TECNOLOGICO NACIONAL DE MEXICO

INSTITUTO TECNOLOGICO DE MORELIA

"José María Morelos y Pavón"

Búsqueda Y Almacenamiento De Datos

Ferreira Escutia Rogelio

Proyecto BAD

Political Analytics

Carlos Jahir Castro Cázares
Giovanni Hasid Martínez Reséndiz
Jaime Isai Velazquez Aguilar

Semestre Sep - Ene 2020 - 2021

02 de febrero de 2021

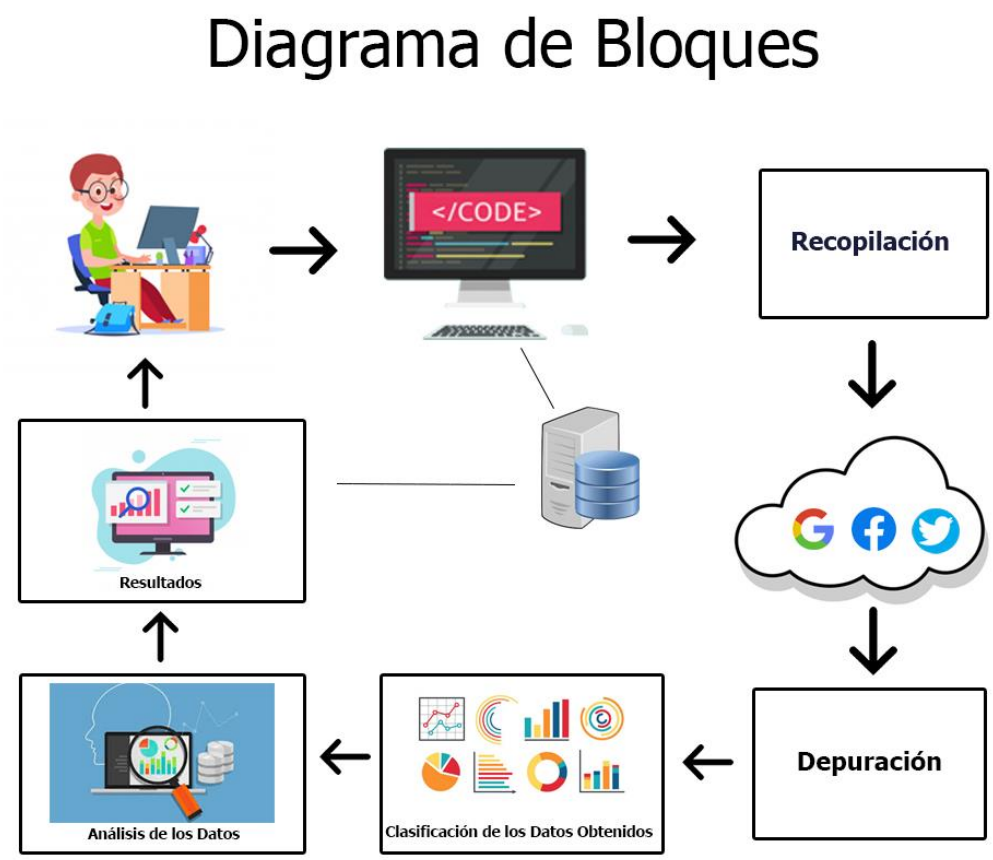
Descripción del proyecto.

Political Analytics se desarrolla en el área de la política, en las elecciones por venir para predecir quien será el gobernador de Michoacán, el objetivo es obtener información relevante de los candidatos electorales a través de las redes sociales, con ayuda de Apis y web scrapping identificando cuantas personas lo siguen.

La intención es mostrar de manera gráfica como estos resultados van cambiando conforme a los días y poder hacer una correlación de los datos y por lo tanto un pronóstico con mayor acierto.

Diagrama a bloques del sistema.

Nuestro sistema usa un modelo cliente – servidor, donde el cliente pide datos al servidory este busca la informacion en internet, la depura y analiza esta misma para guardarla en la base de datos y para mostrar los resultados el cliente pide al servidor los datos guardados en la base de datos y los regresa para visualizarlos en la web.



Estructura de las bases de datos y tablas empleadas.

Las tablas fueron creadas en MongoDB por lo cual no tienen una relación directa entre ellas a no ser SQL.

Candidatos

En esta tabla aguardamos la información de los candidatos, la estructura es:

- id
- nombre (Nombre del candidato)
- url_facebook (Pagina de Facebook)
- url_twitter (Pagina de Twitter)

Facebook

En esta tabla se guardan los datos que se recolectaron de Facebook con web scraping, de cada candidato en la fecha cuando se consultaron.

- Likes (Numero de likes)
- Seguidores (Número de seguidores)
- fecha-hora (Fecha de cuando se consultó)
- id_candidato (Id del candidato que pertenece)

Twitter

En esta tabla se guardan los datos que se recolectaron de Twitter con su API, de cada candidato en la fecha cuando se consultaron.

