SOA & Web Services

Lab Exercises

Thamarai Selvam D

1731054

Ex.No : 1

Question :

Calculate the difference between two dates. (Display the output in No. Of Year, No. Of Month, No. Of Day, No. Of Hour, No. Of Minute, No. Of Seconds)

Solution :

Python :

```
from flask import Flask, request
from flask_restful import Resource, Api, reqparse

app = Flask(__name__)

#
https:://127.0.0.1/datediff?fromYear=2020?fromMonth=06?fromDate=15?fromHour=0
?fromMinute=1?fromSeconds=1?toYear=2020?toMonth=06?toDate=30?toHour=0?toMinut
e=10?toSeconds=1

#Sample UI
#----------------------------------------------------------------
#
# Enter From date             Enter To Date
#   -------------------         --------------------
#   |                 |         |                  |
#   -------------------         --------------------
#
#   ------------
#   | GET DIFF |                OUTPUT____________________
#   ------------                |                         |
#                               |                         |
#                               |_________________________|
#------------------------------------------------------------------
#----------------------------Get,Validate,Set Values------------------------
api = Api(app)

#common error handler
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404



Months = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

class Date:
    def __init__(self, year, month, date, hour, minute, seconds):
        self.date = date
        self.month = month
        self.year = year
        self.hour = hour
        self.minute = minute
        self.seconds = seconds
```

```
def getDifference(fromDate, toDate):

  if (toDate.seconds >= fromDate.seconds):
    rSeconds = toDate.seconds - fromDate.seconds
  else:
    rSeconds = (toDate.seconds + 60) - fromDate.seconds
    toDate.minute -= 1

  if (toDate.minute >= fromDate.minute):
    rMinutes = toDate.minute - fromDate.minute
  else:
    rMinutes = (toDate.minute + 60) - fromDate.minute
    toDate.hour -= 1

  if (toDate.hour >= fromDate.hour):
    rHours = toDate.hour - fromDate.hour
  else :
    rHours = (toDate.hour + 24) - fromDate.hour
    toDate.date -= 1

  if (toDate.date >= fromDate.date):
    rDays = toDate.date - fromDate.date - 1
  else :
    rDays = (toDate.date + Months[toDate.month]) - fromDate.date - 2
    toDate.month -= 1

  if (toDate.month >= fromDate.month):
    rMonths = toDate.month - fromDate.month
  else:
    rMonths = (toDate.month + 12) - fromDate.month
    toDate.year -= 1

  if (toDate.year >= fromDate.year):
    rYears = toDate.year - fromDate.year
  else :
    rYears = 0


  return Date(rYears,rMonths, rDays, rHours, rMinutes, rSeconds)


class DateDifference(Resource):
  def get(self):
    print(request.args)
    if 1000 <= int(request.args['fromYear']) <= 9999 and 1000 <=
int(request.args['toYear']) <= 9999:
      if 1 <= int(request.args['fromMonth']) <= 12 and 1 <=
int(request.args['toMonth']) <= 12:
        if 1 <= int(request.args['fromDate']) <= 31 and 1 <=
int(request.args['toDate']) <= 31:
          if 0 <= int(request.args['fromHour']) <= 24 and 0 <=
int(request.args['toHour']) <= 24:
            if 0 <= int(request.args['fromMinute']) <= 31 and 0 <=
int(request.args['toMinute']) <= 31:
```

```
              if 0 <= int(request.args['fromSeconds']) <= 31 and 0 <=
int(request.args['toSeconds']) <= 31:
                fromDate  =
Date(int(request.args['fromYear']),int(request.args['fromMonth']),
int(request.args['fromDate']), int(request.args['fromHour']),
int(request.args['fromMinute']), int(request.args['fromSeconds']))
                toDate    =
Date(int(request.args['toYear']),int(request.args['toMonth']),
int(request.args['toDate']), int(request.args['toHour']),
int(request.args['toMinute']), int(request.args['toSeconds']))
              else : return 'Invalid Seconds !', 200
            else : return 'Invalid Minutes !', 200
          else : return 'Invalid Hour !', 200
        else : return 'Invalid Date !', 200
      else : return 'Invalid Month !', 200
    else : return 'Invalid Year !', 200


    print('from : ', fromDate,'\nto : ',toDate)
    rDate = getDifference(fromDate, toDate)
    print(f"{rDate.year} Year(s) {rDate.month} Month(s) {rDate.date} Day(s)
{rDate.hour} Hour(s) {rDate.minute} Minute(s) {rDate.seconds} Second(s)")
    return home('',str(fromDate.date), str(fromDate.month),
str(fromDate.year),str(fromDate.hour),
                str(fromDate.minute),str(fromDate.seconds),str(toDate.date),
str(toDate.month),str(toDate.year),
                str(toDate.hour),str(toDate.minute),str(toDate.seconds),
                f"{rDate.year} Year(s) {rDate.month} Month(s) {rDate.date}
Day(s) {rDate.hour} Hour(s) {rDate.minute} Minute(s) {rDate.seconds}
Second(s)", True)

api.add_resource(DateDifference, '/datediff')


@app.route('/', methods=['GET','POST'])
def
home(responseLabel='',fromDate='10',fromMonth='01',fromYear='2020',fromHour='
10',fromMinute='13',fromSeconds='31',toDate='12',toMonth='05',toYear='2021',t
oHour='02',toMinute='25',toSeconds='26',value="", isDone=False):
  if (isDone):
    responseLabel = str('Difference b/w
'+fromDate+'/'+fromMonth+'/'+fromYear+' '+fromHour+' : '+fromMinute+' :
'+fromSeconds+' And '+toDate+'/'+toMonth+'/'+toYear+' '+toHour+' :
'+toMinute+' : '+toSeconds+' is '+value)
    print(responseLabel)
  return '''<!DOCTYPE html>
<html>
<head>
<title>Date Difference</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
        color : white;
    font-style: oblique;
}
.info {
        background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
        background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width:15%;
  padding: 12px 10px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
        padding-top:50px;
}
</style>
</head>
<body>
<div class="info">
<h1>Date Difference</h1>
<p>API Usage : Returns the difference b/w two given dates</p>
<p class="api_proc">API Call Format :
<span>http://localhost:5000/datediff?fromYear=2020&fromMonth=07&fromDate=15&f
romHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=09&toDate=30&toHour=
0&toMinute=10&toSeconds=1</span></p>
</div>
```

```
<div class="workarea">


  <form action="/datediff">
    <label for="fromDate">From Date Time</label><br>
    <div id="fromBlock" name="fromBlock">
    <label for="fromYear">Year</label>
    <input type="text" id="fromYear" name="fromYear"
value="'''+fromYear+'''">
    <label for="fromMonth">Month</label>
    <input type="text" id="fromMonth" name="fromMonth"
value="'''+fromMonth+'''">
    <label for="fromDate">Date</label>
    <input type="text" id="fromDate" name="fromDate"
value="'''+fromDate+'''">
    <br>
    <label for="fromHour">Hour</label>
    <input type="text" id="fromHour" name="fromHour"
value="'''+fromHour+'''">
    <label for="fromMinute">Minute</label>
    <input type="text" id="fromMinute" name="fromMinute"
value="'''+fromMinute+'''">
    <label for="fromSeconds">Seconds</label>
    <input type="text" id="fromSeconds" name="fromSeconds"
value="'''+fromSeconds+'''">
    </div><br>
    <label for="toBlock">To Date Time</label><br>
    <div id="toBlock" name="toBlock">

    <label for="toYear">Year</label>
    <input type="text" id="toYear" name="toYear" value="'''+toYear+'''">
    <label for="toMonth">Month</label>
    <input type="text" id="toMonth" name="toMonth" value="'''+toMonth+'''">
    <label for="toDate">Date</label>
    <input type="text" id="toDate" name="toDate" value="'''+toDate+'''">
    <br>
    <label for="toHour">Hour</label>
    <input type="text" id="toHour" name="toHour" value="'''+toHour+'''">
    <label for="toMinute">Minute</label>
    <input type="text" id="toMinute" name="toMinute"
value="'''+toMinute+'''">
    <label for="toSeconds">Seconds</label>
    <input type="text" id="toSeconds" name="toSeconds"
value="'''+toSeconds+'''">
    </div>

    <input type="submit" value="Submit">
  </form>
  <form class="response_form">
   <p>''' +responseLabel+'''</p>
  </form>

  </div>

  </body>
  </html>
  '''

if __name__ == '__main__':
  app.run()
```

JavaScript :

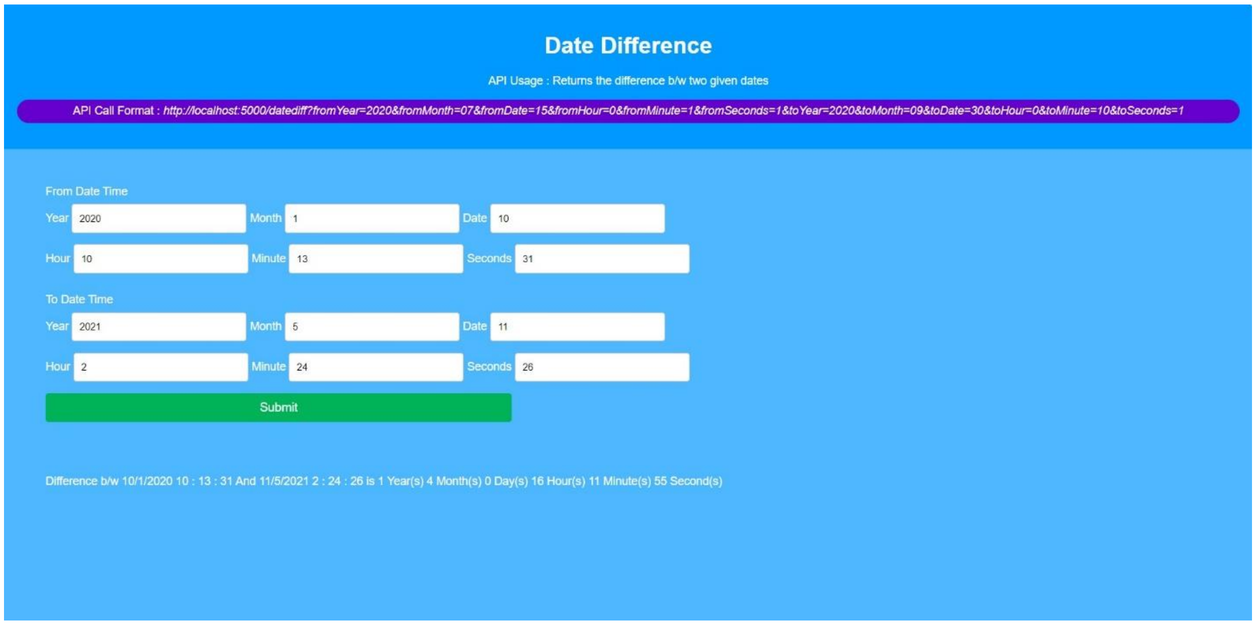
```
const express = require('express')
const app = express()
const bp = require('body-parser')

app.use(bp.text())

app.get('/datediff', (req, res) => {
    var Months = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
    var fromDateSet = []
    var toDateSet = []
    var diffDateSet = []
    var index = []
    console.log(req.query)

    if ((1000 <= req.query.fromYear <= 9999) && (1000 <= req.query.toYear <=
9999))
        if ((1 <= req.query.fromMonth <= 12) && (1 <= req.query.fromMonth <=
12))
            if ((1 <= req.query.fromDate <= 31) && (1 <= req.query.toDate <=
31))
                if ((0 <= req.query.fromHour < 24) && (0 <= req.query.toHour
< 24))
                    if ((0 <= req.query.fromMinute < 60) && (0 <= req.query.t
oMinute < 60))
                        if ((0 <= req.query.fromSeconds < 60) && (0 <= req.qu
ery.toSeconds < 60)) {} else res.send('Invalid Seconds !')
    else res.send('Invalid Minutes !')
    else res.send('Invalid Hours !')
    else res.send('Invalid Date !')
    else res.send('Invalid Month !')
    else res.send('Invalid Year !')

    if (req.query.toSeconds >= req.query.fromSeconds)
        diffDateSet.push(req.query.toSeconds - req.query.fromSeconds)
    else {
        diffDateSet.push((req.query.toSeconds + 60) - req.query.fromSeconds)
        req.query.toMinute -= 1
    }

    if (req.query.toMinute >= req.query.fromMinute)
        diffDateSet.push(req.query.toMinute - req.query.fromMinute)
    else {
        diffDateSet.push((req.query.toMinute + 60) - req.query.fromMinute)
        req.query.toHour -= 1
    }
    if (req.query.toHour >= req.query.fromHour)
        diffDateSet.push(req.query.toHour - req.query.fromHour)
    else {
        diffDateSet.push((req.query.toHour + 24) - req.query.fromHour)
        req.query.toDate -= 1
    }

    if (req.query.toDate >= req.query.fromDate)
        diffDateSet.push(req.query.toDate - req.query.fromDate)
    else {
        diffDateSet.push((req.query.toDate + Months[req.query.toMonth]) - req
.query.fromDate)
        req.query.toMonth -= 1
    }

    if (req.query.toMonth >= req.query.fromMonth)
        diffDateSet.push(req.query.toMonth - req.query.fromMonth)
    else {
        diffDateSet.push((req.query.toMonth + 12) - req.query.fromMonth)
        req.query.toYear -= 1
    }

    if (req.query.toYear >= req.query.fromYear)
        diffDateSet.push(req.query.toYear - req.query.fromYear)
    else
        diffDateSet.push(0)

    var resultDiff = diffDateSet[5] + ' Years' + diffDateSet[4] + ' Months' +
 diffDateSet[3] + ' Days' + diffDateSet[2] + ' Hours' + diffDateSet[1] + ' Mi
nutes' + diffDateSet[0] + ' Seconds'



    res.send(resultDiff);
})

//start
app.listen(3000)
```

