Rithvik reddy

Chezuba

FORM FILLER BOT REPORT

# FORM FILLER BOT:

# Pre-requisites:

* Knowledge of [Selenium Python.](https://selenium-python.readthedocs.io/index.html)
* Knowledge of AWS services such as S3, Lambda, SQS, ECR and ECS.
* [Boto3 library](https://aboto3.amazonaws.com/v1/documentation/api/latest/index.html).
* Docker.

# GitHub repo:

* <https://github.com/rithvik2211/Form-Filling-bot>
* Repository contains code for lambda function, docker file, form filler bot code, and edge driver.

# Updating code:

* Pull the repository from the link above.
* Make required changes in the form filler bot code.
* Download the latest version of the Stable Linux Edge driver from the [link](https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/) and place it in the current folder.
* Run the docker file. You will be able to see the image in the docker app. Run the image in a container to test it. You can verify the image running properly by looking in AWS.
* If the test is successful push the image into ECR using the steps below.

# Steps to push the docker image into ECR:

* Install AWS CLI from the [link](https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html).
* If it’s already installed. Type - aws configure. And give your access ID, secret access password, and region.
* If it fails delete the config file from C://../user/docke

Push commands

* aws ecr get-login-password --region eu-north-1 | docker login --username AWS --password-stdin\*\*\*\*\*\*\*.dkr.ecr.eu-north-1.amazonaws.com

#After the login is successful

* docker tag <image name>\*\*\*\*\*\*\*.dkr.ecr.eu-north-1.amazonaws.com/formfiller-container:latest
* docker push\*\*\*\*\*\*\*\*.dkr.ecr.eu-north-1.amazonaws.com/formfiller-container:latest

The image will be pushed into ECR, and the Image URI of the older version will be transferred to the new version. (Note: Use the same name as previous image)

# Working:

* Form filler bot 1st loads the page and waits 2 seconds for the page to load.
* Then find a link that has a substring “contact” in it (It is case insensitive.) and opens the link and waits 2 seconds for the page to load.
* Finds a “form” tag (HTML tag).
* There are “input” tags in the form tag. Input tags have attributes (identifiers). I have chosen “name” and” placeholder” to identify and fill in the details.

|  |  |  |
| --- | --- | --- |
| To fill ⬇️ / attributes ➡️ | Name | placeholder |
| Email | Mail | Email |
| First name | Fname, first | Fname |
| Last name | Lname, last | Lname |
| Full name | Name | Name |
| Subject | Subject | Subject, data-id = subject |
| Phone number |  | Phone |

\*\*fills the details in the order mentioned in the table

* It checks if it filled 1st name and last name if not it fills full name.
* Message to be filled is found by textarea tag.
* Submit button is then pressed and waits 4 seconds.

|  |  |  |
| --- | --- | --- |
|  | Type | Data-testid |
| Input tag | Submit |  |
| Button tag | Submit, buttonElement | buttonElement |

## Version 1 (local testing):

* Checks if the button is pressed and then takes a full-page screenshot and stores the screenshot in the ‘DATA’ folder and moves to the next link.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | Links | Form | Broken Link | Time Taken |

* Makes an Excel sheet schema:
* If it did not find a “form” tag it fills “no form” in the Form column else, it leaves it empty. If the link is not working, it fills “no page” in the broken link column else it is empty.

## Version 2 (AWS):

* Links are accessed through “get\_message” through access keys from SQS and the screenshot is stored in S3 through “upload\_fileobj” using the Boto3 API service. Make sure the access keys you use have permission to the resources.

# Problems:

* If the Contact page link was of a different format than a substring of “contact”. We cannot open the link.

Example: contact page and form are present in the link [www.xyz.com/apply-now](http://www.xyz.com/apply-now) or [www.xyz.com/connect](http://www.xyz.com/connect)

* Contact page is stored in blocks that need to be pressed to find the link.
* Form is on the main page not on the contact page.
* Form is sometimes present in “div” and “ul” tags.
* There are multiple contact pages. Could not open the one which had form.
* Could not fill WordPress forms. Reason not found.

