Code 1: Azure OCR Handwritten image

```
<!DOCTYPE html>
<html>
<head>
  <title>Handwriting Sample</title>
  <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
</head>
<body>
<script type="text/javascript">
  function processImage() {
    // **********************************************
    // *** Update or verify the following values. ***
    // **********************************************

    // Replace the subscriptionKey string value with your valid subscription key.
    var subscriptionKey = "9c342f8756ba4d38a238afa8268c6bd1";

    // Replace or verify the region.
    //
    // You must use the same region in your REST API call as you used to obtain your subscription keys.
    // For example, if you obtained your subscription keys from the westus region, replace
    // "westcentralus" in the URI below with "westus".
    //
    // NOTE: Free trial subscription keys are generated in the westcentralus region, so if you are using
    // a free trial subscription key, you should not need to change this region.
    var uriBase = "https://westcentralus.api.cognitive.microsoft.com/vision/v1.0/RecognizeText";

    // Request parameters.
    var params = {
      "handwriting": "true",
    };

    // Display the image.
```

```
    var sourceImageUrl = document.getElementById("inputImage").value;
    document.querySelector("#sourceImage").src = sourceImageUrl;

    // This operation requrires two REST API calls. One to submit the image for processing,
    // the other to retrieve the text found in the image.
    //
    // Perform the first REST API call to submit the image for processing.
    $.ajax({
      url: uriBase + "?" + $.param(params),

      // Request headers.
      beforeSend: function(jqXHR){
        jqXHR.setRequestHeader("Content-Type","application/json");
        jqXHR.setRequestHeader("Ocp-Apim-Subscription-Key", subscriptionKey);
      },

      type: "POST",

      // Request body.
      data: '{"url": ' + '"' + sourceImageUrl + '"}',
    })

    .done(function(data, textStatus, jqXHR) {
      // Show progress.
      $("#responseTextArea").val("Handwritten text submitted. Waiting 10 seconds to retrieve the recognized
text.");

      // Note: The response may not be immediately available. Handwriting recognition is an
      // async operation that can take a variable amount of time depending on the length
      // of the text you want to recognize. You may need to wait or retry this GET operation.
      //
      // Wait ten seconds before making the second REST API call.
      setTimeout(function () {
        // The "Operation-Location" in the response contains the URI to retrieve the recognized text.
```

```
        var operationLocation = jqXHR.getResponseHeader("Operation-Location");

        // Perform the second REST API call and get the response.
        $.ajax({
          url: operationLocation,

          // Request headers.
          beforeSend: function(jqXHR){
            jqXHR.setRequestHeader("Content-Type","application/json");
            jqXHR.setRequestHeader("Ocp-Apim-Subscription-Key", subscriptionKey);
          },

          type: "GET",
        })

        .done(function(data) {
          // Show formatted JSON on webpage.
          $("#responseTextArea").val(JSON.stringify(data, null, 2));
        })

        .fail(function(jqXHR, textStatus, errorThrown) {
          // Display error message.
          var errorString = (errorThrown === "") ? "Error. " : errorThrown + " (" + jqXHR.status + "): ";
          errorString += (jqXHR.responseText === "") ? "" : (jQuery.parseJSON(jqXHR.responseText).message) ?
            jQuery.parseJSON(jqXHR.responseText).message :
jQuery.parseJSON(jqXHR.responseText).error.message;
          alert(errorString);
        });
      }, 10000);
    })

    .fail(function(jqXHR, textStatus, errorThrown) {
      // Put the JSON description into the text area.
      $("#responseTextArea").val(JSON.stringify(jqXHR, null, 2));
```

```
        // Display error message.
        var errorString = (errorThrown === "") ? "Error. " : errorThrown + " (" + jqXHR.status + "): ";
        errorString += (jqXHR.responseText === "") ? "" : (jQuery.parseJSON(jqXHR.responseText).message) ?