PHP :

```
<!DOCTYPE html>
<html>
<head>
<title>Date Difference</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
    color : white;
    font-style: oblique;
}
.info {
    background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
    background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.err_proc {
            background-color: red;
            padding: 7px;
            border-radius: 20px;
        }
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width:15%;
  padding: 12px 10px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
    padding-top:50px;
}
```

```
</style>
</head>
<body>
    <?php
     $fromDate = 12; $fromMonth = 10; $fromYear = 2020; $fromHour = 10; $from
Minute = 44; $fromSeconds = 30;
     $toDate = 10; $toMonth = 8; $toYear = 2020; $toHour = 11; $toMinute = 32
; $toSeconds = 21;
     $diff = "";
     if ($_SERVER["REQUEST_METHOD"] == "POST") {
        // collect value of input field
        $fromDate = $_POST['fromDate']; $fromMonth = $_POST['fromMonth']; $fr
omYear = $_POST['fromYear']; $fromHour = $_POST['fromHour']; $fromMinute = $_
POST['fromMinute']; $fromSeconds = $_POST['fromSeconds'];
        $toDate = $_POST['toDate']; $toMonth = $_POST['toMonth']; $toYear = $
_POST['toYear']; $toHour = $_POST['toHour']; $toMinute = $_POST['toMinute'];
$toSeconds = $_POST['toSeconds'];


        $fromPeriod = date_create($fromYear.'-'.$fromMonth.'-
'.$fromDate.' '.$fromHour.':'.$fromMinute.':'.$fromSeconds);
        $toPeriod = date_create($toYear.'-'.$toMonth.'-
'.$toDate.' '.$toHour.':'.$toMinute.':'.$toSeconds);


        // send the result now
        if (empty($fromPeriod) || empty($toPeriod)) {
            echo "<p class='err_proc'>Invalid Date Format</p>";
        } else {
             $diff= date_diff($toPeriod,$fromPeriod);
            // echo date_format($fromPeriod,"Y/m/d H:iP");
            // echo date_format($toPeriod,"Y/m/d H:iP");
             $diff = $diff-
>format('%y Years %m Months %d Days %h Hours %i Minutes %s Seconds');
        }
    }


    ?>
<div class="info">
<h1>Date Difference</h1>
<p>API Usage : Returns the difference b/w two given dates</p>
<p class="api_proc">API Call Format : <span>http://localhost:5000/datediff?fr
omYear=2020&fromMonth=07&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&to
Year=2020&toMonth=09&toDate=30&toHour=0&toMinute=10&toSeconds=1</span></p>
</div>
<div class="workarea">

<form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF'])
;?>">
    <label for="fromDate">From Date Time</label><br>
<div id="fromBlock" name="fromBlock">
    <label for="fromYear">Year</label>
    <input type="text" id="fromYear" name="fromYear" value='<?php echo $fromY
ear?>'>
    <label for="fromMonth">Month</label>
    <input type="text" id="fromMonth" name="fromMonth" value='<?php echo $fro
mMonth?>'>
    <label for="fromDate">Date</label>
    <input type="text" id="fromDate" name="fromDate" value='<?php echo $fromD
ate?>'>
    <br>
    <label for="fromHour">Hour</label>
    <input type="text" id="fromHour" name="fromHour" value='<?php echo $fromH
our?>'>
    <label for="fromMinute">Minute</label>
    <input type="text" id="fromMinute" name="fromMinute" value='<?php echo $f
romMinute?>'>
    <label for="fromSeconds">Seconds</label>
    <input type="text" id="fromSeconds" name="fromSeconds" value='<?php echo
$fromSeconds?>'>
    </div><br>
    <label for="toBlock">To Date Time</label><br>
    <div id="toBlock" name="toBlock">

    <label for="toYear">Year</label>
    <input type="text" id="toYear" name="toYear" value='<?php echo $toYear?>'
>
    <label for="toMonth">Month</label>
    <input type="text" id="toMonth" name="toMonth" value='<?php echo $toMonth
?>'>
    <label for="toDate">Date</label>
    <input type="text" id="toDate" name="toDate" value='<?php echo $toDate?>'
>
    <br>
    <label for="toHour">Hour</label>
    <input type="text" id="toHour" name="toHour" value='<?php echo $toHour?>'
>
    <label for="toMinute">Minute</label>
    <input type="text" id="toMinute" name="toMinute" value='<?php echo $toMin
ute?>'>
    <label for="toSeconds">Seconds</label>
    <input type="text" id="toSeconds" name="toSeconds" value='<?php echo $toS
econds?>'>
    </div>

    <input type="submit" value="Submit">
  </form>
  <form class="response_form">
   <p><?php echo $diff?></p>
  </form>

  </div>

  </body>
  </html>
```

Screens :



Date Difference

API Usage : Returns the difference b/w two given dates

API Call Format : *http://localhost:5000/datediff?fromYear=2020&fromMonth=07&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=09&toDate=30&toHour=0&toMinute=10&toSeconds=1*

From Date Time

Year: 2020 Month: 1 Date: 10

Hour: 10 Minute: 13 Seconds: 31

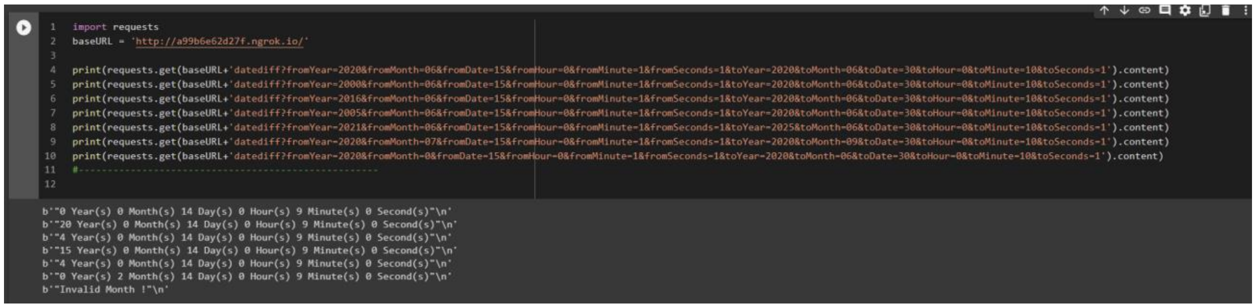
To Date Time

Year: 2021 Month: 5 Date: 11

Hour: 2 Minute: 24 Seconds: 26

Difference b/w 10/1/2020 10 : 13 : 31 And 11/5/2021 2 : 24 : 26 is 1 Year(s) 4 Month(s) 0 Day(s) 16 Hour(s) 11 Minute(s) 55 Second(s)

Test Report :



```
import requests
baseURL = 'http://a99b6e62d27f.ngrok.io/'

print(requests.get(baseURL+'datediff?fromYear=2020&fromMonth=06&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=06&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
print(requests.get(baseURL+'datediff?fromYear=2000&fromMonth=06&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=06&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
print(requests.get(baseURL+'datediff?fromYear=2016&fromMonth=06&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=06&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
print(requests.get(baseURL+'datediff?fromYear=2005&fromMonth=06&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=06&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
print(requests.get(baseURL+'datediff?fromYear=2021&fromMonth=06&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2025&toMonth=06&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
print(requests.get(baseURL+'datediff?fromYear=2020&fromMonth=07&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=09&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
print(requests.get(baseURL+'datediff?fromYear=2020&fromMonth=0&fromDate=15&fromHour=0&fromMinute=1&fromSeconds=1&toYear=2020&toMonth=06&toDate=30&toHour=0&toMinute=10&toSeconds=1').content)
#-----------------------------------------------------

```

```
b'"0 Year(s) 0 Month(s) 14 Day(s) 0 Hour(s) 9 Minute(s) 0 Second(s)"\n'
b'"20 Year(s) 0 Month(s) 14 Day(s) 0 Hour(s) 9 Minute(s) 0 Second(s)"\n'
b'"4 Year(s) 0 Month(s) 14 Day(s) 0 Hour(s) 9 Minute(s) 0 Second(s)"\n'
b'"15 Year(s) 0 Month(s) 14 Day(s) 0 Hour(s) 9 Minute(s) 0 Second(s)"\n'
b'"4 Year(s) 0 Month(s) 14 Day(s) 0 Hour(s) 9 Minute(s) 0 Second(s)"\n'
b'"0 Year(s) 2 Month(s) 14 Day(s) 0 Hour(s) 9 Minute(s) 0 Second(s)"\n'
b'"Invalid Month !"\n'
```

Ex.No : 2

Question :

Perform Set theory operations such as Union, Minus, Intersection for the group of data.

Solution :
Python :

```
from flask import Flask, request
from flask_restful import Resource, Api, reqparse

app = Flask(__name__)


api = Api(app)

setA = set()
setB = set()

#------------------------------------------------
class SetTheory(Resource):
  def get(self):

    print(request.args)
    operation = int(request.args['operation'])
    setA = set(request.args['setA'].split(','))
    setB = set(request.args['setB'].split(','))

    if (operation == 0):
      print('Union', setA | setB)
      # return {'Union' : list(setA | setB)}, 200
      return home(str(operation),"Union", request.args['setA'],
request.args['setB'], str(setA | setB))
    elif (operation == 1):
      print('Intersection', setA & setB)
      return home(str(operation),"Intersection", request.args['setA'],
request.args['setB'], str(setA & setB))
      # return {'Intersection' : list(setA & setB)}, 200
    else:
      print('Minus', setA - setB)
      # return {'Minus' : list(setA - setB)}, 200
      return home(str(operation),"Minus", request.args['setA'],
request.args['setB'], str(setA - setB))



api.add_resource(SetTheory,"/settheory")

@app.route('/setops', methods=['GET'])
def setOps():

  print(request.args)
  operation = int(request.args['operation'])
  setA = set(request.args['setA'].split(','))
  setB = set(request.args['setB'].split(','))

  if (operation == 0):
      print('Union', setA | setB)
      # return {'Union' : list(setA | setB)}, 200
      return home(str(operation),"Union", request.args['setA'],
request.args['setB'], str(setA | setB))
  elif (operation == 1):
      print('Intersection', setA & setB)
      return home(str(operation),"Intersection", request.args['setA'],
request.args['setB'], str(setA & setB))
      # return {'Intersection' : list(setA & setB)}, 200
  else:
      print('Minus', setA - setB)
      # return {'Minus' : list(setA - setB)}, 200
      return home(str(operation),"Minus", request.args['setA'],
request.args['setB'], str(setA - setB))

#common error handler
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404


@app.route('/', methods=['GET','POST'])
def home(op="0",opName="",setA="1,2,3,4,5", setB="4,5,6,7,8",value=""):
  return '''<!DOCTYPE html>
<html>
<head>
<title>Set Theory</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
        color : white;
    font-style: oblique;
}
.info {
        background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
        background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width:150%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
        padding-top:50px;
}
</style>
</head>
<body>
<div class="info">
<h1>Set Theory</h1>
<p>API Usage : Returns the result of given set and operation</p>
<p class="api_proc">API Call Format :
<span>http://localhost:5000/settheory?operation=0&setA=1,2,3,4,4,5,5,5,6,7&se
tB=1,2,3,3,3,4,5,6,7,9,0</span></p>
</div>
<div class="workarea">

<form action="/setops">
  <label for="message">Operation ( 0 - Union, 1 - Intersection, 2 -
Minus)</label><br>
  <input type="text" id="operation" name="operation" value="'''+op+'''"><br>
  <label for="message">Set A ( Comma Seperate Values. Ex=
1,2,3,4,5)</label><br>
  <input type="text" id="setA" name="setA" value="'''+setA+'''"><br>
  <label for="message">Set A ( Comma Seperate Values. Ex=
4,5,6,7,8)</label><br>
  <input type="text" id="setB" name="setB" value="'''+setB+'''"><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">Result of '''+opName+'''</label><br>
  <input type="text" id="response" name="response" value="'''+value+'''"><br>
</form>

</div>
```

```
</body>
</html>

'''

if __name__ == '__main__':
  app.run()
```

JavaScript :

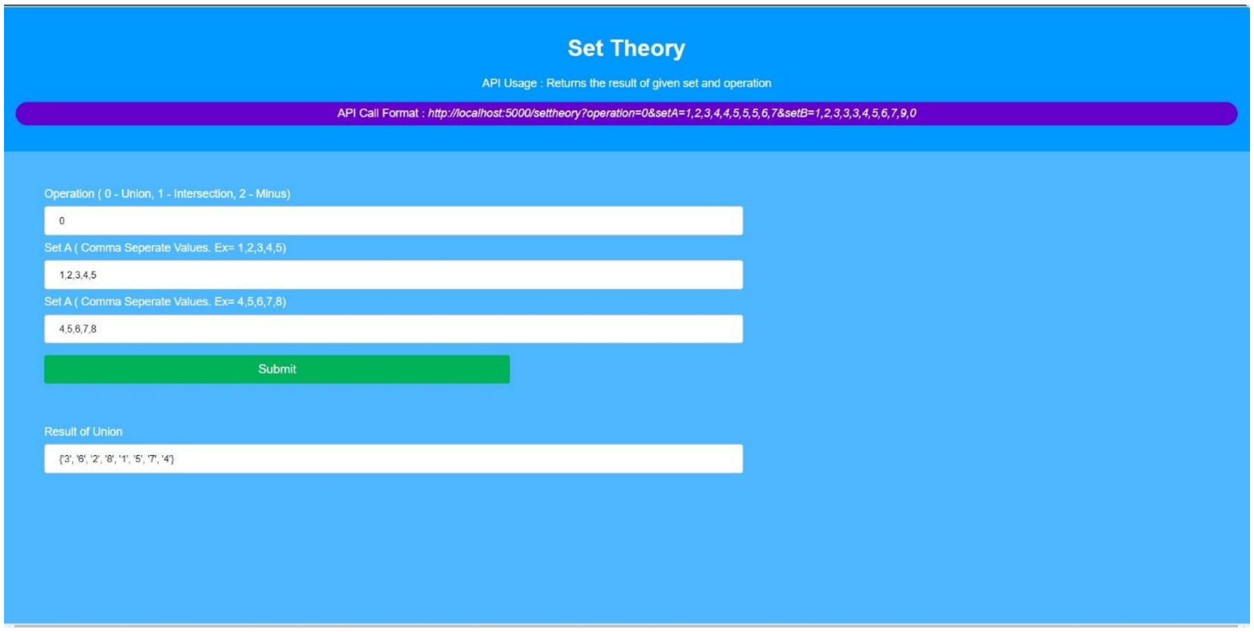
```
const express = require('express')
const app = express()
const bp = require('body-parser')

app.use(bp.text())

Set.prototype.union = function(otherSet) {
    var unionSet = new Set();
    for (var elem of this) {
        unionSet.add(elem);
    }
    for (var elem of otherSet)
        unionSet.add(elem);

    return unionSet;
}

Set.prototype.intersection = function(otherSet) {
    var intersectionSet = new Set();
    for (var elem of otherSet) {
        if (this.has(elem))
            intersectionSet.add(elem);
    }
    return intersectionSet;
}
Set.prototype.minus = function(otherSet) {
    var differenceSet = new Set();
    for (var elem of this) {
        if (!otherSet.has(elem))
            differenceSet.add(elem);
    }
    return differenceSet;
}
app.get('/setops', (req, res) => {

    var operation = req.query.operation
    var setA = new Set(req.query.setA)
    var setB = new Set(req.query.setB)

    console.log(req.query)
    if (operation == 0) {
        console.log(setA.union(setB))
        return res.send({ 'Union': [...setA.union(setB)] });
    } else if (operation == 1) {
        console.log(setA.intersection(setB))
        return res.send({ 'Intersection': [...setA.intersection(setB)] });
    } else {
        console.log(setA.minus(setB))
        return res.send({ 'Minus': [...setA.minus(setB)] });
    }

})

//start
app.listen(3000)
```