# Results:

### Test with 500 forms:

Successfully filled= 50 forms

Screenshots captured = 315

Failed due to captcha = 125

Others failed due to not being able to fill one or 2 fields or fill all details but not being able to click submit button.

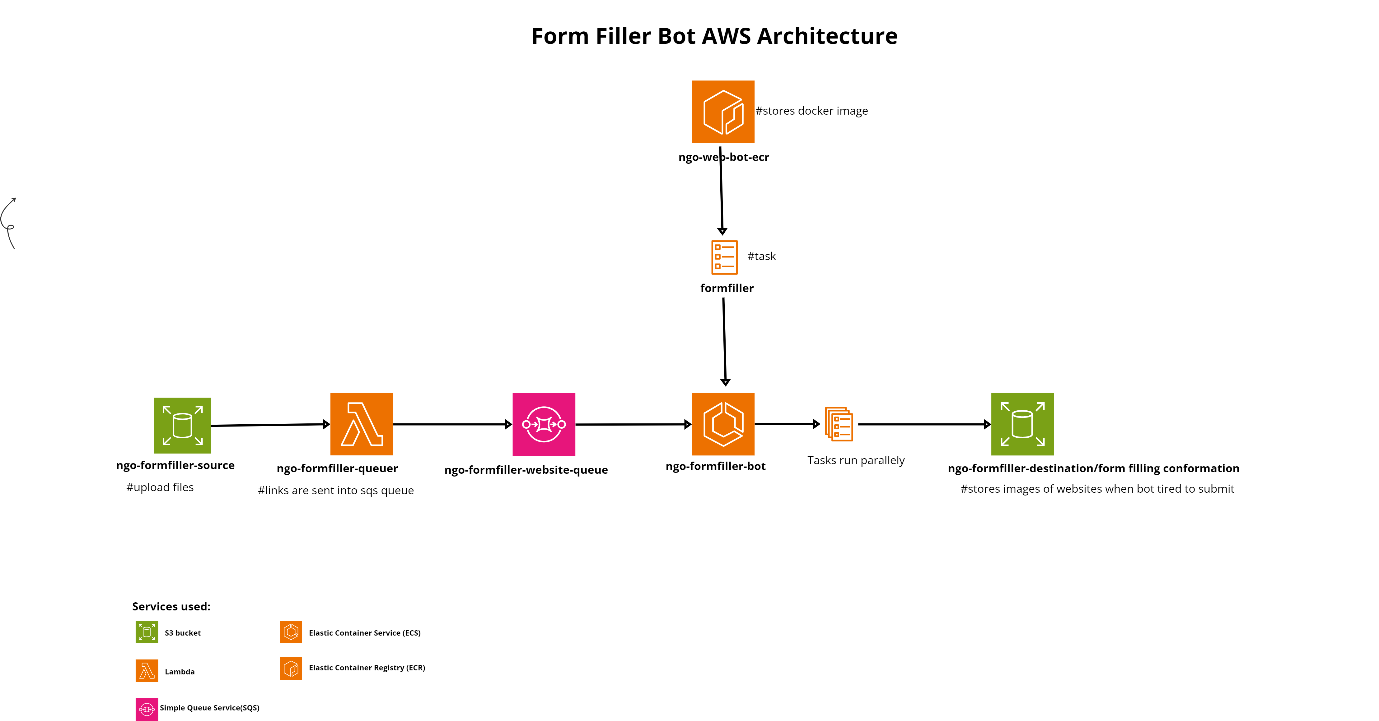
### Test with random 1000 links:

Found forms= 630

But in reality, I had to go through 1060 links to find 500 forms.

# Scaling:

Scaling is done using AWS cloud services.



1. Before uploading ‘.xlsx’ files make change the column name to “**Links**” where the URLs are present.
2. After uploading files Lambda function will be directly triggered sending all the URLs into the SQS queue.
3. We are processing links as tasks. If we update our container, do it with the same tag so that the ARN of the image would not change.
4. Refer user guide for using the form filler bot.

# Suggested improvements:

* Using DynomoDB metrics generated while form filling can be stored.
* Using Network Capture confirmation request might be captured which removes the usage destination bucket.
* To improve security access keys can be added as Environment variables.
* Improvements can be made to accept different file types.