Git-hub link : https://github.com/ashraf-jp/NNLIFE/
=========

problem 1: FizzBuzz Problem
=========

Solution:
========

```
<?php
function solution ($arr){
    $last_val = $arr[count($arr)-1];
    $flag = 1;
    $output = "";
        foreach($arr as $value){
            if(!is_INT($value)){
                $explod = explode(":",$value);
                if ($last_val % $explod[0] == 0) {
                    $output .= $explod[1];
                    $flag = 0;
                }
                } elseif ($flag == 1) {
                    $output .= $value;
            }
        }

    echo $output;
}

//$arr = array("3:Fizz","5:Buzz",15);
//$arr = array("3:Tsukemen","5:Mazesoba",29);
$arr = array("3:Sweet","5:Bitter","7:Beauty","11:Song",1150);

solution($arr);

?>
```

Test Pattern for Problem 1:
==========================

```
[
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
       1
    ],
    "output": "1"
  },
  {
```

```
  "input": [
    "3:Fizz",
    "5:Buzz",
    2
  ],
  "output": "2"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    3
  ],
  "output": "Fizz"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    4
  ],
  "output": "4"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    5
  ],
  "output": "Buzz"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    6
  ],
  "output": "Fizz"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    7
  ],
  "output": "7"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    8
    ],
    "output": "8"
  },
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
      9
    ],
    "output": "Fizz"
  },
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
      10
    ],
    "output": "Buzz"
  },
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
      11
    ],
    "output": "11"
  },
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
      12
    ],
    "output": "Fizz"
  },
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
      13
    ],
    "output": "13"
  },
  {
    "input": [
      "3:Fizz",
      "5:Buzz",
      14
    ],
    "output": "14"
  },
  {
      "input": [
        "3:Fizz",
        "5:Buzz",
        15
      ],
      "output": "FizzBuzz"
    },
    {
      "input": [
        "3:Fizz",
        "5:Buzz",
        16
      ],
      "output": "16"
    },
    {
      "input": [
        "3:Fizz",
        "5:Buzz",
        17
      ],
      "output": "17"
    },
    {
      "input": [
        "3:Fizz",
        "5:Buzz",
        18
      ],
      "output": "Fizz"
    },
    {
      "input": [
        "3:Fizz",
        "5:Buzz",
        19
      ],
      "output": "19"
    },
    {
      "input": [
        "3:Fizz",
        "5:Buzz",
        20
      ],
      "output": "Buzz"
    },
    {
      "input": [
        "3:Fizz",
        "5:Buzz",
        21
```

```
  ],
  "output": "Fizz"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    22
  ],
  "output": "22"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    23
  ],
  "output": "23"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    24
  ],
  "output": "Fizz"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    25
  ],
  "output": "Buzz"
},
{
  "input": [
    "3:Fizz",
    "5:Buzz",
    26
  ],
  "output": "26"
},
{
  "input": [
    "3:Uden",
    "5:Odon",
    27
  ],
  "output": "Uden"
},
{
      "input": [
        "3:Odon",
        "5:Uden",
        28
      ],
      "output": "28"
    },
    {
      "input": [
        "3:Tsukemen",
        "5:Mazesoba",
        29
      ],
      "output": "29"
    },
    {
      "input": [
        "3:Ikemen",
        "5:Tsukemen",
        30
      ],
      "output": "IkemenTsukemen"
    },
    {
      "input": [
        "2:Test",
        "3:Driven",
        "5:Development",
        55
      ],
      "output": "Development"
    },
    {
      "input": [
        "2:Test",
        "3:Driven",
        "5:Development",
        56
      ],
      "output": "Test"
    },
    {
      "input": [
        "2:Test",
        "3:Driven",
        "5:Development",
        57
      ],
      "output": "Driven"
    },
    {
      "input": [
    "2:Test",
    "3:Driven",
    "5:Development",
    58
  ],
  "output": "Test"
},
{
  "input": [
    "2:Test",
    "3:Driven",
    "5:Development",
    59
  ],
  "output": "59"
},
{
  "input": [
    "2:Test",
    "3:Driven",
    "5:Development",
    60
  ],
  "output": "TestDrivenDevelopment"
},
{
  "input": [
    "2:Test",
    "3:Driven",
    "5:Development",
    61
  ],
  "output": "61"
},
{
  "input": [
    "2:Test",
    "3:Driven",
    "5:Development",
    62
  ],
  "output": "Test"
},
{
  "input": [
    "3:Sweet",
    "5:Bitter",
    "7:Beauty",
    "11:Song",
    1150
  ],
  "output": "Bitter"
```

```
},
{
  "input": [
    "3:Sweet",
    "5:Bitter",
    "7:Beauty",
    "11:Song",
    1151
  ],
  "output": "1151"
},
{
  "input": [
    "3:Sweet",
    "5:Bitter",
    "7:Beauty",
    "11:Song",
    1152
  ],
  "output": "Sweet"
},
{
  "input": [
    "3:Sweet",
    "5:Bitter",
    "7:Beauty",
    "11:Song",
    1153
  ],
  "output": "1153"
},
{
  "input": [
    "3:Sweet",
    "5:Bitter",
    "7:Beauty",
    "11:Song",
    1154
  ],
  "output": "1154"
},
{
  "input": [
    "3:Sweet",
    "5:Bitter",
    "7:Beauty",
    "11:Song",
    1155
  ],
  "output": "SweetBitterBeautySong"
},
{
    "input": [
      "3:Sweet",
      "5:Bitter",
      "7:Beauty",
      "11:Song",
      1156
    ],
    "output": "1156"
  }
]
```