PHP :

```
<!DOCTYPE html>
<html><head>
    <title>Set Theory</title>
    <style>
        body {
            background-color: #4db8ff;
            text-align: center;
            color: white;
            font-family: Arial, Helvetica, sans-serif;
        }

        span {
            color: white;
            font-style: oblique;
        }

        .info {
            background-color: #0099ff;
            padding: 20px;
            margin: -10px -10px 0 -10px;
        }

        .err_proc {
            background-color: red;
            padding: 7px;
            border-radius: 20px;
        }
        .api_proc {
            background-color: #6600cc;
            padding: 7px;
            border-radius: 20px;
        }

        .workarea {
            text-align: left;
            padding: 50px;
        }

        input[type=text] {
            width: 80%;
            padding: 12px 20px;
            margin: 8px 0;
            display: inline-block;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        input[type=submit] {
            width: 40%;
            background-color: #00b359;
            color: white;
            padding: 10px 20px;
            margin: 8px 0;
            border: none;
            border-radius: 4px;
            cursor: pointer;
            font-size: 17px;
        }

        input[type=submit]:hover {
            background-color: #00994d;
        }

        .response_form {
            padding-top: 50px;
        }
    </style>

</head>
<body>
<?php
    $operation = "1";
    $setA = "1,2,3,4,5";
    $setB = "4,5,6,7,8";
    $operName = "";
    $resultSet = "";
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $operation = $_POST['operation'];
        $setA = $_POST['setA'];
        $setB = $_POST['setB'];
        $Aset = array();
        $Bset = array();
        if (empty($operation)) {
            echo "<p class='err_proc'>Invalid Operation</p>";
        }
        elseif (empty($setA)){
            echo "<p class='err_proc'>Invalid Set A</p>";
        }elseif (empty($setB)){
            echo "<p class='err_proc'>Invalid Set B</p>";
        }else{
            $Aset = explode(',',$setA);
            $Bset = explode(',',$setB);
        }

        if($operation == 1){
            $operName = "Union";
            $resultSet = implode(',', (array_merge($Aset, $Bset)));
        } elseif($operation == 2){
            $operName = "Intersection";
            $resultSet = implode(',', (array_intersect($Aset, $Bset)));
        }elseif($operation == 3){
            $operName = "Minus";
            $resultSet = implode(',', (array_diff($Aset, $Bset)));
        }
        else{
            echo "<p class='err_proc'>Something went wrong !</p>";
        }

        }

?>

    <div class="info">
        <h1>Set Theory</h1>
        <p>Returns the result of the given operation on sets</p>
        <p class="api_proc">API Call Format : <span>http://localhost:5000/set
theory?operation=0&setA=1,2,3,4,4,5,5,5,6,7&setB=1,2,3,3,3,4,5,6,7,9,0</span>
</p>
    </div>
    <div class="workarea">

        <form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP
_SELF']);?>">
            <label for="message">Operation ( 1 - Union, 2 - Intersection, 3 -
 Minus)</label><br>
            <input type="text" id="operation" name="operation" value='<?php e
cho $operation ?>'><br>
            <label for="message">Set A ( Comma Seperate Values. Ex= 1,2,3,4,5
)</label><br>
            <input type="text" id="setA" name="setA" value='<?php echo $setA
?>'><br>
            <label for="message">Set A ( Comma Seperate Values. Ex= 4,5,6,7,8
)</label><br>
            <input type="text" id="setB" name="setB" value='<?php echo $setB
?>'><br>
            <input type="submit" value="Submit">
            </form>
            <form class="response_form">
            <label for="response">Result of <?php echo $operName ?></label><b
r>
```

```
            <input type="text" id="response" name="response" value='<?php ech
o $resultSet ?>'><br>
        </form>

    </div>

</body>
</html>
```

Screens :



Set Theory

API Usage : Returns the result of given set and operation

API Call Format : *http://localhost:5000/settheory?operation=0&setA=1,2,3,4,4,5,5,5,6,7&setB=1,2,3,3,3,4,5,6,7,9,0*

Operation (0 - Union, 1 - Intersection, 2 - Minus)

0

Set A (Comma Seperate Values. Ex= 1,2,3,4,5)

1,2,3,4,5

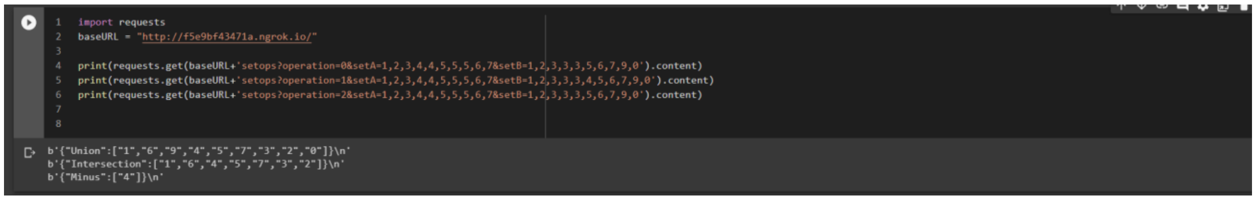
Set A (Comma Seperate Values. Ex= 4,5,6,7,8)

4,5,6,7,8

Result of Union

{'3', '6', '2', '8', '1', '5', '7', '4'}

Test Report:



```
import requests
baseURL = "http://f5e9bf43471a.ngrok.io/"

print(requests.get(baseURL+'setops?operation=0&setA=1,2,3,4,4,5,5,5,6,7&setB=1,2,3,3,3,5,6,7,9,0').content)
print(requests.get(baseURL+'setops?operation=1&setA=1,2,3,4,4,5,5,5,6,7&setB=1,2,3,3,3,4,5,6,7,9,0').content)
print(requests.get(baseURL+'setops?operation=2&setA=1,2,3,4,4,5,5,5,6,7&setB=1,2,3,3,3,5,6,7,9,0').content)
```

```
b'{"Union":["1","6","9","4","5","7","3","2","0"]}\n'
b'{"Intersection":["1","6","4","5","7","3","2"]}\n'
b'{"Minus":["4"]}\n'
```

Ex.No : 3

Question :

Perform matrix operations like Transpose, Lower Diagonal (Left & Right), Upper Diagonal (Left & Right) and Swivel.

Solution :
Python :

```
from flask import Flask, request, Response, jsonify, render_template_string
from flask_restful import Resource, Api, reqparse

app = Flask(__name__)

api = Api(app)


#------------------------------------------------
class MatOps(Resource):
  def get(self):
    print(request.args)

    operation = int(request.args['op'])
    row = int(request.args['row'])
    col = int(request.args['col'])

    matrix = map(int,request.args['matrix'].split(','))

    tempMat = []
    matrixA = []
    colBkp = col
    for idx,key in enumerate(matrix):
      if (idx < col):
        tempMat.append(key)
      else:
        matrixA.append(tempMat)
        tempMat = []
        col += colBkp
        tempMat.append(key)
    matrixA.append(tempMat)

    print('Orgmatrix',matrixA)
    col = colBkp

    if (operation == 0):
      print([[matrixA[j][i] for j in range(len(matrixA))] for i in
range(len(matrixA[0]))])
      return self.home(str(operation),'Transpose',str(row),
str(col),str(matrixA),str(transpose(matrixA,row,col)))
    elif (operation == 1):
      return self.home(str(operation),'Upper Diagonal',str(row),
str(col),str(matrixA),str(upperDiagonal(matrixA,row,col)))
    elif (operation == 2):
      return self.home(str(operation),'Lower Diagonal',str(row),
str(col),str(matrixA),str(lowerDiagonal(matrixA,row, col)))
    else:
      return self.home(str(operation),'Swivel',str(row),
str(col),str(matrixA),str(swivel(matrixA,row, col)))
    return 'Invalid Request !', 200


  def lowerDiagonal(matrixA, row, col):
    rtempMat = []
    ltempMat = []
    lMatrix = []
    rMatrix = []

    for i in range(0, row):
      for j in range(0, col):
        if (j < i):
          ltempMat.append(matrixA[i][j])
        else:
          ltempMat.append(0)

        if (j >= 1) and (i + j > col - 1):
          rtempMat.append(matrixA[i][j])
        else:
          rtempMat.append(0)
      rMatrix.append(rtempMat)
      lMatrix.append(ltempMat)
      rtempMat = []
      ltempMat = []

    print(lMatrix)
    print(rMatrix)

    return {'OriginalMatrix' : matrixA, 'Lower_Right' : rMatrix, 'Lower_Left'
: lMatrix}


  def upperDiagonal(matrixA,row, col):
    rtempMat = []
    ltempMat = []
    lMatrix = []
    rMatrix = []
    if row == col:
      for i in range(0, row):
        for j in range(0, col):
          if (j > i):
            rtempMat.append(matrixA[i][j])
          else:
            rtempMat.append(0)
          if (j <= 1) and (i + j < col - 1):
            ltempMat.append(matrixA[i][j])
          else:
            ltempMat.append(0)
        rMatrix.append(rtempMat)
        lMatrix.append(ltempMat)
        rtempMat = []
        ltempMat = []
    return {'OriginalMatrix' : matrixA, 'Upper_Right' : rMatrix,
'Upper_Left': lMatrix}


  def transpose(matrixA,row, col):
    rMatrix = [[matrixA[j][i] for j in range(len(matrixA))] for i in
range(len(matrixA[0]))]

    return {'OriginalMatrix' : matrixA, 'Tranpose' : rMatrix}


  def swivel(matrixA,row,col):

    return {'OriginalMatrix' : matrixA}



  @app.route('/', methods=['GET','POST'])
  def home(self='',opType='0',operation='',row='3',
col='4',matrix='1,2,3,4,5,6,7,8,9,10,11,12',result=""):
    return render_template_string('''<!DOCTYPE html>
    <html>
    <head>
    <title>Matrix Operations</title>
    <style>
    body {
      background-color: #4db8ff;
      text-align: center;
      color: white;
      font-family: Arial, Helvetica, sans-serif;
    }
    span {
      color : white;
        font-style: oblique;
    }
    .info {
      background-color : #0099ff;
        padding : 20px;
        margin: -10px -10px 0 -10px;
    }
    .api_proc {
      background-color: #6600cc;
        padding:7px;
        border-radius:20px;
    }
    .workarea {
    text-align : left;
    padding : 50px;
    }

    input[type=text] {
      width:150%;
      padding: 12px 20px;
      margin: 8px 0;
      display: inline-block;
      border: 1px solid #ccc;
      border-radius: 4px;
      box-sizing: border-box;

    }
    input[type=submit] {
      width: 40%;
      background-color:   #00b359;
      color: white;
      padding: 10px 20px;
      margin: 8px 0;
      border: none;
      border-radius: 4px;
      cursor: pointer;
      font-size:17px;
    }

    input[type=submit]:hover {
      background-color: #00994d;
    }
    .response_form {
      padding-top:50px;
    }
    </style>
    </head>
    <body>
    <div class="info">
    <h1>Set Theory</h1>
    <p>API Usage : Returns the result of given set and operation</p>
    <p class="api_proc">API Call Format :
<span>http://localhost:5000/settheory?operation=0&setA=1,2,3,4,4,5,5,5,6,7&se
tB=1,2,3,3,3,4,5,6,7,9,0</span></p>
    </div>
    <div class="workarea">

      <form action="/matops">
        <label for="op">Operation </br> (0 - Transpose, 1 - Upper Diagonal
Left & Right, 2 - Upper Diagonal Left & Right, 3 - Swivel)</label><br>
        <input type="text" id="op" name="op" value="'''+opType+'''"><br>
        <label for="row">Rows</label><br>
        <input type="text" id="row" name="row" value="'''+row+'''"><br>
        <label for="col">Columns</label><br>
        <input type="text" id="col" name="col" value="'''+col+'''"><br>
        <label for="matrix">Matrix </br>(Comma seperated values. Ex.
1,2,3,4,5,6,7,8,9,0,1,2,3,4,5)</label><br>
        <textarea type="text" id="matrix" name="matrix" rows="4" cols="50"
>'''+matrix+'''</textarea>
        <br>
        <input type="submit" value="Submit">
      </form>
      <form class="response_form">
        <label for="response">Result of '''+operation+'''</label><br>
        <textarea id="response" name="response" rows="4" cols="50"
>'''+result+'''</textarea>
      </form>
    </div>

    </body>
    </html>

    ''')

  @app.errorhandler(404)
  def page_not_found(e):
      return "<h1>404</h1><p>Page not found.</p>", 404


api.add_resource(MatOps, '/matops')

if __name__ == '__main__':
  app.run()
```

