Red Hat SRE Challenge: Dockerfile Sources

The challenge

As part of our compliance requirements, we want to make sure that we are building containers with trusted source images.

Your task is to build a tool that given a list of repositories, it identifies all the `Dockerfile` files inside each repository, extracts the image names from the `FROM` statement, and returns a `json` with the aggregated information for all the repositories.

You can find the details of the `FROM` command here:

https://docs.docker.com/engine/reference/builder/#from

The input will be provided as a URL pointing to a plaintext file. Each line will have two fields separated by a space:

- the https url of the github public repository
- the commit SHA to verify.

You can skip any line that doesn't match this pattern.

Example input:

https://gist.githubusercontent.com/jmelis/c60e61a893248244dc4fa12b946585c4/raw/25d39f67f2405330a6314cad64fac423a171162c/sources.txt

Example output:

```
{
  "data": {
    "https://github.com/app-sre/qontract-reconcile.git:30af65af14a2dce962df923446afff24dd8f123e": {
      "dockerfiles/Dockerfile": [
        "quay.io/app-sre/qontract-reconcile-base:0.2.1"
      ]
    },
    "https://github.com/app-sre/container-images.git:c260deaf135fc0efaab365ea234a5b86b3ead404": {
      "jiralert/Dockerfile": [
        "registry.access.redhat.com/ubi8/go-toolset:latest",
        "registry.access.redhat.com/ubi8-minimal:8.2"
      ],
      "qontract-reconcile-base/Dockerfile": [
        "registry.access.redhat.com/ubi8/ubi:8.2",
        "registry.access.redhat.com/ubi8/ubi:8.2",
        "registry.access.redhat.com/ubi8/ubi:8.2"
      ]
    }
  }
}
```

Things to take into account

- We expect the exercise to take around 4 hours. It could be more or less depending on your experience.
- We would like the codebase to be in either of the two languages most commonly used by our team: Python or Golang.
- Do not rush the exercise. Time to delivery will not be part of the evaluation criteria. We care about the quality of the code and making reasoned choices.
- **Execution time will be part of the evaluation criteria**. Considering n as the number of repositories to be processed, we expect an implementation with better time complexity than O(n).
- Invalid lines in the input file should be ignored.
- This should be production grade code.
- You can extend the return payload to include errors.

Deliverables

- URL to *private* GitHub repository with the code and the assignment PDF.
- A README.md file detailing your implementation and any additional features added.
- The Hiring Specialist will share the GitHub username of the reviewer with you. You will need to add them to the private repository with read access.

Bonus points

- Since we are a cloud-native team, we want to run this as a Kubernetes Job. If you already know kubernetes, that is excellent. If you don't, we will hugely value you taking the time to check out minikube and figuring out how to use `Jobs`. The list of repository urls should be provided to the Job with the `REPOSITORY_LIST_URL` environment variable, which should point at an url.
- Please feel free to implement any additional features that make this project more production ready. Do make sure to document them in the README.