==

End of Problem 1

==

==

Start of problem 2

==

Deploy a RESTful API!
This is a codecheck challenge.
If this is your first time, check out the tutorial :-)

RESTful web services are lightweight, highly scalable, maintainable and are very commonly used to create APIs for web based applications.

This is an open ended challenge, and the contents of your code will be reviewed and evaluated. Do not submit a solution that merely passes the given unit tests. Please read this README thoroughly.

Your Mission
Your mission is to build and deploy A RESTful API server for managing recipes that satisfies the architectural constraints of RESTful web services.

Implementation Details
Overview
Build the following endpoints:
POST /recipes -> Creates a recipe
GET /recipes -> List all recipes
GET /recipes/{id} -> Return one recipe
PATCH /recipes/{id} -> Updates a recipe
DELETE /recipes/{id} -> Deletes a recipe
All responses must be in JSON format.
Any other specification details not explicitly specified
should conform to the architectural constraints of RESTful web services.
A database schema is already prepared for your API server.
Build the database using sql/create.sql.
Implement your app in the source directory.

Deploy your app in a server or service of your choice (Heroku, AWS, Azure etc)

Once you deploy your server, enter the baseUrl to codecheck.yml.

The unit tests will access this domain when evaluating your Implementation.

Run the unit tests to confirm that your implementation meets the requirements.

Open this challenge in the codecheck web editor and click "Run Test".

POST /recipes Endpoint

This will create a new recipe.

Request: POST /recipes

Body fields:

title, making_time, serves, ingredients, cost

all fields are required.

see sql/create.sql for a description of these properties

Response:

Success response format:

```
{
  "message": "Recipe successfully created!",
  "recipe": [
    {
      "title": "トマトスープ",
      "making_time": "15 分",
      "serves": "5 人",
      "ingredients": "玉ねぎ, トマト, スパイス, 水",
      "cost": "450"
    }
  ]
}
```

Failed response format:

```
{
 "message": "Recipe creation failed!",
 "required": "title, making_time, serves, ingredients, cost"
}
```

GET /recipes Endpoint

This will return all recipes in the database.

Request: GET /recipes

Response:

```
  {
    "recipes": [
      {
        "id": 1,
        "title": "チキンカレー",
        "making_time": "45 分",
        "serves": "4 人",
        "ingredients": "玉ねぎ,肉,スパイス",
        "cost": "1000"
      },
      {
        "id": 2,
        "title": "オムライス",
        "making_time": "30分",
        "serves": "2人",
        "ingredients": "玉ねぎ,卵,スパイス,醤油",
        "cost": "700"
      },
      {
        "id": 3,
        "title": "トマトスープ",
        "making_time": "15分",
        "serves": "5人",
        "ingredients": "玉ねぎ, トマト, スパイス, 水",
        "cost": "450"
      }
    ]
  }
```

GET /recipes/{id} Endpoint
This will return the details of a recipe with the specified id.
Request: GET /recipes/1