JavaScript :

```
const express = require('express')
const app = express()
const bp = require('body-parser')

app.use(bp.text())

function transpose(array) {
    return array.reduce((prev, next) => next.map((item, i) =>
        (prev[i] || []).concat(next[i])
    ), []);
}

function upperDiagonal(matrix, row, col) {
    var rMatrix = [],
        lMatrix = [],
        rMatRow = [],
        lMatRow = [],
        i, j;

    for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++) {
            if (j > i)
                rMatRow.push(matrix[i][j])
            else
                rMatRow.push(0)
            if (j <= 1 && i + j < col - 1)
                lMatRow.push(matrix[i][j])
            else
                lMatRow.push(0)
        }
        rMatrix.push(rMatRow)
        lMatrix.push(lMatRow)
        rMatRow = []
        lMatRow = []
    }
    console.log(lMatrix)
    console.log(rMatrix)
    return { 'Upper_Left': [...lMatrix], 'Upper_Right': [...rMatrix] }
}

function lowerDiagonal(matrix, row, col) {
    var rMatrix = [],
        lMatrix = [],
        rMatRow = [],
        lMatRow = [],
        i, j;

    for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++) {
            if (j < i)
                lMatRow.push(matrix[i][j])
            else
                lMatRow.push(0)
            if (j >= 1 && i + j > col - 1)
                rMatRow.push(matrix[i][j])
            else
                rMatRow.push(0)
        }
        rMatrix.push(rMatRow)
        lMatrix.push(lMatRow)
        rMatRow = []
        lMatRow = []
    }
    console.log(lMatrix)
    console.log(rMatrix)
    return { 'Lower_Left': [...lMatrix], 'Lower_Right': [...rMatrix] }
}

function swivel(matrix) {

}
app.get('/matops', (req, res) => {

    console.log(req.query)

    var operation = parseInt(req.query.op)
    var row = parseInt(req.query.row)
    var col = parseInt(req.query.col)
    var matrix = req.query.matrix.split(',')
    console.log('matrix', matrix);
    tempMat = []
    matrixA = []
    colBkp = col

    matrix.forEach(function(value, i) {
        if (i < col) {
            tempMat.push(value)
        } else {
            matrixA.push(tempMat)
            tempMat = []
            col += colBkp
            tempMat.push(value)
        }
    });
    matrixA.push(tempMat)
    col = colBkp
    console.log('matrixA', matrixA);

    if (operation == 0)
        return res.send({ 'transpose': [...transpose(matrixA)] });
    else if (operation == 1)
        return res.send(upperDiagonal(matrixA, row, col));
    else
        return res.send(lowerDiagonal(matrixA, row, col));

})

//start
app.listen(3000)
```

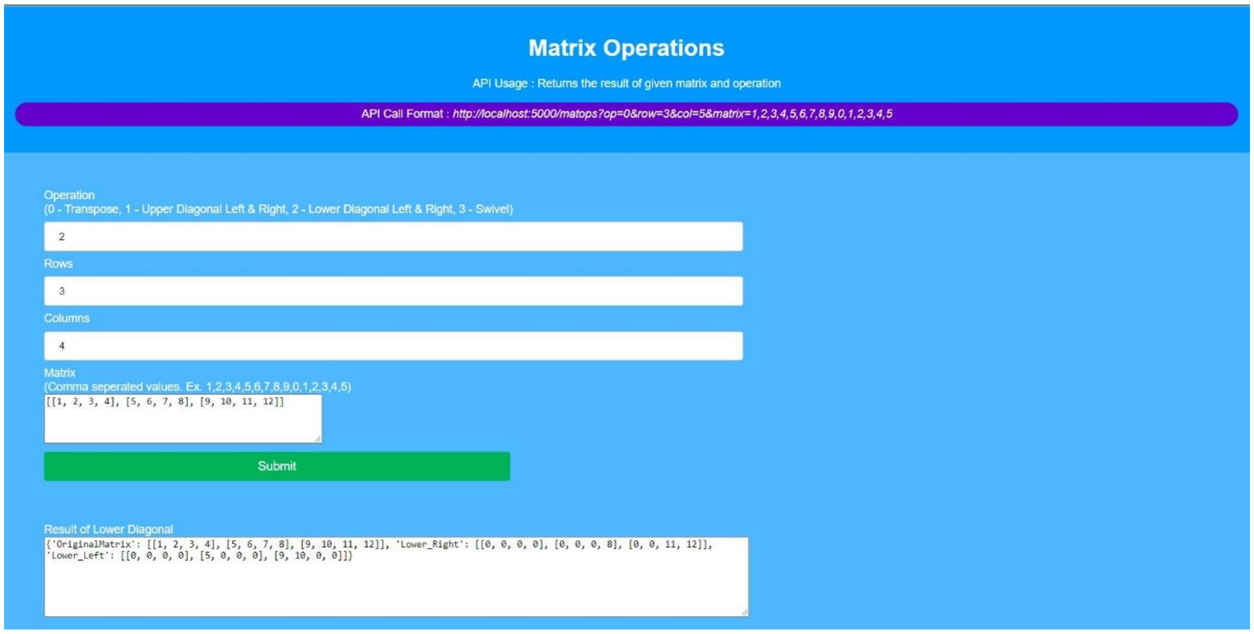
PHP :

```
<!DOCTYPE html>
    <html>
    <head>
    <title>Matrix Operations</title>
    <style>
    body {
      background-color: #4db8ff;
      text-align: center;
      color: white;
      font-family: Arial, Helvetica, sans-serif;
    }
    span {
      color : white;
        font-style: oblique;
    }
    .info {
      background-color : #0099ff;
        padding : 20px;
        margin: -10px -10px 0 -10px;
    }
    .api_proc {
      background-color: #6600cc;
        padding:7px;
        border-radius:20px;
    }
    .err_proc {
            background-color: red;
            padding: 7px;
            border-radius: 20px;
   }
```

```
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width:150%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
  padding-top:50px;
}
</style>
</head>
<body>
  <?php
  $op = 0; $row = 3; $col = 5; $opName = ""; $result = ""; $strMat = "1,2
,3,4,5,6,7,8,9,0,11,12,13,14,15";

  function transpose($row,$col,$matrixA){

    $tMatrix = array();
    foreach($matrixA as $idx=>$drow){
      foreach($drow as $idxv => $val){
          $tMatrix[$idxv][] = $val;
      }
    }
    $tmpArr = array();
    foreach ($tMatrix as $arr) {
    $tmpArr[] = implode(',', $arr);
    }
    return 'Transpose => '.implode(',',$tmpArr);
```

```
      }

    function lowerDiagonal($row,$col, $matrixA){
      $rtempMat = array(); $ltempMat = array(); $lMatrix = array(); $rMatrix
= array();

      for($i=0;$i<$row;$i++){
        for($j=0;$j<$col;$j++){
          if ($j < $i){
            array_push($ltempMat,$matrixA[$i][$j]);
          }
          else{
            array_push($ltempMat, 0);
          }

          if (($j >= 1) && ($i + $j > $col - 1)){
            array_push($rtempMat,$matrixA[$i][$j]);
          }

          else{
            array_push($rtempMat,0);
          }
        }
        array_push($rMatrix,$rtempMat);
        array_push($lMatrix,$ltempMat);
        $rtempMat = array(); $ltempMat = array();
      }
      $tmpArr = array();
      foreach ($lMatrix as $arr) {
      $tmpArr[] = implode(',', $arr);
      }
      $tmpArr2 = array();
      foreach ($rMatrix as $arr) {
      $tmpArr2[] = implode(',', $arr);
      }
      return "LowerLeft => ".implode(",",$tmpArr)." | Lower_Right => ".implod
e(", ",$tmpArr2);
    }

    function upperDiagonal($row,$col, $matrixA){
      $rtempMat = array(); $ltempMat = array(); $lMatrix = array(); $rMatrix
= array();

      for($i=0;$i<$row;$i++){
        for($j=0;$j<$col;$j++){
          if ($j > $i){
            array_push($ltempMat,$matrixA[$i][$j]);
          }
          else{
            array_push($ltempMat, 0);
          }
          if (($j <= 1) && ($i + $j < $col - 1)){
            array_push($rtempMat,$matrixA[$i][$j]);
          }

          else{
            array_push($rtempMat,0);
          }
        }
        array_push($rMatrix,$rtempMat);
        array_push($lMatrix,$ltempMat);
        $rtempMat = array(); $ltempMat = array();
      }
      $tmpArr = array();
      foreach ($lMatrix as $arr) {
      $tmpArr[] = implode(',', $arr);
      }
      $tmpArr2 = array();
      foreach ($rMatrix as $arr) {
      $tmpArr2[] = implode(',', $arr);
      }
      return "UpperLeft => ".implode(", ",$tmpArr)." | UpperRight => ".implod
e(", ",$tmpArr2);
    }



      if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $op = $_POST['op']; $row = $_POST['row']; $col = $_POST['col']; $strM
at = $_POST['matrix'];

        if(empty($op) || empty($row) || empty($col) || empty($strMat)){

          echo "<p class='err_proc'>Invalid Inputs</p>";
        }
        $tempMat = array(); $matrixA = array(); $colBkp = $col;

        $strMat = explode(",",$strMat);
        foreach($strMat as $idx=>$key){
          if($idx < $col){
            array_push($tempMat,$key);
          }
          else{
            array_push($matrixA,$tempMat);
            $tempMat = array();
            $col += $colBkp;
            array_push($tempMat,$key);
          }
          array_push($matrixA,$tempMat);
        }
        $col = $colBkp;

        if($op == 0){
          $result  = transpose($row, $col, $matrixA);
        }elseif($op == 1){
          $result  = upperDiagonal($row, $col, $matrixA);
        }elseif($op == 2){
          $result  = lowerDiagonal($row, $col, $matrixA);
        }else{
          echo "<p class='err_proc'>Invalid Operation</p>";
        }
        $strMat = implode(",",$strMat);
      }
      ?>


    <div class="info">
    <h1>Matrix Operations</h1>
    <p>API Usage : Returns the result of given matrix and operation</p>
    <p class="api_proc">API Call Format : <span>http://localhost:5000/matops?
op=0&row=3&col=5&matrix=1,2,3,4,5,6,7,8,9,0,1,2,3,4,5</span></p>
    </div>
    <div class="workarea">

    <form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SEL
F']);?>">
        <label for="op">Operation </br> (0 - Transpose, 1 - Upper Diagonal Le
ft & Right, 2 - Lower Diagonal Left & Right, 3 - Swivel)</label><br>
        <input type="text" id="op" name="op" value='<?php echo $op?>'><br>
        <label for="row">Rows</label><br>
        <input type="text" id="row" name="row" value='<?php echo $row?>'><br>
        <label for="col">Columns</label><br>
        <input type="text" id="col" name="col" value='<?php echo $col?>'><br>
        <label for="matrix">Matrix </br>(Comma seperated values. Ex. 1,2,3,4,
5,6,7,8,9,0,1,2,3,4,5)</label><br>
        <textarea type="text" id="matrix" name="matrix" rows="4" cols="50" ><
?php echo $strMat?></textarea>
        <br>
        <input type="submit" value="Submit">
      </form>
      <form class="response_form">
        <label for="response">Result of <?php echo $opName?></label><br>
        <textarea id="response" name="response" rows="4" cols="50" > <?php ec
ho $result?></textarea>
      </form>
    </div>

    </body>
    </html>
```

Screens :



Matrix Operations

API Usage : Returns the result of given matrix and operation

API Call Format : *http://localhost:5000/matops?op=0&row=3&col=5&matrix=1,2,3,4,5,6,7,8,9,0,1,2,3,4,5*

Operation
(0 - Transpose, 1 - Upper Diagonal Left & Right, 2 - Lower Diagonal Left & Right, 3 - Swivel)

2

Rows

3

Columns

4

Matrix
(Comma seperated values. Ex. 1,2,3,4,5,6,7,8,9,0,1,2,3,4,5)

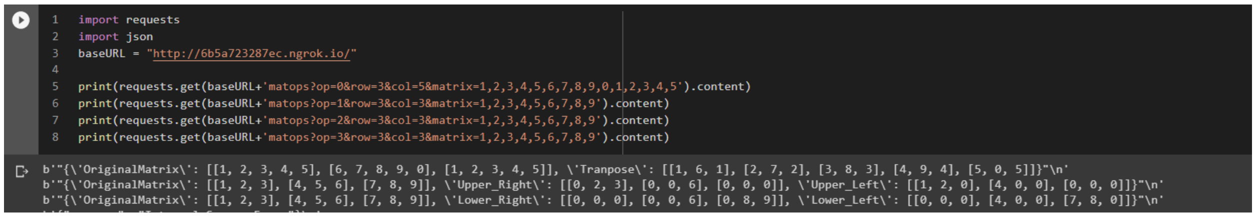
```
[[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]
```

Submit

Result of Lower Diagonal

```
{'OriginalMatrix': [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]], 'Lower_Right': [[0, 0, 0, 0], [0, 0, 0, 8], [0, 0, 11, 12]],
'Lower_Left': [[0, 0, 0, 0], [5, 0, 0, 0], [9, 10, 0, 0]]}
```

Test Report :



```python
import requests
import json
baseURL = "http://6b5a723287ec.ngrok.io/"

print(requests.get(baseURL+'matops?op=0&row=3&col=5&matrix=1,2,3,4,5,6,7,8,9,0,1,2,3,4,5').content)
print(requests.get(baseURL+'matops?op=1&row=3&col=3&matrix=1,2,3,4,5,6,7,8,9').content)
print(requests.get(baseURL+'matops?op=2&row=3&col=3&matrix=1,2,3,4,5,6,7,8,9').content)
print(requests.get(baseURL+'matops?op=3&row=3&col=3&matrix=1,2,3,4,5,6,7,8,9').content)
```

```
b'"{\'OriginalMatrix\': [[1, 2, 3, 4, 5], [6, 7, 8, 9, 0], [1, 2, 3, 4, 5]], \'Tranpose\': [[1, 6, 1], [2, 7, 2], [3, 8, 3], [4, 9, 4], [5, 0, 5]]}"\n'
b'"{\'OriginalMatrix\': [[1, 2, 3], [4, 5, 6], [7, 8, 9]], \'Upper_Right\': [[0, 2, 3], [0, 0, 6], [0, 0, 0]], \'Upper_Left\': [[1, 2, 0], [4, 0, 0], [0, 0, 0]]}"\n'
b'"{\'OriginalMatrix\': [[1, 2, 3], [4, 5, 6], [7, 8, 9]], \'Lower_Right\': [[0, 0, 0], [0, 0, 6], [0, 8, 9]], \'Lower_Left\': [[0, 0, 0], [4, 0, 0], [7, 8, 0]]}"\n'
```

Ex.No : 4

Question :

Convert the figure into words in currency.