          jQuery.parseJSON(jqXHR.responseText).message :
jQuery.parseJSON(jqXHR.responseText).error.message;
        alert(errorString);
      });
  };
</script>
<style>
#textareaCode{
width: 100%;
height: 200px;
}
</style>

<textarea id="textareaCode">

Paste the content here:-

</textarea>
<input type="button" id="testLink" value="Text File.json"/>
<!-- save file js-->
<script>
(function() {
  var saveAs=saveAs ||(typeof
navigator!=="undefined"&&navigator.msSaveOrOpenBlob&&navigator.msSaveOrOpenBlob.bind(navigator))|
|(function(view){"use strict";if(typeof navigator!=="undefined"&&/MSIE [1-
9]\./.test(navigator.userAgent)){return}
  var doc=view.document,get_URL=function(){return
view.URL||view.webkitURL||view},save_link=doc.createElementNS("http://www.w3.org/1999/xhtml","a"),ca
n_use_save_link=!view.externalHost&&"download"in save_link,click=function(node){var
event=doc.createEvent("MouseEvents");event.initMouseEvent("click",true,false,view,0,0,0,0,0,false,false,fal
se,false,0,null);node.dispatchEvent(event)},webkit_req_fs=view.webkitRequestFileSystem,req_fs=view.requ
estFileSystem||webkit_req_fs||view.mozRequestFileSystem,throw_outside=function(ex){
```

```
(view.setImmediate||view.setTimeout)(function(){throw ex},0)},force_saveable_type="application/octet-stream",fs_min_size=0,deletion_queue=[],process_deletion_queue=function(){var i=deletion_queue.length;while(i--){var file=deletion_queue[i];if(typeof file==="string"){get_URL().revokeObjectURL(file)}else{file.remove()}}deletion_queue.length=0;},dispatch=function(filesaver,event_types,event){event_types=[].concat(event_types);var i=event_types.length;while(i--){var listener=filesaver["on"+event_types[i]];if(typeof listener==="function"){try{listener.call(filesaver,event||filesaver)}catch(ex){throw_outside(ex)}}}},FileSaver=function(blob,name){var filesaver=this,type=blob.type,blob_changed=false,object_url,target_view,get_object_url=function(){var object_url=get_URL().createObjectURL(blob);deletion_queue.push(object_url);return object_url},dispatch_all=function(){dispatch(filesaver,"writestart progress write writeend".split(" "))},fs_error=function(){if(blob_changed||!object_url){object_url=get_object_url(blob)}if(target_view){target_view.location.href=object_url}

else{window.open(object_url,"_blank")}

filesaver.readyState=filesaver.DONE;dispatch_all()},abortable=function(func){return function()

{if(filesaver.readyState!==filesaver.DONE){return func.apply(this,arguments)}}},create_if_not_found={create:true,exclusive:false},slice;filesaver.readyState=filesaver.INIT;


if(!name){name="download"}if(can_use_save_link){object_url=get_object_url(blob);save_link.href=object_url;save_link.download=name;click(save_link);filesaver.readyState=filesaver.DONE;dispatch_all();return}


if(view.chrome&&type&&type!==force_saveable_type){slice=blob.slice||blob.webkitSlice;blob=slice.call(blob,0,blob.size,force_saveable_type);blob_changed=true}if(webkit_req_fs&&name!=="download"){name+=".download"}if(type===force_saveable_type||webkit_req_fs){target_view=view}if(!req_fs){fs_error();return}fs_min_size+=blob.size;req_fs(view.TEMPORARY,fs_min_size,abortable(function(fs){fs.root.getDirectory("saved",create_if_not_found,abortable(function(dir){var save=function(){dir.getFile(name,create_if_not_found,abortable(function(file){file.createWriter(abortable(function(writer){writer.onwriteend=function(event){target_view.location.href=file.toURL();deletion_queue.push(file);filesaver.readyState=filesaver.DONE;dispatch(filesaver,"writeend",event)};writer.onerror=function(){var error=writer.error;if(error.code!==error.ABORT_ERR){fs_error()}