Response:

```
{
  "message": "Recipe details by id",
  "recipe": [
    {
      "title": "チキンカレー",
      "making_time": "45分",
      "serves": "4人",
      "ingredients": "玉ねぎ,肉,スパイス",
      "cost": "1000"
    }
  ]
}
```

PATCH /recipes/{id} Endpoint
This will update information of a recipe with the specified id and output the updated recipe.
Request: PATCH /recipes/{id}

Body fields:
title, making_time, serves, ingredients, cost
Response:

```
  {
    "message": "Recipe successfully updated!",
    "recipe": [
      {
        "title": "トマトスープレシピ",
        "making_time": "15分",
        "serves": "5人",
        "ingredients": "玉ねぎ, トマト, スパイス, 水",
        "cost": "450"
      }
    ]
  }
```

DELETE /recipes/{id} Endpoint

This will delete a recipe of the specified id

Request: DELETE /recipes/1

Response:

Success:

```
{  "message": "Recipe successfully removed!" }
```

Failure (there is no recipe to delete):

```
{ "message":"No Recipe found" }
```

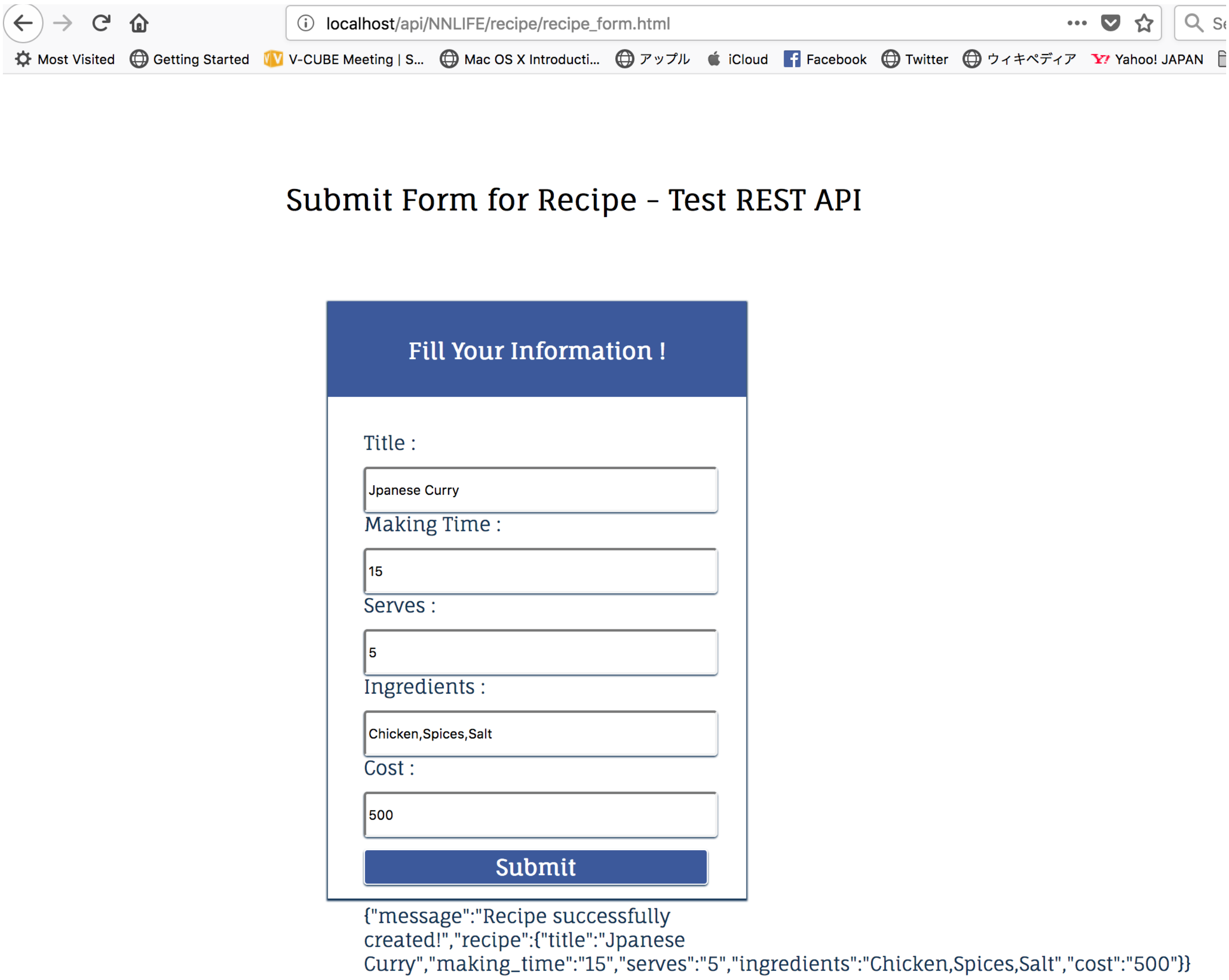
Answer.md

In answer.md write a brief explanation about:

How your code works

Problems faced while solving the challenge

How you solved those problems



Submit Form for Recipe - Test REST API

Fill Your Information !

Title :

Jpanese Curry

Making Time :

15

Serves :

5

Ingredients :