Solution :

Python :

```
from flask import Flask, request, Response, jsonify
from flask_restful import Resource, Api, reqparse

app = Flask(__name__)

api = Api(app)

class FigureToCurrency(Resource):
  def get(self):
    print(request.args)

    number = int(request.args['value'])

    ones = ("", "one", "two", "three", "four", "five", "six", "seven",
"eight", "nine")
    tens = ("", "", "twenty", "thirty", "forty", "fifty", "sixty", "seventy",
"eighty", "ninety")
    teens = ("ten", "eleven", "twelve", "thirteen", "fourteen", "fifteen",
"sixteen", "seventeen", "eighteen", "nineteen")
    levels = ("", "thousand", "million", "billion", "trillion",
"quadrillion", "quintillion", "sextillion", "septillion", "octillion",
"nonillion")

    word = ""
    orgNum = number
    num = reversed(str(number))
    number = ""
    for x in num:
        number += x
    del num
    if len(number) % 3 == 1: number += "0"
    x = 0
    for digit in number:
        if x % 3 == 0:
            word = levels[x // 3] + ", " + word
            n = int(digit)
        elif x % 3 == 1:
            if digit == "1":
                num = teens[n]
            else:
                num = tens[int(digit)]
                if n:
                    if num:
                        num += "-" + ones[n]
                    else:
                        num = ones[n]
            word = num + " " + word
        elif x % 3 == 2:
            if digit != "0":
                word = ones[int(digit)] + " hundred " + word
        x += 1
    return home(str(orgNum),word.strip(", "))


api.add_resource(FigureToCurrency, '/currency')

#common error handler
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404


@app.route('/', methods=['GET','POST'])
def home(gvalue="5001",value=""):
  return '''<!DOCTYPE html>
<html>
<head>
<title>Figures to Currency</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
       color : white;
    font-style: oblique;
}
.info {
       background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
       background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width:150%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
        padding-top:50px;
}
</style>


</head>
<body>
<div class="info">
<h1>Figures to Currency</h1>
<p>API Usage : Returns the words of given number value</p>
<p class="api_proc">API Call Format :
<span>http://localhost:5000/currency?value=5001</span></p>
</div>
<div class="workarea">

<form action="/currency">
  <label for="message">Figures</label><br>
  <input type="text" id="value" name="value" value="'''+gvalue+'''"><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">Words</label><br>
  <input type="text" id="response" name="response" value="'''+value+'''"><br>
</form>

</div>

</body>
</html>

'''

if __name__ == '__main__':
  app.run()
```

JavaScript :

```
const express = require('express')
const app = express()
const bp = require('body-parser')

app.use(bp.text())

app.get('/numtocurrency', (req, res) => {

    var number = req.query.value
    var a = ['', 'one ', 'two ', 'three ', 'four ', 'five ', 'six ', 'seven '
, 'eight ', 'nine ', 'ten ', 'eleven ', 'twelve ', 'thirteen ', 'fourteen ',
'fifteen ', 'sixteen ', 'seventeen ', 'eighteen ', 'nineteen '];
    var b = ['', '', 'twenty', 'thirty', 'forty', 'fifty', 'sixty', 'seventy'
, 'eighty', 'ninety'];

    if ((number = number.toString()).length > 9) return res.send('Overflow')
    n = ('000000000' + number).substr(-
9).match(/^(\d{2})(\d{2})(\d{2})(\d{1})(\d{2})$/);
    if (!n) return res.send('Invalid Request!');
    var inString = '';
    inString += (n[1] != 0) ? (a[Number(n[1])] || b[n[1][0]] + ' ' + a[n[1][1
]]) + 'crore ' : '';
    inString += (n[2] != 0) ? (a[Number(n[2])] || b[n[2][0]] + ' ' + a[n[2][1
]]) + 'lakh ' : '';
    inString += (n[3] != 0) ? (a[Number(n[3])] || b[n[3][0]] + ' ' + a[n[3][1
]]) + 'thousand ' : '';
    inString += (n[4] != 0) ? (a[Number(n[4])] || b[n[4][0]] + ' ' + a[n[4][1
]]) + 'hundred ' : '';
    inString += (n[5] != 0) ? ((inString != '') ? 'and ' : '') + (a[Number(n[
5])] || b[n[5][0]] + ' ' + a[n[5][1]]) + 'only ' : '';
    return res.send(inString)
})

//start
app.listen(3000)
```

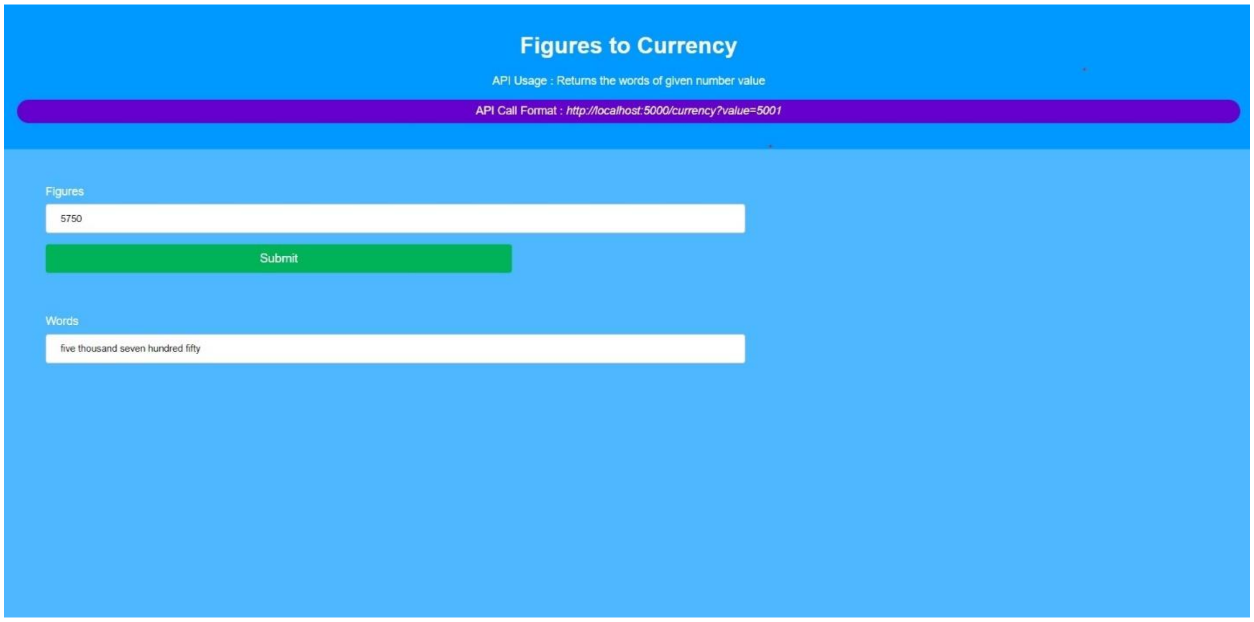
PHP :

```
<!DOCTYPE html>
<html><head>
    <title>FigureToCurrency</title>
    <style>
        body {
            background-color: #4db8ff;
            text-align: center;
            color: white;
            font-family: Arial, Helvetica, sans-serif;
        }

        span {
```

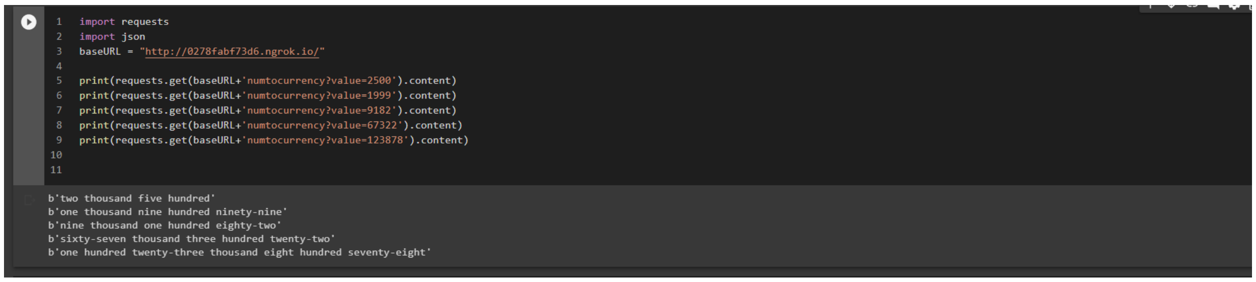
```
        color: white;
        font-style: oblique;
    }

    .info {
        background-color: #0099ff;
        padding: 20px;
        margin: -10px -10px 0 -10px;
    }

    .err_proc {
        background-color: red;
        padding: 7px;
        border-radius: 20px;
    }
    .api_proc {
        background-color: #6600cc;
        padding: 7px;
        border-radius: 20px;
    }

    .workarea {
        text-align: left;
        padding: 50px;
    }

    input[type=text] {
        width: 80%;
        padding: 12px 20px;
        margin: 8px 0;
        display: inline-block;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-sizing: border-box;
    }

    input[type=submit] {
        width: 40%;
        background-color: #00b359;
        color: white;
        padding: 10px 20px;
        margin: 8px 0;
        border: none;
        border-radius: 4px;
        cursor: pointer;
        font-size: 17px;
    }

    input[type=submit]:hover {
        background-color: #00994d;
    }

    .response_form {
            padding-top: 50px;
        }
    </style>

</head>
<body>
<?php
    function console_log($output, $with_script_tags = true) {
        $js_code = 'console.log(' . json_encode($output, JSON_HEX_TAG) . ');'
;
        if ($with_script_tags) {
            $js_code = '<script>' . $js_code . '</script>';
        }
        echo $js_code;
    }
    $number = "3450";
    $words = "";
    $flag = 0;
    $a = array("", "one", "two", "three", "four", "five", "six", "seven", "ei
ght", "nine", "ten", "eleven", "twelve", "thirteen", "fourteen", "fifteen", "
sixteen", "seventeen", "eighteen", "nineteen", "twenty", '30'=>"thirty", '40'
=>"forty", '50'=>"fifty", '60'=>"sixty", '70'=>"seventy", '80'=>"eighty", '80
'=>"ninety");
    $levels = array(" ", "hundred ", "thousand ", "lakh ", "crore ");

    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $number = $numBkp = $_POST['value'];

        if (empty($number)) {
            echo "<p class='err_proc'>Invalid Figure</p>";
            $flag = 1;
        }
        else {
            $noOfDigits = strlen($number);
            console_log('Dnum'.$noOfDigits);
            console_log('num'.$number);

            $digitsLeft = 0;
            $cValue = array();
            if ($noOfDigits > 9)  {
                echo "<p class='err_proc'>Overflow !</p>";
                $flag = 1;
            }
            else{
               while( $digitsLeft < $noOfDigits){

                $curNum = floor($number % (($digitsLeft == 2) ? 10 : 100));
                $number = floor($number / (($digitsLeft == 2) ? 10 : 100));

                if($curNum){
                    $level = count($cValue);
                    $cValue [] = ($curNum < 20) ? $a[$curNum].' '.$levels[$le
vel] : $a[floor($curNum / 10) * 10].' '.$a[$curNum % 10].' '.$levels[$level];
```

```
                }
                else $cValue[] = null;

                $digitsLeft += (($digitsLeft == 2) ? 10 : 100) == 10 ? 1 : 2;
            }
                $words = implode('', array_reverse($cValue));
                $number = $numBkp;
            }
            }


        if (empty($words) && $flag != 1) {
            echo "<p class='err_proc'>Something Went Wrong !</p>".$words;
        }
    }
?>


<div class="info">
<h1>Figures to Currency</h1>
<p>API Usage : Returns the words of given number value</p>
<p class="api_proc">API Call Format : <span>http://localhost:5000/currency?va
lue=5001</span></p>
</div>
<div class="workarea">

<form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF'])
;?>">
  <label for="message">Figures</label><br>
  <input type="text" id="value" name="value" value='<?php echo $number?>'><br
>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">Words</label><br>
  <input type="text" id="response" name="response" value='<?php echo $words?>
'><br>
</form>
</div>

</body>
</html>
```

Screens :


Figures to Currency
API Usage : Returns the words of given number value
API Call Format : http://localhost:5000/currency?value=5001
Figures
5750
Submit
Words
five thousand seven hundred fifty

Test Report :



```
import requests
import json
baseURL = "http://0278fabf73d6.ngrok.io/"

print(requests.get(baseURL+'numtocurrency?value=2500').content)
print(requests.get(baseURL+'numtocurrency?value=1999').content)
print(requests.get(baseURL+'numtocurrency?value=9182').content)
print(requests.get(baseURL+'numtocurrency?value=67322').content)
print(requests.get(baseURL+'numtocurrency?value=123878').content)
```

```
b'two thousand five hundred'
b'one thousand nine hundred ninety-nine'
b'nine thousand one hundred eighty-two'
b'sixty-seven thousand three hundred twenty-two'
b'one hundred twenty-three thousand eight hundred seventy-eight'
```

Ex.No : 5

Question :

Generate the checksum value for the given sentence using md5 algorithm.