- Seguidores (Número de seguidores)
- Favoritos (Numero de Favoritos)
- fecha-hora (Fecha de cuando se consultó)
- id_candidato (Id del candidato que pertenece)

Puntajes_totales

En esta tabla se guardaron todos los puntajes totales que son la suma de los likes, seguidores de Facebook y twitter y favoritos, en la fecha cuando se consultaron.

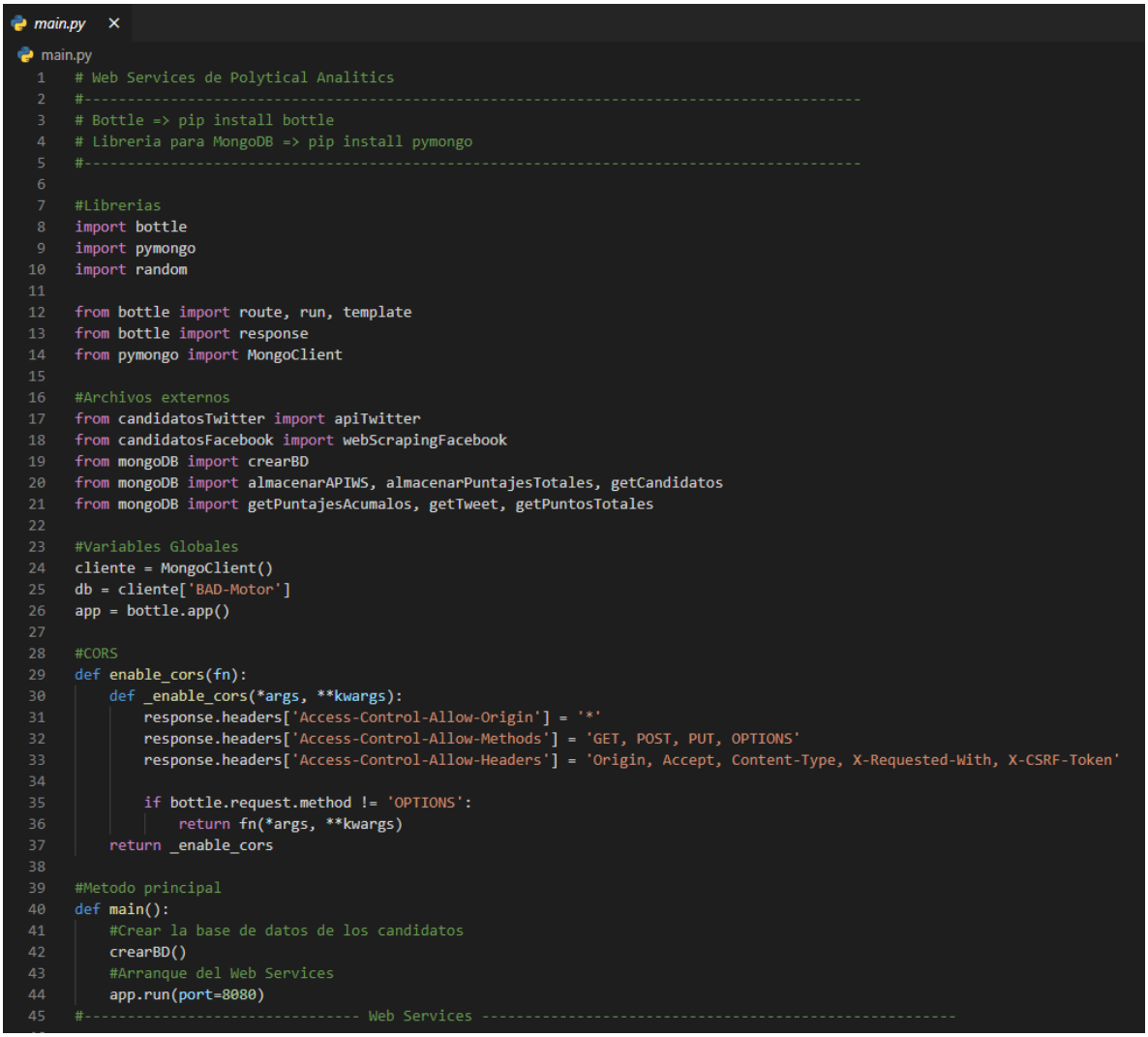
- Puntaje (Puntaje total)
- fecha-hora (Fecha de cuando se consultó)
- id_candidato (Id del candidato que pertenece)

Todos los códigos empleados.