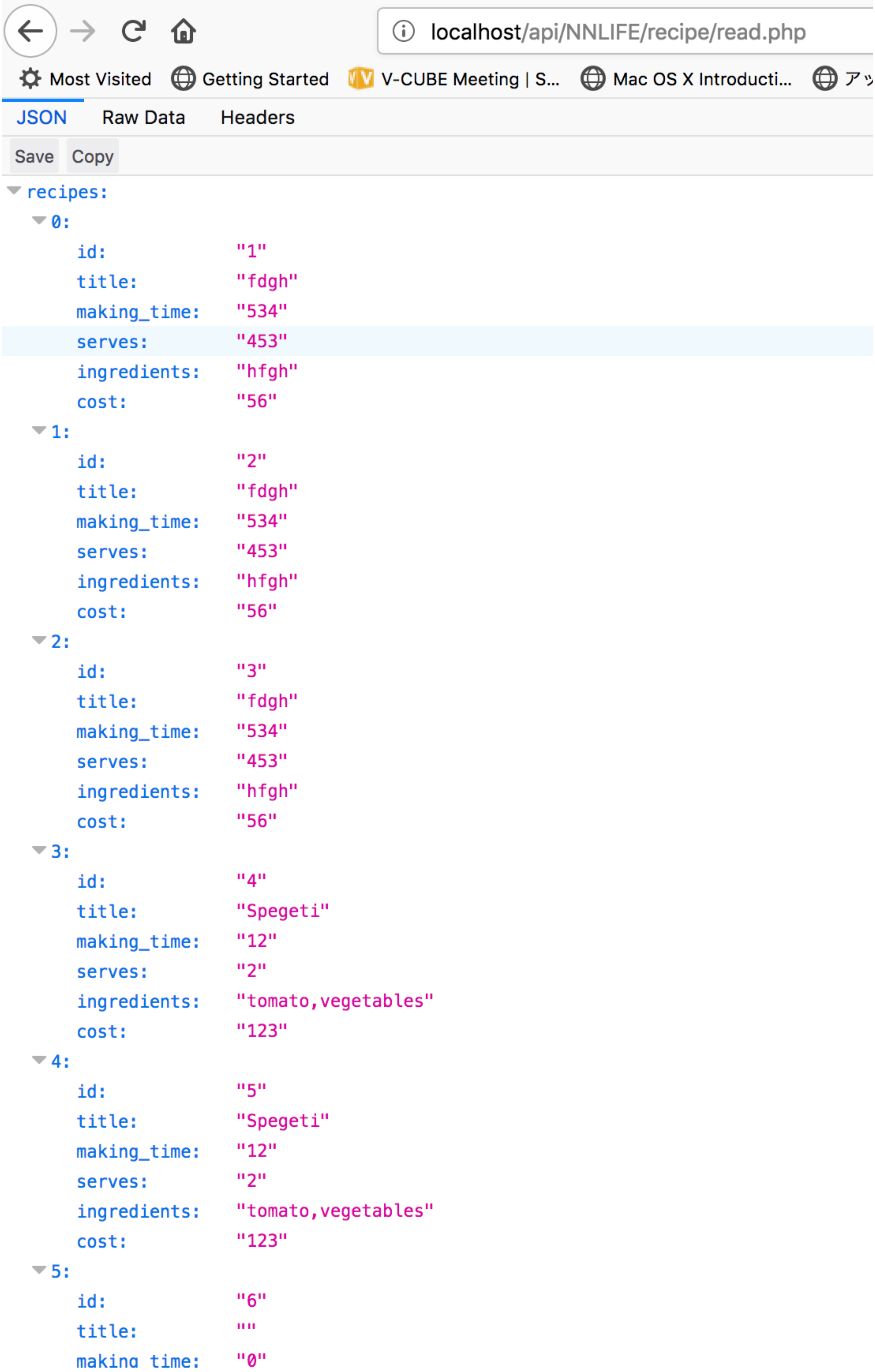
Chicken,Spices,Salt

Cost :

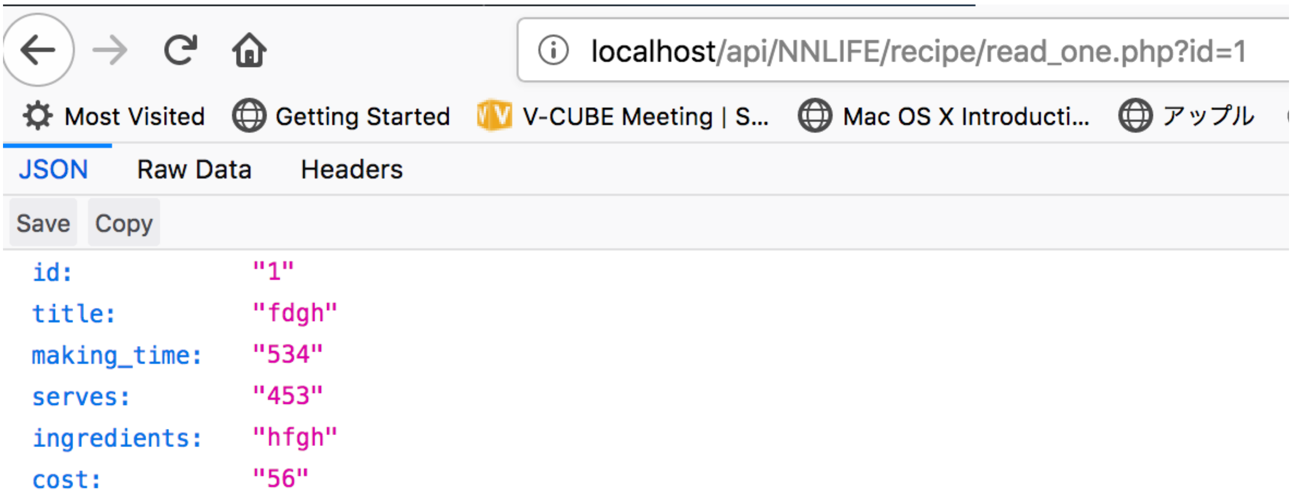
500

Submit

{"message":"Recipe successfully created!","recipe":{"title":"Jpanese Curry","making_time":"15","serves":"5","ingredients":"Chicken,Spices,Salt","cost":"500"}}



```
▼ recipes:
  ▼ 0:
      id:            "1"
      title:         "fdgh"
      making_time:   "534"
      serves:        "453"
      ingredients:   "hfgh"
      cost:          "56"
  ▼ 1:
      id:            "2"
      title:         "fdgh"
      making_time:   "534"
      serves:        "453"
      ingredients:   "hfgh"
      cost:          "56"
  ▼ 2:
      id:            "3"
      title:         "fdgh"
      making_time:   "534"
      serves:        "453"
      ingredients:   "hfgh"
      cost:          "56"
  ▼ 3:
      id:            "4"
      title:         "Spegeti"
      making_time:   "12"
      serves:        "2"
      ingredients:   "tomato,vegetables"
      cost:          "123"
  ▼ 4:
      id:            "5"
      title:         "Spegeti"
      making_time:   "12"
      serves:        "2"
      ingredients:   "tomato,vegetables"
      cost:          "123"
  ▼ 5:
      id:            "6"
      title:         ""
      making_time:   "0"
```



```
id:           "1"
title:        "fdgh"
making_time:  "534"
serves:       "453"
ingredients:  "hfgh"
cost:         "56"
```