is the cause of the error.

Serverless

Pro :

- Everything can be deploy regarding of the hardware knowledge.
- Hardware maintenance is out of the scope of the project life cycle.
- It is probably more robust than your own hardware design.
- It will take less time to launch the product because of there is no infrastructure to design and implement.

Con :

- If your data is confidential, can you trust the provider that they won't take a look at your data.
- It might be more expensive in the long run.
- Although, google cloud provide a lot of control and customization for us (it really is a lot!!!), but we are locked down to only what they support.

Google Cloud

Pro :

- A ton of customization and feature.
- Before you pass the learning curve, you will be overwhelmed by those ton of feature.

Con :

- Learning curve is very high.