Servidor

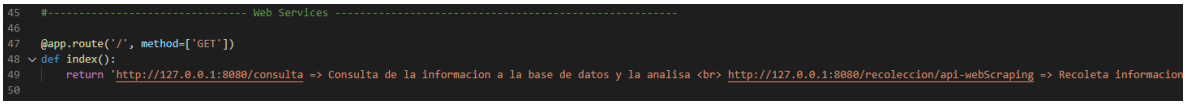
Main.py

En esta parte se implementan las librerías necearías para el web Services y que funcione la conexión con mongo DB y las funciones que necesitamos de otros archivos, la configuración para configurar los CORS y el método para crear la BD y arrancar el servidor en el puerto 8080.



```python
# Web Services de Polytical Analitics
#-----------------------------------------------------------------------------------------
# Bottle => pip install bottle
# Libreria para MongoDB => pip install pymongo
#-----------------------------------------------------------------------------------------

#Librerias
import bottle
import pymongo
import random

from bottle import route, run, template
from bottle import response
from pymongo import MongoClient

#Archivos externos
from candidatosTwitter import apiTwitter
from candidatosFacebook import webScrapingFacebook
from mongoDB import crearBD
from mongoDB import almacenarAPIWS, almacenarPuntajesTotales, getCandidatos
from mongoDB import getPuntajesAcumalos, getTweet, getPuntosTotales

#Variables Globales
cliente = MongoClient()
db = cliente['BAD-Motor']
app = bottle.app()

#CORS
def enable_cors(fn):
    def _enable_cors(*args, **kwargs):
        response.headers['Access-Control-Allow-Origin'] = '*'
        response.headers['Access-Control-Allow-Methods'] = 'GET, POST, PUT, OPTIONS'
        response.headers['Access-Control-Allow-Headers'] = 'Origin, Accept, Content-Type, X-Requested-With, X-CSRF-Token'

        if bottle.request.method != 'OPTIONS':
            return fn(*args, **kwargs)
    return _enable_cors

#Metodo principal
def main():
    #Crear la base de datos de los candidatos
    crearBD()
    #Arranque del Web Services
    app.run(port=8080)
#------------------------------ Web Services ----------------------------------------------
```

Ruta principal del Web Services que muestra la información, de las rutas en este.



```python
#------------------------------ Web Services ----------------------------------------------------

@app.route('/', method=['GET'])
def index():
    return 'http://127.0.0.1:8080/consulta => Consulta de la informacion a la base de datos y la analisa <br> http://127.0.0.1:8080/recoleccion/api-webScraping => Recoleta informacion

```

Esta es la función para la ruta que muestra información que existe en la base de datos y se crea el json que regresara.

```python
@route('/consulta')
@enable_cors
def consulta_info():
    response.headers['Content-type'] = 'application/json'

    #JSON a enviar
    datos = {
        '1': {
            'likes_F': 0,
            'seguidores_F': 0,
            'seguidores_T': 0,
            'favoritos_T': 0,
            'twett': '',
            'puntajes': []
        },
        '2': {
            'likes_F': 0,
            'seguidores_F': 0,
            'seguidores_T': 0,
            'favoritos_T': 0,
            'twett': '',
            'puntajes': []
        },
        '3': {
            'likes_F': 0,
            'seguidores_F': 0,
            'seguidores_T': 0,
            'favoritos_T': 0,
            'twett': '',
            'puntajes': []
        },
    }
```

En esta parte extraemos el número de likes, seguidores y favoritos de twitter y de Facebook, los sumamos para obtener como va su progreso de cada candidato.

```
83      #Analizando los datos de likes, seguidores y favoritos de la Base de Datos
84      puntajes = getPuntajesAcumalos()
85      for i in range(3):
86          i += 1
87          print('Analizando los datos del candidato '+ str(i) + '. . .')
88
89          puntosFB = puntajes[str(i)]['fb']
90          for puntoFB in puntosFB:
91              datos[str(i)]['likes_F'] += int(puntoFB['likes'])
92              datos[str(i)]['seguidores_F'] += int(puntoFB['seguidores'])
93
94          puntosTW = puntajes[str(i)]['tw']
95          for puntoTW in puntosTW:
96              datos[str(i)]['seguidores_T'] += int(puntoTW['seguidores'])
97              datos[str(i)]['favoritos_T'] += int(puntoTW['favoritos'])
98
99          datos[str(i)]['twett'] = getTweet(puntoTW['_id'])
```