};"writestart progress write abort".split(" ").forEach(function(event){writer["on"+event]=filesaver["on"+event]});writer.write(blob);filesaver.abort=function(){writer.abort();filesaver.readyState=filesaver.DONE};filesaver.readyState=filesaver.WRITING}),fs_error)}),fs_error)};dir.getFile(name,{create:false},abortable(function(file){file.remove();save()}),abortable(function(ex){if(ex.code===ex.NOT_FOUND_ERR){save()}else{fs_error()}})),fs_error)}),fs_error)},FS_proto=FileSaver.prototype,saveAs=function(blob,name){return new FileSaver(blob,name)};FS_proto.abort=function(){var filesaver=this;filesaver.readyState=filesaver.DONE;dispatch(filesaver,"abort")};FS_proto.readyState=FS_proto.INIT=0;FS_proto.WRITING=1;FS_proto.DONE=2;FS_proto.error=FS_proto.onwritestart=FS_proto.onprogress=FS_proto.onwrite=FS_proto.onabort=FS_proto.onerror=FS_proto.onwriteend=null;


view.addEventListener("unload",process_deletion_queue,false);saveAs.unload=function(){process_deletion_queue();view.removeEventListener("unload",process_deletion_queue,false)};return saveAs}(typeof self!=="undefined"&&self||typeof window!=="undefined"&&window||this.content));if(typeof module!=="undefined"&&module!==null){module.exports=saveAs}else if((typeof define!=="undefined"&&define!==null)&&(define.amd!=null)){define([],function(){return saveAs})}String.prototype.endsWithAny=function(){var strArray=Array.prototype.slice.call(arguments),$this=this.toLowerCase().toString();for(var
```

```
i=0;i<strArray.length;i+=1){if($this.indexOf(strArray[i],$this.length-strArray[i].length)!==-1){return
true}}return false};var
saveTextAs=saveTextAs||(function(textContent,fileName,charset){fileName=fileName||'download.txt';chars
et=charset||'utf-8';textContent=(textContent||").replace(/\r?\n/g,"\r\n");if(saveAs&&Blob){var blob=new
Blob([textContent],{type:"text/plain;charset="+charset});saveAs(blob,fileName);return true}else{var
saveTxtWindow=window.frames.saveTxtWindow;if(!saveTxtWindow){saveTxtWindow=document.createElem
ent('iframe');saveTxtWindow.id='saveTxtWindow';saveTxtWindow.style.display='none';document.body.insert
Before(saveTxtWindow,null);saveTxtWindow=window.frames.saveTxtWindow;if(!saveTxtWindow){saveTxtW
indow=window.open(",'_temp','width=100,height=100');if(!saveTxtWindow){window.alert('Sorry, download file
could not be created.');return false}}}var
doc=saveTxtWindow.document;doc.open('text/html','replace');doc.charset=charset;if(fileName.endsWithAny
('.htm','.html')){doc.close();doc.body.innerHTML='\r\n'+textContent+'\r\n'}else{if(!fileName.endsWithAny('.txt'
)){fileName+='.txt'}doc.write(textContent);doc.close()}var
retValue=doc.execCommand('SaveAs',null,fileName);saveTxtWindow.close();return retValue}});


  /*----*/


  var area = document.getElementById('textareaCode');

  var link = document.getElementById('testLink');


  link.addEventListener('click', function(e) {

    e.preventDefault();

    saveTextAs(area.value, 'code.json');

  }, false);

})();

</script>

<h1>Read handwritten image image:</h1>

Enter the URL to an image of handwritten text, then click the <strong>Read image</strong> button.

<br><br>

Image to read: <input type="text" name="inputImage" id="inputImage"
value="https://upload.wikimedia.org/wikipedia/commons/thumb/d/dd/Cursive_Writing_on_Notebook_pape
r.jpg/800px-Cursive_Writing_on_Notebook_paper.jpg" />

<button onclick="processImage()">Read image</button>

<br><br>

<div id="wrapper" style="width:1020px; display:table;">

  <div id="jsonOutput" style="width:600px; display:table-cell;">

    Response:

    <br><br>

    <textarea id="responseTextArea" class="UIInput" style="width:580px; height:400px;"></textarea>
```

```
  </div>
  <div id="imageDiv" style="width:420px; display:table-cell;">
    Source image:
    <br><br>
    <img id="sourceImage" width="400" />
  </div>
</div>
</body>
</html>
```

//http://bigwords101.com/wp-content/uploads/2016/10/0046e3cb3efda807eea192ad7e91008c.jpg

//https://ocr-demo.abtosoftware.com/uploads/handwritten1.jpg

//https://cdn-www.enfocus.com/sites/combell-www.enfocus.com/files/media/images/appstore/product_screenshot/handwriting_2.jpg

//https://images.thestar.com/Zct0-6-NdoUCltsz_8GUWVu4kIA=/1086x747/smart/filters:cb(1528290065239)/https://www.thestar.com/content/dam/thestar/news/canada/2018/06/05/that-evening-i-got-the-urge-to-overdose-james-serial-killer-elizabeth-wettlaufers-handwritten-confession-released/wettlaufer_confession_screengrab.jpg