Solution :

Python :

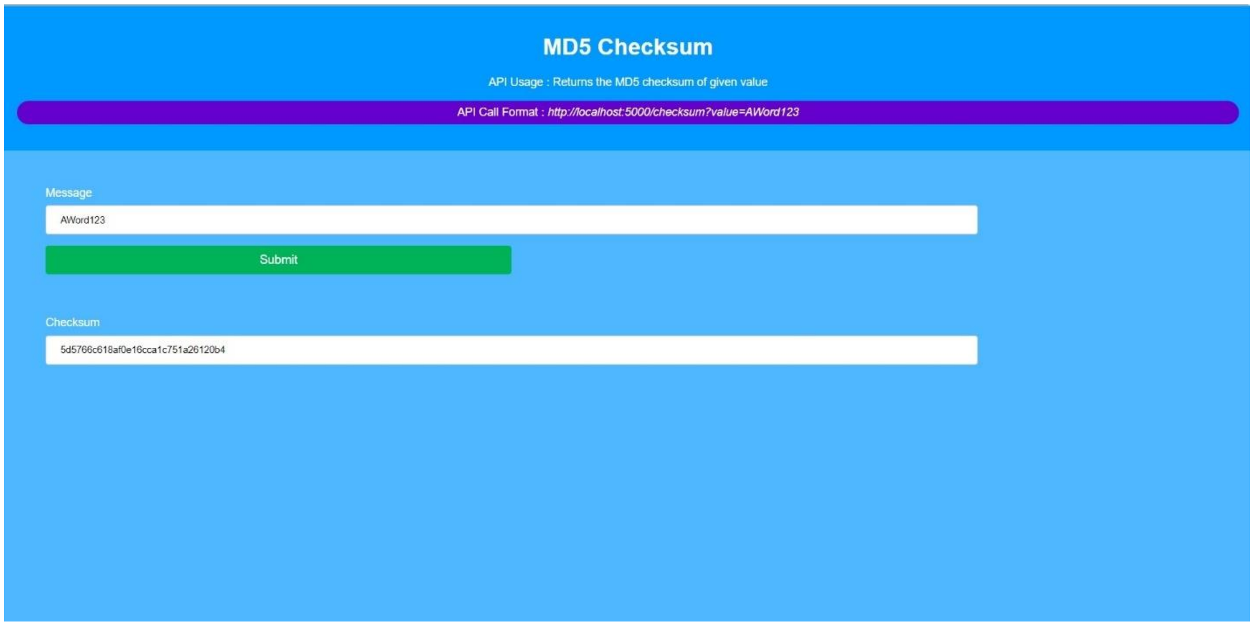
```
from flask import Flask, request, Response, jsonify
import json
import math
import struct
import sys

app = Flask(__name__)

def prepare_message(message):

    paddingSize = (64 - 1 - 8 - len(message) % 64) % 64
    lengthInBits = (len(message) * 8) % 2 ** 64
    return message + b"\x80" + paddingSize * b"\x00" + struct.pack("<Q",
lengthInBits)

def rotate_left(n, amount):
    return ((n << amount) & 0xffff_ffff) | (n >> (32 - amount))

def hash_chunk(state, chunk):
    (a, b, c, d) = state

    for i in range(64):
        if i < 16:
            bits = (b & c) | (~b & d)
            index = i
            shift = (7, 12, 17, 22)[i % 4]
        elif i < 32:
            bits = (d & b) | (c & ~d)
            index = (5 * i + 1) % 16
            shift = (5, 9, 14, 20)[i % 4]
        elif i < 48:
            bits = b ^ c ^ d
            index = (3 * i + 5) % 16
            shift = (4, 11, 16, 23)[i % 4]
        else:
            bits = c ^ (b | ~d)
            index = 7 * i % 16
            shift = (6, 10, 15, 21)[i % 4]

        const = math.floor(abs(math.sin(i + 1)) * 2 ** 32)
        bAdd = (const + a + bits + chunk[index]) & 0xffff_ffff
        bAdd = rotate_left(bAdd, shift)

        (a, b, c, d) = (d, (b + bAdd) & 0xffff_ffff, b, c)

    return (a, b, c, d)
```

```
def md5(message):

    # set initial state
    state = [0x67452301, 0xefcdab89, 0x98badcfe, 0x10325476]
    # prepare message and process it in chunks of 64 bytes (16 32-bit
integers)
    for chunk in struct.iter_unpack("<16I", prepare_message(message)):
        # hash the chunk; add each 32-bit integer to the corresponding
integer in the state
        hash_ = hash_chunk(state, chunk)
        state = [(s + h) & 0xffff_ffff for (s, h) in zip(state, hash_)]
    # the final state is the hash
    return b"".join(struct.pack("<I", number) for number in state)

@app.route('/checksum', methods=['GET'])
def getChecksum():

    message = request.args['message']
    return home(message,md5(message.encode("utf-8")).hex()), 200

#common error handler
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404

@app.route('/', methods=['GET','POST'])
def home(gvalue="AWord123",value=""):
  return '''<!DOCTYPE html>
<html>
<head>
<title>MD5 Checksum</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
       color : white;
    font-style: oblique;
}
.info {
       background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
       background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}
```

```
input[type=text] {
  width: 80%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
        padding-top:50px;
}
</style>

</head>
<body>
<div class="info">
<h1>CAPTCHA</h1>
<p>API Usage : Returns the MD5 checksum of given value</p>
<p class="api_proc">API Call Format :
<span>http://localhost:5000/checksum?value=AWord123</span></p>
</div>
<div class="workarea">

<form action="/checksum">
  <label for="message">Message</label><br>
  <input type="text" id="message" name="message" value="'''+gvalue+'''"><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">Checksum</label><br>
  <input type="text" id="response" name="response" value="'''+value+'''"><br>
</form>

</div>

</body>
</html>

'''
```

```
if __name__ == "__main__":
  app.run()
```

Screens :



MD5 Checksum

API Usage : Returns the MD5 checksum of given value

API Call Format : *http://localhost:5000/checksum?value=AWord123*

Message

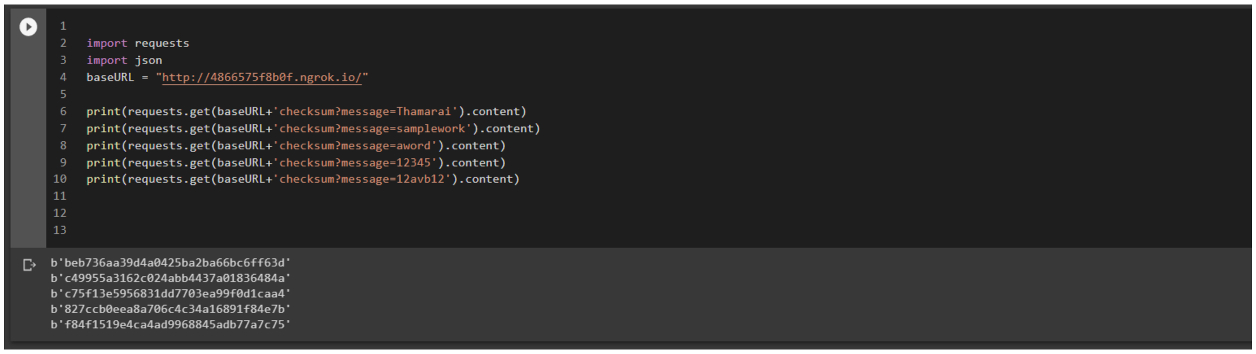
AWord123

Submit

Checksum

5d5766c618af0e16cca1c751a26120b4

Test Report :



```

import requests
import json
baseURL = "http://4866575f8b0f.ngrok.io/"

print(requests.get(baseURL+'checksum?message=Thamarai').content)
print(requests.get(baseURL+'checksum?message=samplework').content)
print(requests.get(baseURL+'checksum?message=aword').content)
print(requests.get(baseURL+'checksum?message=12345').content)
print(requests.get(baseURL+'checksum?message=12avb12').content)



```

```
b'beb736aa39d4a0425ba2ba66bc6ff63d'
b'c49955a3162c024abb4437a01836484a'
b'c75f13e5956831dd7703ea99f0d1caa4'
b'827ccb0eea8a706c4c34a16891f84e7b'
b'f84f1519e4ca4ad9968845adb77a7c75'
```

Ex.No : 6

Question :

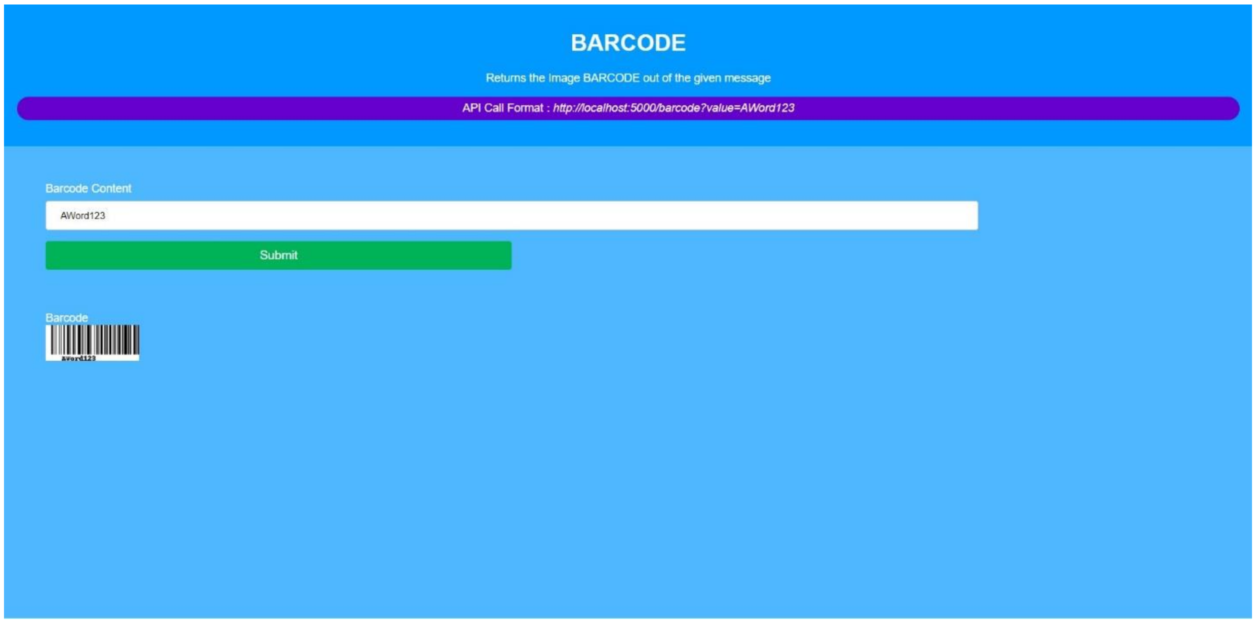
Generate 128-bit bar code for alphanumeric data.

Solution :