Sigue sacando todos los puntajes totales de la base de datos y con estos puntos se hace una regresión lineal, para predecir cómo serán los resultados en 7 días más y regresamos el json al cliente.

```
100
101     #Analizando puntajes totales
102     puntosTotales = getPuntosTotales()
103     for i in range(3):
104         i += 1
105
106         #Variables regresion lineal
107         #-------------------------------------------
108         # x => 'orden'
109         # y => 'puntaje'
110         #-------------------------------------------
111
112         n = len(puntosTotales[str(i)])
113         xy = 0
114         x = 0
115         y = 0
116         x2 = 0
117
118         for punto in puntosTotales[str(i)]:
119             xy += int(punto['orden']) * int(punto['puntaje'])
120             x += int(punto['orden'])
121             y += int(punto['puntaje'])
122             x2 += int(punto['orden']) * int(punto['orden'])
123
124         #Formulas de regresion lineal
125         #-------------------------------------------
126         # a = (n*Exy - ExEy) / (nEx2 - (Ex)2)
127         # b = (Ey - aEx) / n
128         # y = ax + b
129         #-------------------------------------------
130         a = ((n*xy) - (x*y)) / ((n*x2) - (x*x))
131         b = (y - (a*x)) / n
132
133         for j in range(7):
134             j += 1
135             y = int((a*(n+j)) + b)
136             puntosTotales[str(i)].append({
137                 'puntaje': y,
138                 'fecha-hora': '+'+str(j)+' dia',
139                 'orden': n+j
140             })
141
142     datos['1']['puntajes'] = puntosTotales['1']
143     datos['2']['puntajes'] = puntosTotales['2']
144     datos['3']['puntajes'] = puntosTotales['3']
145
146     #Retornar JSON con el analisis de los datos
147     return datos
```

En esta función es la ruta cuando recolectamos datos de internet, con la API de Twitter y haciendo Web Scraping de páginas de Facebook, los almacenamos en la base de datos, sumamos los puntajes obtenidos en las paginas y los guardamos en los puntajes totales, obtenemos la información de los candidatos y lo juntamos con la información recolectada para regresarla al cliente en json.

```python
@route('/recoleccion/api-webScraping')
@enable_cors
def data_mine():
    response.headers['Content-type'] = 'application/json'

    #Peticiones a API y Web Scraping
    datosTwitter = apiTwitter()
    datosFacebook = webScrapingFacebook()

    #Almacenar los datos en la base de datos
    datos = almacenarAPIWS(datosTwitter, datosFacebook)

    #Analizar el puntaje total de los candidatos
    totales = [0 ,0 ,0]
    for i in range(3):
        totales[i] = int(datos['api'][i]['seguidores']) + int(datos['api'][i]['favoritos']) + int(datos['webScraping'][i]['likes']) + int(datos['webScraping'][i]['seguidores'])
    datos = almacenarPuntajesTotales(totales, datos)

    #Optener informacion de los candidatos
    datos['candidatos'] = getCandidatos()

    #Quitar _id
    for ele in datos['api']:
        ele['_id'] = ''
    for ele in datos['webScraping']:
        ele['_id'] = ''
    for ele in datos['puntajesTotales']:
        ele['_id'] = ''

    #Retornar JSON con los datos extraidos
    return datos
#---------------------------------------------------------------------------------------------
if __name__=="__main__":
    main()
```

MongoDB.py

En este archivo hacemos la conexión con la base de datos y guardamos o consultamos la información almacenada.

En esta parte importamos la librería para la conexión, y con la función creamos la base de datos, con la información de los candidatos.

```python
#Contiene funciones para conectar con la base de datos
import pymongo
import time

from pymongo import MongoClient
from datetime import datetime


cliente = MongoClient()
db = cliente['Political_Analytics']

#Crea la base de datos
def crearBD():
    print('Conectando con Mongo DB . . .')
    coleccion = db['candidatos']

    #Crear tabla de candidatos
    if coleccion.count_documents({}) == 0:
        print('Creando candidatos . . .')
        candidatos = [
            {'id': 1, 'nombre': 'Raúl Morón Orozco', 'url_facebook': 'https://www.facebook.com/raulmoronorozco/?ref=page_internal', 'url_twitter': 'https://twitter.com/raulmoronO?r
            {'id': 2, 'nombre': 'Carlos Herrera Tello', 'url_facebook': 'https://www.facebook.com/CarlosHerreraSI/?ref=page_internal', 'url_twitter': 'https://twitter.com/carlosher
            {'id': 3, 'nombre': 'Juan Antonio Magaña de la Mora', 'url_facebook': 'https://www.facebook.com/juanAntonioMdelaMora/community/?ref=page_internal', 'url_twitter': 'http
        ]
        coleccion.insert_many(candidatos)
    return
```