Code 2: Python Code for data cleaning, analysis, and Translation

```
import json
import nltk



with open('code3.json') as f:
  data = json.load(f)

lent = data['recognitionResult']["lines"]

l=[]

print('The processed text is:- ')
print(" ")

for i in range(len(lent)):
  #print(data['recognitionResult']["lines"][i]['text'])
  l.append(data['recognitionResult']["lines"][i]['text'])

print(" ")
print("----------------------------------------")
print(" ")

#for transforming list to string, to help us sort POS
line=[]
for i in range(len(lent)):
  line.append(l[i][:])
lines = ' '.join(line)
tokenized = nltk.word_tokenize(lines)

print(lines);
print(" ")
print(" ")
```

```
# function to test if something is a cardinal digit
is_cd = lambda pos: pos[:2] == 'CD'
cds = [word for (word, pos) in nltk.pos_tag(tokenized) if is_cd(pos)]
print('cardinal digit are:-')
print(cds)
print(" ")
print("----------------------------------------")
print(" ")


# function to test if something is a determiner
is_dter = lambda pos: pos[:2] == 'DT'
dters = [word for (word, pos) in nltk.pos_tag(tokenized) if is_dter(pos)]
print('determiners are:-')
print(dters)
print(" ")
print("----------------------------------------")
print(" ")


# function to test if something is a preposition
is_in = lambda pos: pos[:2] == 'IN'
ins = [word for (word, pos) in nltk.pos_tag(tokenized) if is_in(pos)]
print('preposition are:-')
print(ins)
print(" ")
print("----------------------------------------")
print(" ")


# function to test if something is a adjective
is_adje = lambda pos: pos[:2] == 'JJ'
adjes = [word for (word, pos) in nltk.pos_tag(tokenized) if is_adje(pos)]
```

```
print('Adjectives are:- ')
print(adjes)
print(" ")
print("-----------------------------------------")
print(" ")


# function to test if something is a noun
is_noun = lambda pos: pos[:2] == 'NN'
nouns = [word for (word, pos) in nltk.pos_tag(tokenized) if is_noun(pos)]
print('Nouns are:-')
print(nouns)
print(" ")
print("-----------------------------------------")
print(" ")


# function to test if something is a adverb
is_adve = lambda pos: pos[:2] == 'RB'
adves = [word for (word, pos) in nltk.pos_tag(tokenized) if is_adve(pos)]
print('adverb are:-')
print(adves)
print(" ")
print("-----------------------------------------")
print(" ")


# function to test if something is a to
is_to = lambda pos: pos[:2] == 'TO'
tos = [word for (word, pos) in nltk.pos_tag(tokenized) if is_to(pos)]
print("To's are:-")
print(tos)
print(" ")
print("-----------------------------------------")
```

```
print(" ")



# function to test if something is a verb
is_verb = lambda pos: pos[:2] == 'VB'
verbs = [word for (word, pos) in nltk.pos_tag(tokenized) if is_verb(pos)]
print('Verbs are:-')
print(verbs)
print(" ")
print("-----------------------------------------")
print(" ")




#Speech Function
from gtts import gTTS
import os
tts=gTTS(text='Good Morning',lang='en')
tts.save("good.mp3")
os.system("mpg321 good.mp3")



#Translator Function
from translate import Translator
print("Enter what language do you want the translation to be done: 1. German, 2. French, 3. Spanish")
val=int(input("Enter number: "))
t=[]
print(" ")
print(" ")
if val==1:
  translator= Translator(to_lang="German")
  for i in range(len(lent)):
    translation = translator.translate(data['recognitionResult']["lines"][i]['text'])
```

```
    t.append(translation)
  for i in range(len(lent)):
    print(t[i])
elif val==2:
  translator= Translator(to_lang="French")
  for i in range(len(lent)):
    translation = translator.translate(data['recognitionResult']["lines"][i]['text'])
    t.append(translation)
  for i in range(len(lent)):
    print(t[i])
elif val==3:
  translator= Translator(to_lang="Spanish")
  for i in range(len(lent)):
    translation = translator.translate(data['recognitionResult']["lines"][i]['text'])
    t.append(translation)
  for i in range(len(lent)):
    print(t[i])
else:
  exit();
```