```
import requests

import os

from bs4 import BeautifulSoup

from twilio.rest import Client

import yagmail

import time

import logging


URL_TO_MONITOR = "" #change this to the URL you want to monitor

DELAY_TIME = 15 # seconds


TWILIO_ACCOUNT_SID = "" # replace with your Account SID

TWILIO_AUTH_TOKEN = "" # replace with your Auth Token

TWILIO_PHONE_SENDER = "+12345678901" # replace with the phone number you registered in
twilio

TWILIO_PHONE_RECIPIENT = "+12345678901" # replace with your phone number


SENDING_EMAIL_USERNAME = "" # replace with the username of the gmail account you created
(e.g. "john.webmonitor" if the email is "john.webmonitor@gmail.com")

SENDING_EMAIL_PASSWORD = "" # replace with the password of the gmail account you created

RECIPIENT_EMAIL_ADDRESS = "" # replace with the email address that will receive the notification


def send_email_alert(alert_str):

    """Sends an email alert. The subject and body will be the same. """

    yagmail.SMTP(SENDING_EMAIL_USERNAME, SENDING_EMAIL_PASSWORD).send(

        RECIPIENT_EMAIL_ADDRESS, alert_str, alert_str)


def send_text_alert(alert_str):
    """Sends an SMS text alert."""
    client = Client(TWILIO_ACCOUNT_SID, TWILIO_AUTH_TOKEN)
    message = client.messages.create(
        to=TWILIO_PHONE_RECIPIENT,
        from_=TWILIO_PHONE_SENDER,
        body=alert_str)


def process_html(string):
    soup = BeautifulSoup(string, features="lxml")


    # make the html look good
    soup.prettify()


    # remove script tags
    for s in soup.select('script'):
        s.extract()


    # remove meta tags
    for s in soup.select('meta'):
        s.extract()


    # convert to a string, remove '\r', and return
    return str(soup).replace('\r', '')


def webpage_was_changed():
    """Returns true if the webpage was changed, otherwise false."""
    headers = {'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_1) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36',
    'Pragma': 'no-cache', 'Cache-Control': 'no-cache'}
    response = requests.get(URL_TO_MONITOR, headers=headers)

    # create the previous_content.txt if it doesn't exist
    if not os.path.exists("previous_content.txt"):
        open("previous_content.txt", 'w+').close()

    filehandle = open("previous_content.txt", 'r')
    previous_response_html = filehandle.read()
    filehandle.close()

    processed_response_html = process_html(response.text)

    if processed_response_html == previous_response_html:
        return False
    else:
        filehandle = open("previous_content.txt", 'w')
        filehandle.write(processed_response_html)
        filehandle.close()
        return True

def main():
    log = logging.getLogger(__name__)
    logging.basicConfig(level=os.environ.get("LOGLEVEL", "INFO"), format='%(asctime)s %(message)s')
    log.info("Running Website Monitor")
    while True:
        try:
            if webpage_was_changed():
                log.info("WEBPAGE WAS CHANGED.")
                send_text_alert(f"URGENT! {URL_TO_MONITOR} WAS CHANGED!")
                send_email_alert(f"URGENT! {URL_TO_MONITOR} WAS CHANGED!")
            else:
                log.info("Webpage was not changed.")
        except:
            log.info("Error checking website.")
        time.sleep(DELAY_TIME)


if __name__ == "__main__":
    main()
```