En esta función almacenamos los datos de la API y del Web Scraping en las tablas de Facebook y Twitter.

```python
#Almacenar resultados de la API y WS en la BD
def almacenarAPIWS(api, webscraping):
    print('Conectando y Almacenando con Mongo DB . . .')
    datosAlmacenados = {
        'api': [],
        'webScraping': [],
        'fechaHora': time.strftime("%d/%m/%y %H:%M:%S")
    }

    #Almacenar datos del api de Twitter
    print('Registrando datos de api de twitter . . .')
    coleccion = db['twitter']
    for datosTW in api:
        documento = {'seguidores': datosTW[1], 'favoritos': datosTW[2], 'twitt': datosTW[3], 'fecha-hora': datosAlmacenados['fechaHora'], 'id_candidato': datosTW[0]}
        datosAlmacenados['api'].append(documento)
        coleccion.insert_one(documento)

    #ALmacenar datos del web scraping de facebook
    print('Registrando datos del web scraping de facebook . . .')
    coleccion = db['facebook']
    for datosFB in webscraping:
        documento = {'likes': datosFB[1], 'seguidores': datosFB[2], 'fecha-hora': datosAlmacenados['fechaHora'], 'id_candidato': datosFB[0]}
        datosAlmacenados['webScraping'].append(documento)
        coleccion.insert_one(documento)

    return datosAlmacenados
```

Almacenamos los puntajes totales en la base de datos en la tabla con el mismo nombre.

```python
def almacenarPuntajesTotales(totales, datos):
    print('Conectando y Almacenando con Mongo DB . . .')
    coleccion = db['puntajes_totales']

    #Almacenar Datos
    puntajesTotales = [
        {'puntaje': totales[0], 'fecha-hora': datos['fechaHora'], 'id_candidato': 1},
        {'puntaje': totales[1], 'fecha-hora': datos['fechaHora'], 'id_candidato': 2},
        {'puntaje': totales[2], 'fecha-hora': datos['fechaHora'], 'id_candidato': 3},
    ]
    datos['puntajesTotales'] = puntajesTotales
    coleccion.insert_many(puntajesTotales)

    return datos
```

Obtenemos los datos del candidato.

```python
def getCandidatos():
    print('Conectando y Almacenando con Mongo DB . . .')
    coleccion = db['candidatos']
    #Optener a los candidatos
    candidatos = []
    for candidato in coleccion.find({}, {'_id': 0}):
        candidatos.append(candidato)
    return candidatos
```

Obtenemos los puntajes acumulado de la tabla de Facebook y twitter.

```python
def getPuntajesAcumalos():
    print('Conectando y Almacenando con Mongo DB . . .')
    datos = {
        '1': {
            'tw': [],
            'fb': []
        },
        '2': {
            'tw': [],
            'fb': []
        },
        '3': {
            'tw': [],
            'fb': []
        },
    }

    print('Extrayendo datos almacenados . . .')
    #Datos de Twitter
    coleccion = db['twitter']
    for i in (range(3)):
        i += 1
        for puntajes in coleccion.find({'id_candidato': str(i)}, {'twitt':0, 'fecha-hora': 0, 'id_candidato': 0}):
            datos[str(i)]['tw'].append(puntajes)

    #Datos de Facebook
    coleccion = db['facebook']
    for i in (range(3)):
        i += 1
        for puntajes in coleccion.find({'id_candidato': str(i)}, {'_id': 0, 'fecha-hora': 0, 'id_candidato': 0}):
            datos[str(i)]['fb'].append(puntajes)
    return datos
```

Obtenemos los puntajes totales de cada candidato y ponemos un índice llamado orden para hacer la regresión lineal.

```python
def getPuntosTotales():
    print('Conectando y Almacenando con Mongo DB . . .')
    coleccion = db['puntajes_totales']
    datos = {
        '1': [],
        '2': [],
        '3': [],
    }

    #Extrayendo puntajes totales
    print('Extrayendo datos almacenados . . .')
    for i in (range(3)):
        i += 1
        orden = 1
        for puntos in coleccion.find({'id_candidato': i}, {'_id': 0, 'id_candidato': 0}):
            puntos['orden'] = orden
            datos[str(i)].append(puntos)
            orden += 1
    return datos
```

CandidatosFacebook.py

Explicando la parte para obtener información de la red social de Facebook, primero se deben importar dos librerias que es la de request y beautifulsoup4, ya que la API que nos brinda fb se encuentra un poco limitada al acceso, por lo cual se concluyo en hacer WebScrapping a Facebook.

Esta parte será mandado a llamar desde otro código rutinariamente, ya que la función de este es regresar un array de los datos obtenidos de fb.

