Getting started

Your Serverless account

If you don't already have a Serverless account, go ahead and create one. Once you've created your Serverless account and you have logged into your account, you should see you org name in the far right of the navigation menu at the top of the page. Copy your org name and paste it on line 1 of your `serverless.yml` file. Also, while you're logged into your Serverless account, go ahead and click the "apps" navigation item and then click on the orange button that says "add app". Name your new app "jobtracker" and click on add app.

You'll also need to make sure you have the Serverless CLI downloaded. You can find the instructions to get that setup https://serverless.com/framework/docs/getting-started/.

Your AWS account

You will also need an AWS account. If you don't already have one, go ahead and create an account. Don't worry, this project is basically free to run on AWS.

Prerequisites

This project runs on Node.js, so you'll need to make sure you have Node.js installed on your computer. Once you have access to Node.js, go to your terminal and move into the project

```
cd /path/to/jobtracker
```

Now, we'll need to install our Node packages, so go ahead and run the following command in your terminal

```
npm install
```

Database setup

We're using MongoDB as a database for this project. You'll need to setup an account with a service like mLab or MongoDB Atlas for the serverless lambdas to be able to talk to the database. For testing purposes, you'll probably want to setup a MongoDB database locally on your computer as well.

When you get your accounts setup, we'll need to create a database named `jobtracker`. Once that database has been created, we'll want to create a collection inside of that database named `companies`. This is where your jobs data will be stored.

Now, for each of the companies you want to track, you'll need to add a document in your `companies` colletion for that company. You should just be able to add the name and everything else will be populated once you run the scraper for that company. So, add a new document that looks like this for example: `{"name": "workOS"}`. Then, when you run `serverless invoke -f getworkosjobs`, there shouldn't be an issue.

Usage

Command	**Description**
`serverless deploy`	Deploys all of your lambda functions to your AWS account.

`serverless deploy -f FUNCTION_NAME`	Deploys just one of your lambda functions to your AWS account.
`serverless invoke -f FUNCTION_NAME`	Invoke one of your lambda functions in the Serverless environment
`serverless invoke local -f FUNCTION_NAME`	Invoke one of your lambda functions locally on your machine

Cron configuration

Once you have deployed your lambda functions to your AWS account, you'll need to set these up to be invoked once per day so that you can get the new jobs daily. This will be handled in your AWS Console. For each of your lambda functions in your Lambda console, you'll need to click into them and then click on the "+ Add trigger" button in the Designer panel. From the "Select a trigger" dropdown, go ahead and choose the "Cloudwatch Events/Eventbridge" selection. Now, under the "Rule" dropdown, we're going to create a new rule for each function (you can only have 5 resources per rule, so it's easier to create a new rule for each resource - aka lambda function in this case). Give the new rule a name. I usually go with `FUNCTION_NAME_cron` (replace `FUNCTION_NAME` with the actual function name of each function). Now, we need to move down to the "Schedule expression" input field. Here's where you'll customize the schedule you want your functions to scrape the new jobs on. Here's an example of how I set the schedule for my functions: `cron(0 14 * * ? *)`. Here's a useful tool for generating cron expressions - crontab.guru. Remember to wrap your cron expression in `cron()` in the AWS console though.

Wrapping up

Once you've setup your Serverless and AWS accounts, downloaded and configured your source code, and deployed your project, and configured your cron settings, you should be off to the races!

Troubleshooting

If you're running into any issues

, feel free to reach out to me via email: parker@parkeragee.com.