Python :

```
from flask import Flask, request, Response
import math, random
import base64

app = Flask(__name__)


#common error handler
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404


class Code128:

   CharSetA =  {
                ' ':0, '!':1, '"':2, '#':3, '$':4, '%':5, '&':6, "'":7,
                '(':8, ')':9, '*':10, '+':11, ',':12, '-':13, '.':14, '/':15,
                '0':16, '1':17, '2':18, '3':19, '4':20, '5':21, '6':22,
'7':23,
                '8':24, '9':25, ':':26, ';':27, '<':28, '=':29, '>':30,
'?':31,
                '@':32, 'A':33, 'B':34, 'C':35, 'D':36, 'E':37, 'F':38,
'G':39,
                'H':40, 'I':41, 'J':42, 'K':43, 'L':44, 'M':45, 'N':46,
'O':47,
                'P':48, 'Q':49, 'R':50, 'S':51, 'T':52, 'U':53, 'V':54,
'W':55,
                'X':56, 'Y':57, 'Z':58, '[':59, '\\':60, ']':61, '^':62,
'_':63,
                '\x00':64, '\x01':65, '\x02':66, '\x03':67, '\x04':68,
'\x05':69, '\x06':70, '\x07':71,
                '\x08':72, '\x09':73, '\x0A':74, '\x0B':75, '\x0C':76,
'\x0D':77, '\x0E':78, '\x0F':79,
                '\x10':80, '\x11':81, '\x12':82, '\x13':83, '\x14':84,
'\x15':85, '\x16':86, '\x17':87,
                '\x18':88, '\x19':89, '\x1A':90, '\x1B':91, '\x1C':92,
'\x1D':93, '\x1E':94, '\x1F':95,
                'FNC3':96, 'FNC2':97, 'SHIFT':98, 'Code C':99, 'Code B':100,
'FNC4':101, 'FNC1':102, 'START A':103,
                'START B':104, 'START C':105, 'STOP':106
            }

   CharSetB = {
                ' ':0, '!':1, '"':2, '#':3, '$':4, '%':5, '&':6, "'":7,
                '(':8, ')':9, '*':10, '+':11, ',':12, '-':13, '.':14, '/':15,
                '0':16, '1':17, '2':18, '3':19, '4':20, '5':21, '6':22,
'7':23,
                '8':24, '9':25, ':':26, ';':27, '<':28, '=':29, '>':30,
'?':31,
                '@':32, 'A':33, 'B':34, 'C':35, 'D':36, 'E':37, 'F':38,
'G':39,
                'H':40, 'I':41, 'J':42, 'K':43, 'L':44, 'M':45, 'N':46,
'O':47,
                'P':48, 'Q':49, 'R':50, 'S':51, 'T':52, 'U':53, 'V':54,
'W':55,
                'X':56, 'Y':57, 'Z':58, '[':59, '\\':60, ']':61, '^':62,
'_':63,
                '' :64, 'a':65, 'b':66, 'c':67, 'd':68, 'e':69, 'f':70,
'g':71,
                'h':72, 'i':73, 'j':74, 'k':75, 'l':76, 'm':77, 'n':78,
'o':79,
                'p':80, 'q':81, 'r':82, 's':83, 't':84, 'u':85, 'v':86,
'w':87,
                'x':88, 'y':89, 'z':90, '{':91, '|':92, '}':93, '~':94,
'\x7F':95,
                'FNC3':96, 'FNC2':97, 'SHIFT':98, 'Code C':99, 'FNC4':100,
'Code A':101, 'FNC1':102, 'START A':103,
                'START B':104, 'START C':105, 'STOP':106
           }

   CharSetC = {
                '00':0, '01':1, '02':2, '03':3, '04':4, '05':5, '06':6,
'07':7,
                '08':8, '09':9, '10':10, '11':11, '12':12, '13':13, '14':14,
'15':15,
                '16':16, '17':17, '18':18, '19':19, '20':20, '21':21,
'22':22, '23':23,
                '24':24, '25':25, '26':26, '27':27, '28':28, '29':29,
'30':30, '31':31,
                '32':32, '33':33, '34':34, '35':35, '36':36, '37':37,
'38':38, '39':39,
                '40':40, '41':41, '42':42, '43':43, '44':44, '45':45,
'46':46, '47':47,
                '48':48, '49':49, '50':50, '51':51, '52':52, '53':53,
'54':54, '55':55,
                '56':56, '57':57, '58':58, '59':59, '60':60, '61':61,
'62':62, '63':63,
                '64':64, '65':65, '66':66, '67':67, '68':68, '69':69,
'70':70, '71':71,
                '72':72, '73':73, '74':74, '75':75, '76':76, '77':77,
'78':78, '79':79,
                '80':80, '81':81, '82':82, '83':83, '84':84, '85':85,
'86':86, '87':87,
                '88':88, '89':89, '90':90, '91':91, '92':92, '93':93,
'94':94, '95':95,
                '96':96, '97':97, '98':98, '99':99, 'Code B':100, 'Code
A':101, 'FNC1':102, 'START A':103,
                'START B':104, 'START C':105, 'STOP':106
           }


   ValueEncodings = {  0:'11011001100',  1:'11001101100',  2:'11001100110',
       3:'10010011000',  4:'10010001100',  5:'10001001100',
       6:'10011001000',  7:'10011000100',  8:'10001100100',
        9:'11001001000', 10:'11001000100', 11:'11000100100',
        12:'10110011100', 13:'10011011100', 14:'10011001110',
        15:'10111001100', 16:'10011101100', 17:'10011100110',
        18:'11001110010', 19:'11001011100', 20:'11001001110',
        21:'11011100100', 22:'11001110100', 23:'11101101110',
        24:'11101001100', 25:'11100101100', 26:'11100100110',
        27:'11101100100', 28:'11100110100', 29:'11100110010',
        30:'11011011000', 31:'11011000110', 32:'11000110110',
        33:'10100011000', 34:'10001011000', 35:'10001000110',
        36:'10110001000', 37:'10001101000', 38:'10001100010',
        39:'11010001000', 40:'11000101000', 41:'11000100010',
        42:'10110111000', 43:'10110001110', 44:'10001101110',
        45:'10111011000', 46:'10111000110', 47:'10001110110',
        48:'11101110110', 49:'11010001110', 50:'11000101110',
        51:'11011101000', 52:'11011100010', 53:'11011101110',
        54:'11101011000', 55:'11101000110', 56:'11100010110',
        57:'11101101000', 58:'11101100010', 59:'11100011010',
        60:'11101111010', 61:'11001000010', 62:'11110001010',
        63:'10100110000', 64:'10100001100', 65:'10010110000',
        66:'10010000110', 67:'10000101100', 68:'10000100110',
        69:'10110010000', 70:'10110000100', 71:'10011010000',
        72:'10011000010', 73:'10000110100', 74:'10000110010',
        75:'11000010010', 76:'11001010000', 77:'11110111010',
        78:'11000010100', 79:'10001111010', 80:'10100111100',
        81:'10010111100', 82:'10010011110', 83:'10111100100',
        84:'10011110100', 85:'10011110010', 86:'11110100100',
        87:'11110010100', 88:'11110010010', 89:'11011011110',
        90:'11011110110', 91:'11110110110', 92:'10101111000',
        93:'10100011110', 94:'10001011110', 95:'10111101000',
        96:'10111100010', 97:'11110101000', 98:'11110100010',
        99:'10111011110',100:'10111101110',101:'11101011110',
        102:'11110101110',103:'11010000100',104:'11010010000',
        105:'11010011100',106:'11000111010'
                        }



   def makeCode(self, code):
    """ Create the binary code return a string which contains "0" for white
bar, "1" for black bar """

    current_charset = None
    pos=sum=0
    skip=False
    for c in range(len(code)):
        if skip:
            skip=False
            continue

        #Only switch to char set C if next four chars are digits
        if len(code[c:]) >=4 and code[c:c+4].isdigit() and
current_charset!=self.CharSetC or len(code[c:]) >=2 and code[c:c+2].isdigit()
and current_charset==self.CharSetC:
            #If char set C = current and next two chars ar digits, keep C
            if current_charset!=self.CharSetC:
                #Switching to Character set C
                if pos:
                    strCode += self.ValueEncodings[current_charset['Code C']]
                    sum  += pos * current_charset['Code C']
                else:
                    strCode= self.ValueEncodings[self.CharSetC['START C']]
                    sum = self.CharSetC['START C']
                current_charset= self.CharSetC
                pos+=1
        elif  code[c] in self.CharSetB and current_charset!=self.CharSetB and
not( code[c] in self.CharSetA and current_charset==self.CharSetA):
            #If char in chrset A = current, then just keep that
            # Switching to Character set B
            if pos:
                strCode += self.ValueEncodings[current_charset['Code B']]
                sum  += pos * current_charset['Code B']
            else:
                strCode= self.ValueEncodings[self.CharSetB['START B']]
                sum = self.CharSetB['START B']
            current_charset= self.CharSetB
            pos+=1
        elif code[c] in self.CharSetA and current_charset!=self.CharSetA and
not(code[c] in self.CharSetB  and current_charset==self.CharSetB):
            # if char in chrset B== current, then just keep that
            # Switching to Character set A
            if pos:
                strCode += self.ValueEncodings[current_charset['Code A']]
                sum  += pos * current_charset['Code A']
            else:
                strCode += self.ValueEncodings[self.CharSetA['START A']]
                sum = self.CharSetA['START A']
            current_charset= self.CharSetA
            pos+=1

        if current_charset==self.CharSetC:
            val= self.CharSetC[code[c:c+2]]
            skip=True
        else:
            val=current_charset[code[c]]

        sum += pos * val
        strCode += self.ValueEncodings[val]
        pos+=1

    #Checksum
    checksum= sum % 103

    strCode +=  self.ValueEncodings[checksum]

    #The stop character
    strCode += self.ValueEncodings[current_charset['STOP']]

    #Termination bar
    strCode += "11"
    return strCode



   def getImage(self, value, height = 50, extension = "PNG", path = "\\"):
      """ Get an image with PIL library value code barre value height height
in pixel of the bar code extension image file extension"""
      from PIL import Image, ImageFont, ImageDraw
      import os
      # from string import lower, upper
      path = os.getcwd()

      # Get the bar code list
      bits = self.makeCode(value)

      # Create a new image
      position = 8
      im = Image.new("1",(len(bits)+position,height))

      # Load font/content/courB08.pil
      font = ImageFont.load(path+"/courB08.pil")

      # Create drawer
      draw = ImageDraw.Draw(im)

      # Erase image
      draw.rectangle(((0,0),(im.size[0],im.size[1])),fill=256)

      # Draw text
      draw.text((23, height-9), value, font=font, fill=0)

      # Draw the bar codes
      for bit in range(len(bits)):
         if bits[bit] == '1':
            draw.rectangle(((bit+position,0),(bit+position,height-
10)),fill=0)

      # Save the result image
      im.save(path+"/"+value+"."+extension.lower(),extension.upper())
      im.show()
      from IPython.display import display
      display(im)
      return im


def testWithChecksum():
   """ Test bar code with checksum """
   bar = Code128()

assert(bar.makeCode('HI345678')=='1101001000011000101000110001000101011101111
0100010110001110001011011000010100100001001101100011101011')

def testImage(value):
  import io
  """ Test images generation with PIL """
  bar = Code128()
  img = bar.getImage(value,50,"gif")
  fp = io.BytesIO()

  img.save(fp,"PNG")
  fp.seek(0)
  # resp = Response(fp.getvalue(),mimetype="image/png")
  # fp.close()

  # return resp
  # return bar.getImage("978221211070",50,"png"),200
  return fp.getvalue()

@app.route('/barcode', methods=['GET'])
def test():
   """ Execute all tests """
   testWithChecksum()
   value = request.args['value']
   data64 = base64.b64encode(testImage(value))
   return home(value,u'data:img/jpeg;base64,'+data64.decode('utf-8'))
   return


@app.route('/', methods=['GET','POST'])
def home(value="AWord123",img=""):
  print(str(value))
  return '''<!DOCTYPE html>
<html>
<head>
<title>BARCODE</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
        color : white;
    font-style: oblique;
}
.info {
        background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
        background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width: 80%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;
}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
        padding-top:50px;
}
</style>
</head>
<body>
<div class="info">
<h1>BARCODE</h1>
<p>Returns the Image BARCODE out of the given message </p>
<p class="api_proc">API Call Format :
<span>http://localhost:5000/barcode?value=AWord123</span></p>
</div>
<div class="workarea">

<form action="/barcode">
  <label for="message">Barcode Content</label><br>
  <input type="text" id="value" name="value" value="'''+value+'''"><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">Barcode</label><br>
  <img src="'''+img+'''"></img></form>
</div>

</body>
</html>
'''


if __name__ == "__main__":
   app.run()
```

Screens :



BARCODE

Returns the Image BARCODE out of the given message

API Call Format : *http://localhost:5000/barcode?value=AWord123*

Barcode Content

AWord123

Submit

Barcode

Ex.No : 7

Question :

Generate a one-time password (OTP) in numbers, alphabet and alphanumeric (Note: OTP Size should be varied.)

Solution :

Python :

```
from flask import Flask, request
from flask_restful import Resource, Api, reqparse
import math, random

app = Flask(__name__)


#common error handler
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404

#---------------Tranpose--------------------
@app.route('/otp', methods=['GET'])
def otpGen():

  length = int(request.args['length'])
  typeOtp = int(request.args['typeotp'])
  if (typeOtp == 0):
    typeContents = "1234567890"
  elif (typeOtp == 1):
    typeContents = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
  else:
    typeContents =
"1234567890abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"

  genOTP = ""

  for i in range(length) :
      genOTP += typeContents[math.floor(random.random() * len(typeContents))]

  return home(str(length),str(typeOtp),genOTP)


@app.route('/', methods=['GET','POST'])
def home(len='4', typeotp='0',value=""):
  return '''<!DOCTYPE html>
<html>
<head>
<title>OTP</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
       color : white;
    font-style: oblique;
}
.info {
       background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
       background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width: 80%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
       padding-top:50px;
}
</style>
</head>
<body>
<div class="info">
<h1>CAPTCHA</h1>
<p>Returns the OTP of the given length and type </p>
```

```
<p class="api_proc">API Call Format :
<span>http://localhost:5000/otp?length=4&typeotp=</span></p>
</div>
<div class="workarea">

<form action="/otp">
  <label for="message">OTP Length</label><br>
  <input type="text" id="length" name="length" value="'''+len+'''"><br>
  <label for="message">Type( 0 - Numeric, 1 - Alphabets, 2 -
Alphanumeric)</label><br>
  <input type="text" id="typeotp" name="typeotp" value="'''+typeotp+'''"><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">OTP</label><br>
  <input type="text" id="response" name="response" value="'''+value+'''"><br>
</form>
</div>

</body>
</html>

'''

if __name__ == '__main__':
  app.run()
```

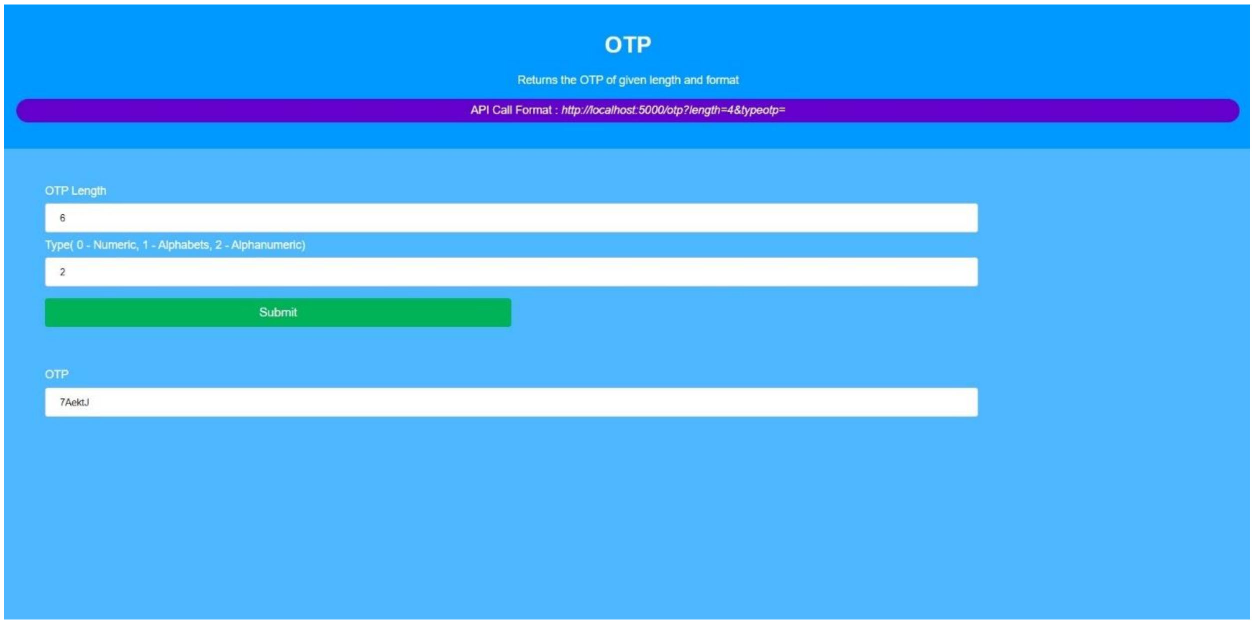
JavaScript :

```
const express = require('express')
const app = express()
const bp = require('body-parser')

app.use(bp.text())

app.get('/otp', (req, res) => {

    var length = parseInt(req.query.length)
    var typeOfOtp = parseInt(req.query.typeotp)

    if (typeOfOtp == 0)
        typeContents = "1234567890"
    else if (typeOfOtp == 1)
        typeContents = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
    else
        typeContents = "1234567890abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQ
RSTUVWXYZ"

    var genOTP = "",
        i;
    console.log(typeOfOtp, typeContents)
    for (i = 0; i < length; i++) {
        genOTP += typeContents[Math.floor(Math.random() * typeContents.length
)]
    }

    return res.send(genOTP)
})

//start
app.listen(3000)
```