Iniciando a describir el código se tienen definidos las url's de los 3 candidatos que se tienen hasta el momento, donde obtendrá los likes y los seguidores basándonos en la estructura que tiene Facebook para mostrar la información, una vez tomada dichos datos se retornarán al código principal, donde se insertarán en la base de datos

```python
# pip install beautifulsoup4
#pip install request

import requests
from bs4 import BeautifulSoup

def webScrapingFacebook():
    print('Web Scraping de paguinas estaticas de facebook . . .')
    urls=[]
    urls.append("https://www.facebook.com/pg/raulmoronorozco/community/?ref=page_internal")
    urls.append("https://www.facebook.com/pg/CarlosHerreraSi/community/?ref=page_internal")
    urls.append("https://www.facebook.com/pg/juanAntonioMdelaMora/community/?ref=page_internal")

    candidatos=[]
    cont=0

    for url in urls:
        datos=[]
        cont=cont+1

        datos.append(str(cont))

        #Conectamos con la pagina
        print('\tConectando con la paguina: '+url+' . . .')
        try:
            pag = requests.get(url)
            html = BeautifulSoup(pag.content, 'html.parser')
            conexion = True
        except:
            print('\tNo puede conectarse a la página '+url)
            conexion = False

        if conexion:
            print('\tAnalizando paguina: '+url+' . . .')
            comunidad = html.find('div', class_='clearfix _ikh _3xol')
```

```
            #Obtener los likes de la página
            seccion = (comunidad.find_all('div', class_='_3xom'))
            seccion_likes=seccion[0].text
            likes=seccion_likes.replace("\xa0mil", "000")

            seccion_seguidores=seccion[1].text
            seguidores=seccion_seguidores.replace("\xa0mil", "000")

            likes=likes.replace(".", "")
            seguidores=seguidores.replace(".", "")


            datos.append(likes)
            datos.append(seguidores)

            candidatos.append(datos)
        else:
            datos.append("")
            datos.append("")
            candidatos.append(datos)
    return candidatos
```

CandidatosTwitter.py

El objetivo dentro de este programa Python es consumir la api de Twitter para poder obtener información específica de cada candidato a la gobernatura de Michoacán, para poder realizar esto hacemos uso de la librería tweepy, lo primero que hacemos es autenticarnos con nuestros 4 tokens que se muestran de la línea 8 a 11. Una vez hecho lo anterior obtenemos toda la información de cada candidato que nos proporciona la api, después por cada candidato declaramos un arreglo e insertamos para cada uno almacenamos sus followers y favoritos recibidos, una vez hecho esto retornamos un arreglo que contiene los 3 arreglos de los candidatos.

```
1   #API de Twitter => pip install tweepy
2   import tweepy
3   import json
4
5   def apiTwitter():
6       print('Accediendo a la api de Twitter . . .')
7       #autenticación poner sus propias claves :)
8       consumer_key="oU3eRnB0ENoGq124quINXdrAZ"
9       consumer_secret="2QMnOEuGbG6EUXccaOndksaTvT4LAmrKJ838bb2IuCZASonxCg"
10      access_token = "418197577-pXUdPpPC6NJDbmaAgQSpNRkvEMSt1xep5foYjBqk"
11      access_token_secret = "JxArOpeZg8B29dRsdnRGSQYWOtC1xeWe1DbrZULFNA6s0"
12
13      auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
14      auth.set_access_token(access_token, access_token_secret)
15
16      api = tweepy.API(auth, wait_on_rate_limit=True,
17      wait_on_rate_limit_notify=True)
18
19      data = api.get_user("CarlosHerreraSi")
20      dataRM = api.get_user("raulmoronO")
21      dataJM = api.get_user("Magana_DeLaMora")
22      Candidatos = []
23
24      #print(json.dumps(data._json, indent=2))
25      #Carlos Herrera
26      print('\tCarlos Herrera . . .')
27      CarlosHerrera = []
28      followers = str(data.followers_count)
29      noFavoritos = str(data.favourites_count)
30      uTweet = str(data.status.text)
31      CarlosHerrera.append("2")
32      CarlosHerrera.append(followers)
33      CarlosHerrera.append(noFavoritos)
34      CarlosHerrera.append(uTweet)
```

```
36      #Raúl Morón Orozco
37      print('\tRaúl Morón Orozco . . .')
38      RaulMoron = []
39      RMfollowers = str(dataRM.followers_count)
40      RMnoFavoritos = str(dataRM.favourites_count)
41      RMuTweet = str(dataRM.status.text)
42      RaulMoron.append("1")
43      RaulMoron.append(RMfollowers)
44      RaulMoron.append(RMnoFavoritos)
45      RaulMoron.append(RMuTweet)
46
47      #Juan Antonio Magaña de la Mora
48      print('\tJuan Antonio Magaña de la Mora . . .')
49      JuanMagania = []
50      JMfollowers = str(dataJM.followers_count)
51      JMnoFavoritos = str(dataJM.favourites_count)
52      JMuTweet = str(dataJM.status.text)
53      JuanMagania.append("3")
54      JuanMagania.append(JMfollowers)
55      JuanMagania.append(JMnoFavoritos)
56      JuanMagania.append(JMuTweet)
57
58      print('\tAnalizando datos . . .')
59      Candidatos.append(RaulMoron)
60      Candidatos.append(CarlosHerrera)
61      Candidatos.append(JuanMagania)
62
63      return Candidatos
```

Cliente

Documentos HTML, para mostrar los datos recolectados, que contiene a los candidatos y las graficas de progreso y comparación de puntajes.

