const puppeteer = require('puppeteer');

const fs = require('fs');

const path = require('path');

async function login() {

  const browser = await puppeteer.launch({ headless: false, defaultViewport: null });

  const page = await browser.newPage();

  await page.goto('https://twitter.com/login');

  console.log('Please wait 20 seconds to log in...');

  // you have 60 seconds to log in

  await page.waitForTimeout(60000);

  const hashtags = await readHashtagsFromFile('hashtags.txt');

  // Loop to search and retweet

  while (true) {

    // Generate a random index to select a hashtag from the list

    const randomIndex = getRandomInt(0, hashtags.length);

    // Select a random hashtag from the list

    const hashtag = hashtags[randomIndex];

    // Search for the hashtag

    await page.goto(`https://twitter.com/search?q=${encodeURIComponent(`#${hashtag}`)}&src=typed\_query&f=top`);

    // Check if the search tab is accessible

    const searchTabExists = await page.waitForSelector('div[data-testid="SearchNavigation"]', { visible: true, timeout: 5000 })

      .then(() => true)

      .catch(() => false);

    if (!searchTabExists) {

      console.log('Search tab not accessible. Taking a screenshot and exiting...');

      await page.screenshot({ path: path.join(\_\_dirname, 'ScreenShots', 'failure.png') });

      break;

    }

    // Wait for the tweets to load

    await page.waitForSelector('div[data-testid="tweet"]', { visible: true });

    // Get the list of tweets

    const tweets = await page.$$('div[data-testid="tweet"]');

    // Choose a random tweet index from the top 30

    const randomTweetIndex = getRandomInt(0, Math.min(tweets.length, 30));

    // Retweet the chosen tweet

    const tweet = tweets[randomTweetIndex];

    await tweet.hover();

    await tweet.click('div[data-testid="retweet"]');

    console.log(`Retweeted a tweet with the hashtag #${hashtag}`);

  }

  // Close the browser after breaking the loop

  await browser.close();

}

function readHashtagsFromFile(filename) {

  return new Promise((resolve, reject) => {

    fs.readFile(filename, 'utf8', (err, data) => {

      if (err) {

        console.error(`Error reading hashtags from file: ${err}`);

        reject([]);

      } else {

        const hashtags = data.split('\n').map((line) => line.trim()).filter((line) => line !== '');

        resolve(hashtags);

      }

    });

  });

}

function getRandomInt(min, max) {

  return Math.floor(Math.random() \* (max - min) + min);

}

login();

Version 2:  
works for the most part, but forgets to retweet

const puppeteer = require('puppeteer');

const fs = require('fs');

const path = require('path');

async function login() {

  const browser = await puppeteer.launch({ headless: false, defaultViewport: null });

  const page = await browser.newPage();

  await page.goto('https://twitter.com/login');

  console.log('Please wait 40 seconds to log in...');

  // you have 60 seconds to log in

  await page.waitForTimeout(30000);

  const screenshotsFolderPath = path.join(\_\_dirname, 'screenshots');

  fs.mkdirSync(screenshotsFolderPath, { recursive: true });

  const hashtags = await readHashtagsFromFile('hashtags.txt');

  let counter = 0; // Counter to limit the number of iterations

  while (counter < 10) { // Adjust the number of iterations as needed

    try {

      // Loop to search and retweet

      for (const hashtag of hashtags) {

        // Search for the hashtag

        await page.goto(`https://twitter.com/search?q=${encodeURIComponent(`#${hashtag}`)}&src=typed\_query&f=top`);

        await page.screenshot({ path: path.join(screenshotsFolderPath, 'search.png') });

        // Wait for the tweets to load

        await page.waitForSelector('div[data-testid="tweet"]', { visible: true });

        await page.screenshot({ path: path.join(screenshotsFolderPath, 'tweets.png') });

        // Get the list of tweets

        const tweets = await page.$$('div[data-testid="tweet"]');

        // Choose a random tweet index from the top 30

        const randomTweetIndex = getRandomInt(0, Math.min(tweets.length, 30));

        // Capture a screenshot of the chosen tweet

        const chosenTweet = tweets[randomTweetIndex];

        await chosenTweet.screenshot({ path: path.join(screenshotsFolderPath, 'chosen\_tweet.png') });

        // Retweet the chosen tweet

        await chosenTweet.hover();

        await chosenTweet.click('div[data-testid="retweet"]');

        await page.screenshot({ path: path.join(screenshotsFolderPath, 'retweet.png') });

        // Wait for the retweet menu to appear

        await page.waitForSelector('div[data-testid="retweetConfirm"]', { visible: true });

        // Retweet the tweet

        const retweetButton = await page.$('div[data-testid="retweetConfirm"]');

        await retweetButton.click();

        console.log(`Retweeted a tweet with the hashtag #${hashtag}`);

        counter++; // Increment the counter

      }

    } catch (error) {

      console.error('An error occurred:', error);

    }

  }

  // Close the browser after breaking the loop

  await browser.close();

}

function readHashtagsFromFile(filename) {

  return new Promise((resolve, reject) => {

    fs.readFile(filename, 'utf8', (err, data) => {

      if (err) {

        console.error(`Error reading hashtags from file: ${err}`);

        reject([]);

      } else {

        const hashtags = data.split('\n').map((line) => line.trim()).filter((line) => line !== '');

        resolve(hashtags);

      }

    });

  });

}

function getRandomInt(min, max) {

  return Math.floor(Math.random() \* (max - min) + min);

}

login();