PHP :

```
<!DOCTYPE html>
<html><head>
    <title>OTP</title>
    <style>
        body {
            background-color: #4db8ff;
            text-align: center;
            color: white;
            font-family: Arial, Helvetica, sans-serif;
        }

        span {
            color: white;
            font-style: oblique;
        }

        .info {
            background-color: #0099ff;
            padding: 20px;
            margin: -10px -10px 0 -10px;
        }

        .err_proc {
            background-color: red;
            padding: 7px;
            border-radius: 20px;
        }
        .api_proc {
            background-color: #6600cc;
            padding: 7px;
            border-radius: 20px;
        }

        .workarea {
            text-align: left;
            padding: 50px;
        }

        input[type=text] {
            width: 80%;
            padding: 12px 20px;
            margin: 8px 0;
            display: inline-block;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        input[type=submit] {
            width: 40%;
            background-color: #00b359;
            color: white;
            padding: 10px 20px;
            margin: 8px 0;
            border: none;
            border-radius: 4px;
            cursor: pointer;
            font-size: 17px;
        }

        input[type=submit]:hover {
            background-color: #00994d;
        }

        .response_form {
            padding-top: 50px;
        }
    </style>

</head>
<body>
<?php
    $typeContent = "";
    $otpContent = "";
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $length = $_POST['length'];
        $otpType = $_POST['typeotp'];

        if (empty($length)) {
            echo "<p class='err_proc'>Invalid Length</p>";
        }
        if($otpType == 0){
            $typeContent = "1234567890";
        } elseif($otpType == 1){
            $typeContent = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVW
XYZ";
        }elseif($otpType == 2){
            $typeContent = "1234567890abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLM
NOPQRSTUVWXYZ";
        }


        if (empty($typeContent)) {
            echo "<p class='err_proc'>Invalid Type</p>";
        }
        else{
            for ($i = 1; $i <= $length; $i++) {
                $otpContent .= substr($typeContent, (rand()%(strlen($typeCont
ent))), 1);
            }
        }



    }
?>

    <div class="info">
        <h1>OTP</h1>
        <p>Returns the OTP of desired length</p>
        <p class="api_proc">API Call Format : <span>http://localhost:5000/otp
?length=4&typeotp=</span></p>
    </div>
    <div class="workarea">
        <form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP
_SELF']);?>">
            <label for="message">OTP Length</label><br>
            <input type="text" id="length" name="length" value=""><br>
            <label for="message">Type( 0 - Numeric, 1 - Alphabets, 2 - Alphan
umeric)</label><br>
            <input type="text" id="typeotp" name="typeotp" value=""><br>
            <input type="submit" value="Submit">
        </form>
        <form class='response_form'>
           <label for='response'>OTP</label><br>
           <input type='text' id='response' name='response' value='<?php echo
 $otpContent; ?>'>
           <br>
         </form>
    </div>

</body>
</html>
```

Screens :



OTP

Returns the OTP of given length and format

API Call Format : *http://localhost:5000/otp?length=4&typeotp=*

OTP Length

6

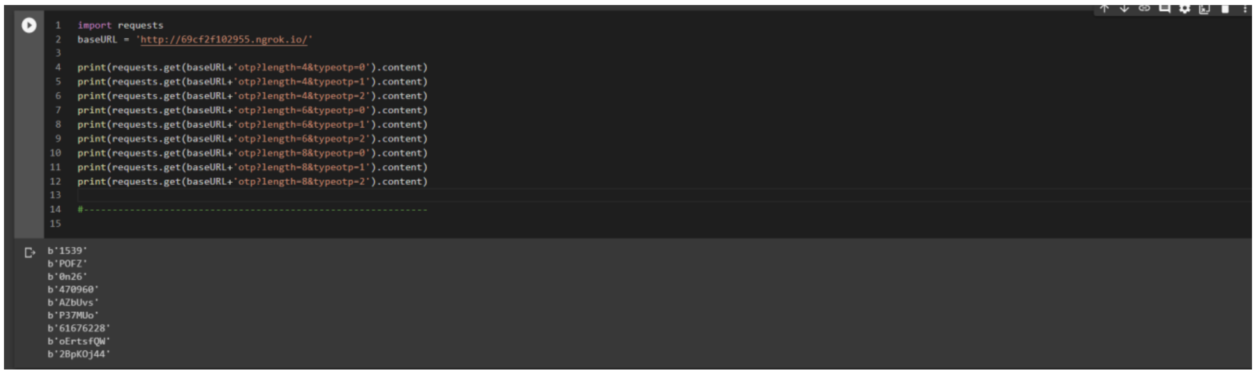
Type(0 - Numeric, 1 - Alphabets, 2 - Alphanumeric)

2

OTP

7AektJ

Test Report :



```python
import requests
baseURL = 'http://69cf2f102955.ngrok.io/'

print(requests.get(baseURL+'otp?length=4&typeotp=0').content)
print(requests.get(baseURL+'otp?length=4&typeotp=1').content)
print(requests.get(baseURL+'otp?length=4&typeotp=2').content)
print(requests.get(baseURL+'otp?length=6&typeotp=0').content)
print(requests.get(baseURL+'otp?length=6&typeotp=1').content)
print(requests.get(baseURL+'otp?length=6&typeotp=2').content)
print(requests.get(baseURL+'otp?length=8&typeotp=0').content)
print(requests.get(baseURL+'otp?length=8&typeotp=1').content)
print(requests.get(baseURL+'otp?length=8&typeotp=2').content)

#-----------------------------------------------------------

```

```
b'1539'
b'POFZ'
b'0n26'
b'470960'
b'AZbUvs'
b'P37MUo'
b'61676228'
b'oErtsfQW'
b'2BpKOj44'
```

Ex.No : 8

Question :

Generate a Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) for the given string.

Solution :

Python :

```
import random
import numpy as np
from PIL import Image, ImageFont,ImageDraw
import glob
import string
import cv2
import os,io
from io import BytesIO
import base64
from flask_ngrok import run_with_ngrok
from flask import Flask, request, Response, make_response

app = Flask(__name__)
run_with_ngrok(app)  # Start ngrok when app is run


@app.route('/getcaptcha', methods=['GET'])
def getCaptcha():

        text = request.args['value']
        # Setting up the canvas
        size = random.randint(10,50)
        length = len(text)
        img = np.zeros(((size*2)+5, length*size, 3), np.uint8)
        img_pil = Image.fromarray(img+255)

        # Drawing text and lines
        fontsPath = r"C:\Windows\Fonts"
        fonts = glob.glob(fontsPath+"\\ari*.ttf")
        font = ImageFont.truetype("Roboto-Regular.ttf", size)
        draw = ImageDraw.Draw(img_pil)
        draw.text((5, 10), text, font=font,
                  fill=(random.randint(0,255), random.randint(0,255),
random.randint(0,255)))
        draw.line([(random.choice(range(length*size)),
random.choice(range((size*2)+5)))
            ,(random.choice(range(length*size)),
random.choice(range((size*2)+5)))]
          , width=1, fill=(random.randint(0,255), random.randint(0,255),
random.randint(0,255)))

        # Adding noise and blur
        img = np.array(img_pil)
        thresh = random.randint(1,5)/100
        for i in range(img.shape[0]):
            for j in range(img.shape[1]):
                rdn = random.random()
                if rdn < thresh:
                    img[i][j] = random.randint(0,123)
                elif rdn > 1-thresh:
                    img[i][j] = random.randint(123,255)
        img =
cv2.blur(img,(int(size/random.randint(5,10)),int(size/random.randint(5,10))))

        #Displaying image
        #cv2.imshow(f"{text}", img)
        #cv2.waitKey()
        #cv2.destroyAllWindows()
        #cv2.imwrite(f"{os.getcwd()}\{text}.png", img) #if you want to save
the image
        retval, buffer = cv2.imencode('.png', img)
        response = Response(buffer.tobytes(), mimetype="image/png")
        return home(text,pil2datauri(buffer.tobytes()))

def pil2datauri(img):
    #converts PIL image to datauri
    data = BytesIO()
    # img.save(data, "JPEG")
    data64 = base64.b64encode(img)
    return u'data:img/jpeg;base64,'+data64.decode('utf-8')

@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>Page not found.</p>", 404


@app.route('/', methods=['GET','POST'])
def home(value="AWord123",img=""):
  print(str(value))
  return '''<!DOCTYPE html>
<html>
<head>
<title>CAPTCHA</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
        color : white;
    font-style: oblique;
}
.info {
        background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
        background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width: 80%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
       padding-top:50px;
}
</style>
</head>
<body>
<div class="info">
<h1>CAPTCHA</h1>
<p>Returns the Image CAPTCHA out of the given message </p>
<p class="api_proc">API Call Format :
<span>http://localhost:5000/getcaptcha?value=AWord123</span></p>
</div>
<div class="workarea">

<form action="/getcaptcha">
  <label for="message">CAPTCHA Content</label><br>
  <input type="text" id="value" name="value" value="'''+value+'''"><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">CAPTCHA</label><br>
  <img src="'''+img+'''"></img></form>
</div>
```

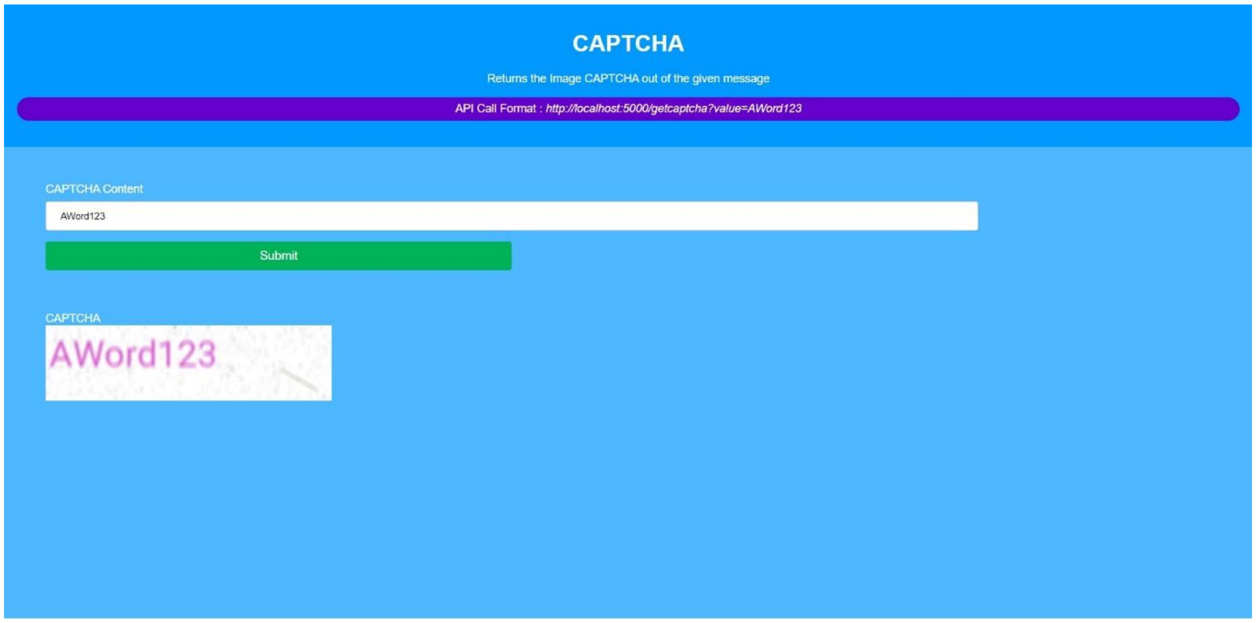
```
</body>
</html>
'''


if __name__ == "__main__":
   app.run()
```

PHP :

```
<!DOCTYPE html>
<html>
<head>
<title>CAPTCHA</title>
<style>
body {
  background-color: #4db8ff;
  text-align: center;
  color: white;
  font-family: Arial, Helvetica, sans-serif;
}
span {
    color : white;
    font-style: oblique;
}
.info {
    background-color : #0099ff;
    padding : 20px;
    margin: -10px -10px 0 -10px;
}
.api_proc {
    background-color: #6600cc;
    padding:7px;
    border-radius:20px;
}
.workarea {
text-align : left;
padding : 50px;
}

input[type=text] {
  width: 80%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;

}
input[type=submit] {
  width: 40%;
  background-color:   #00b359;
  color: white;
  padding: 10px 20px;
  margin: 8px 0;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size:17px;
}

input[type=submit]:hover {
  background-color: #00994d;
}
.response_form {
    padding-top:50px;
}
</style>
</head>
<body>

<?php
    $value =  "";$image = "";
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $capText = $_POST['value'];
        $capImage = imagecreatetruecolor(120, 56);
        $bg = imagecolorallocate($capImage, 245, 245, 237);
        $fg = imagecolorallocate($capImage, 132, 47, 161);
        imagefill($capImage, 0, 0, $bg);
        imagestring($capImage, rand(1, 9), rand(1, 9), rand(1, 9), $capText,
$fg);

        define('Root', dirname(__FILE__));
        $file = Root . $capText . '.png';
        imagepng($capImage,$file);
        if ($capImage) {
        ob_start();
        imagepng($capImage);
        $imgData=ob_get_clean();

        $image = '<img src="data:image/png;base64,'.base64_encode($imgData).'
" />';

    }
        $value = $capText;
        imagedestroy($capImage);

    }
?>

<div class="info">
<h1>CAPTCHA</h1>
<p>Returns the Image CAPTCHA out of the given message </p>
<p class="api_proc">API Call Format : <span>http://localhost:5000/getcaptcha?
value=AWord123</span></p>
```

```
</div>
<div class="workarea">

<form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF'])
;?>">
  <label for="message">CAPTCHA Content</label><br>
  <input type="text" id="value" name="value" value='<?php echo $value?>'><br>
  <input type="submit" value="Submit">
</form>
<form class="response_form">
  <label for="response">CAPTCHA</label><br>
  <?php echo $image?>
</div>

</body>
</html>
```

Screens :



CAPTCHA

Returns the Image CAPTCHA out of the given message

API Call Format : *http://localhost:5000/getcaptcha?value=AWord123*

CAPTCHA Content

AWord123