Index.html

```html
<!DOCTYPE html>
<html lang="en" dir="ltr">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="description" content="">
  <meta name="author" content="TecNM Big Data CIH">
  <title>Candidatos a Gobernador</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <link href="css/styles.css" rel="stylesheet">
</head>

<body>

  <header>
    <nav class="navbar navbar-light bg-light fixed-top">
      <div class="container-fluid">
        <a class="navbar-brand" href="#">
          <img src="img/logo.png" alt="" width="50" height="34" class="d-inline-block align-top">
          Political Analytics
        </a>
      </div>
    </nav>
  </header>

  <div class="container Candidatos">
    <div class="row">
      <h2 class="tituloG">CANDIDATOS A LA GOBERNATURA DE MICHOACÁN</h2>
      <div class="col-lg-4">
        <img class="bd-placeholder-img rounded-circle" src="img/carlosHerrera.jpg" alt="foto de carlos herrera" width="140" height="140">
        <h3>Carlos Herrera Tello</h3>
        <div id="datosCH"></div>
        <img class="bd-placeholder-img partidoImg" src="img/CHPRD.png" alt="partido politico" width="70" height="70">
      </div><!-- /.col-lg-4 -->
      <div class="col-lg-4">
        <img class="bd-placeholder-img rounded-circle" src="img/raulMoron.jpg" alt="foto de raul moron" width="140" height="140">
        <h3>Raúl Morón Orozco</h3>
        <div id="datosRM"></div>
        <img class="bd-placeholder-img partidoImg" src="img/morenaRM.png" alt="partido politico" width="70" height="70">
      </div><!-- /.col-lg-4 -->
      <div class="col-lg-4">
        <img class="bd-placeholder-img rounded-circle" src="img/JuanMora.jpg" alt="foto de raul moron" width="140" height="140">
        <h3>Juan Antonio Magaña</h3>
        <div id="datosJM"></div>
        <img class="bd-placeholder-img partidoImg" src="img/partidoVerdeMora.png" alt="partido politico" width="70" height="70">
      </div>
    </div>
```

```html
    <div class="row graphics">
      <h2 class="tituloG">ESTADÍSTICAS EN TIEMPO REAL</h2>
      <div class="col-md-12 col-lg-6 col-sm-12" id="curve_chart" style="width: 800px ; height: 500px"></div>
      <div class="chart col-md-12 col-lg-6 col-sm-12" id="columnchart_values" ></div>
    </div>
    <br>
    <hr>
    <br>
    <div class="row">
      <div class="col-lg-12">
        <div class="text-center">
          <button type="button" class="btn btn-outline-primary" title="Consulta API y Web Scraping" id="btnRecoleccion">Recoleccion de Datos</button>
        </div>
      </div>
    </div>
  </div>

  <footer class="bg-light text-center mt-3">
    <br>
    <!-- Copyright -->
    <div class="text-center p-3" style="background-color: rgba(0, 0, 0, 0.2)">
      <p> © 2021 BAD</p>
      <p>Giovanni Martínez - Carlos Castro - Jaime Velázquez</p>
    </div>
    <!-- Copyright -->
  </footer>
</body>
<script src="js/bootstrap.bundle.min.js"></script>
<script src="https://code.jquery.com/jquery-3.5.1.min.js" integrity="sha256-9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0=" crossorigin="anonymous"></script>
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script src="js/principal.js"></script>
</html>
```

Principal.js

En esta función hacemos cuando el documento se haya cargado completamente hacemos una petición AJAX, al servidor la cual recibe el json con la información, la pone en su respectivo lugar dentro de index.html.

```
$(document).ready(function(){
    $.ajax({
        type: "get",
        url: "http://127.0.0.1:8080/consulta",
        data: {},
        dataType: "json",
        success: function (json) {
            console.log(json);

            //Carlos Herrera Tello
            let templateCH = `
            <p class="follows">Followers en Twitter: `+ json["1"].seguidores_T +`</p>
            <p class="follows">Favoritos en Twitter: `+ json["1"].favoritos_T+`</p>
            <p class="follows">Total de likes en Fb: `+ json["1"].likes_F +`</p>
            <p class="follows">Total de seguidores en Fb: `+ json["1"].seguidores_F +`</p>
            `;
            $("#datosCH").html(templateCH);

            //Raúl Morón
            let templateRM = `
            <p class="follows">Followers en Twitter: `+ json["2"].seguidores_T +`</p>
            <p class="follows">Favoritos en Twitter: `+ json["2"].favoritos_T+`</p>
            <p class="follows">Total de likes en Fb: `+ json["2"].likes_F +`</p>
            <p class="follows">Total de seguidores en Fb: `+ json["2"].seguidores_F +`</p>
            `;
            $("#datosRM").html(templateRM);

            //Juan mora
            let templateJM = `
            <p class="follows">Followers en Twitter: `+ json["3"].seguidores_T +`</p>
            <p class="follows">Favoritos en Twitter: `+ json["3"].favoritos_T+`</p>
            <p class="follows">Total de likes en Fb: `+ json["3"].likes_F +`</p>
            <p class="follows">Total de seguidores en Fb: `+ json["3"].seguidores_F +`</p>
            `;
            $("#datosJM").html(templateJM);
```

Llena la gráfica de puntos con los puntajes totales y la predicción a 7 días.

```
36
37          //Graphics
38          google.charts.load('current', {'packages':['corechart']});
39
40          /* Historial */
41          google.charts.setOnLoadCallback(drawChart);
42          function drawChart() {
43              let templateGraphics = ``;
44
45              let graficaPuntajes = [
46                  ['Día', 'Carlos Herrera', 'Raúl Morón', 'Juan Magaña'],
47              ];
48
49              let longitud = json["1"]["puntajes"].length;
50              for (let i = 0; i < longitud; i++) {
51                  let col = [
52                      json["1"]["puntajes"][i]['fecha-hora'],
53                      json["1"]["puntajes"][i]['puntaje'],
54                      json["2"]["puntajes"][i]['puntaje'],
55                      json["3"]["puntajes"][i]['puntaje']
56                  ];
57                  graficaPuntajes.push(col);
58              }
59
60              var data = google.visualization.arrayToDataTable(graficaPuntajes);
61
62              var options = {
63                  title: 'Presencia en redes sociales',
64                  curveType: 'function',
65                  legend: { position: 'bottom' },
66                  series: {
67                      0: { color: '#FFC300' },
68                      1: { color: '#C70039' },
69                      2: { color: '#0eb62f' },
70                      }
71              };
72              var chart = new google.visualization.LineChart(document.getElementById('curve_chart'));
73              chart.draw(data, options);
74          }
```

Llena la gráfica de barras que muestra la comparación entre los puntajes de los candidatos.

```
        /* Barras */
        google.charts.setOnLoadCallback(drawBarras);
        function drawBarras() {
            var data = google.visualization.arrayToDataTable([
            ["Element", "Density", { role: "style" } ],
            ["Carlos Herrera", (json["1"].seguidores_T + json["1"].favoritos_T + json["1"].likes_F + json["1"].seguidores_F), "#FFC300"],
            ["Raúl Morón", (json["2"].seguidores_T + json["2"].favoritos_T + json["2"].likes_F + json["2"].seguidores_F), "#C70039 "],
            ["Juan Magaña", (json["3"].seguidores_T + json["3"].favoritos_T + json["3"].likes_F + json["3"].seguidores_F), "#0eb62f"]
            ]);

            var view = new google.visualization.DataView(data);
            view.setColumns([0, 1,
                { calc: "stringify",
                    sourceColumn: 1,
                    type: "string",
                    role: "annotation" },
                2]);

            var optionsBarras = {
                title: "Cantidad de personas que los siguen",
                height: 500,
                bar: {groupWidth: "95%"},
                legend: { position: "none" },
            };
            var chartBarras = new google.visualization.ColumnChart(document.getElementById("columnchart_values"));
            chartBarras.draw(view, optionsBarras);
        }

        $(window).resize(function(){
            drawChart();
            drawBarras();
        });
    }
});
```

Función clic del botón para consultar nuevos datos del internet a través de la API y el Web Scraping de Facebook, para esto hacemos una petición AJAX, al servidor para hacer esta acción y nos regrese una respuesta, para mostrar un mensaje.

```
110
111     $('#btnRecoleccion').click(function (e) {
112         e.preventDefault();
113         $('#btnRecoleccion').attr("disabled", true);
114         $('#btnRecoleccion').text('Recolectando Datos');
115         $.ajax({
116             type: "get",
117             url: "http://127.0.0.1:8080/recoleccion/api-webScraping",
118             data: {},
119             dataType: "json",
120             success: function (response) {
121                 console.log(response);
122                 alert('Recoleccion de datos terminado, recarge la pagina')
123             }
124         });
125     });
126 });
127
```

Capturas de pantalla de la impresión de resultados.

Los resultados que son mostraran son los que recabamos en una semana de extraer datos del internet.

En la primera parte podemos ver a cada candidato, con su información y numero de seguidores de facebook y de twitter, así como el numero de likes de Facebook y de favoritos de Twitter.

Por último, vemos las gráficas, la primera es del avance en redes sociales que a tenido un candidato, y posteriormente la predicción a futuro de como estará en redes sociales, seguidamente una grafica que compara la presencia de estos en redes sociales.

Como podemos ver en la grafica de comparaciones, el posible futuro ganador será **Raúl morón.** Por superar a los demás en presencia en redes sociales.