JobScout

Mentors: Sohan Gharpure, David Mandelbaum
Group Members: Nessim Betesh, Owen Caplan, Adir Koenig, Sam Stolarov, Ilan Wurmann

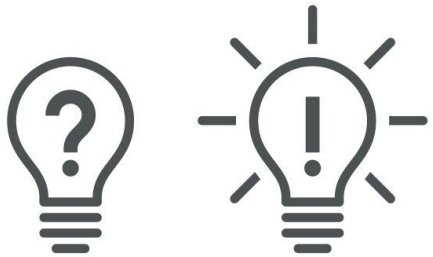
GitHub Repository: https://github.com/SM24-Industrial-Software-Dev/Job-Scout

JobScout Website: http://18.191.83.191:8501

Introduction

Problem: There are many websites for job searching, and it takes a lot of time for users to find every relevant job as well as to stay up to date on new job postings.

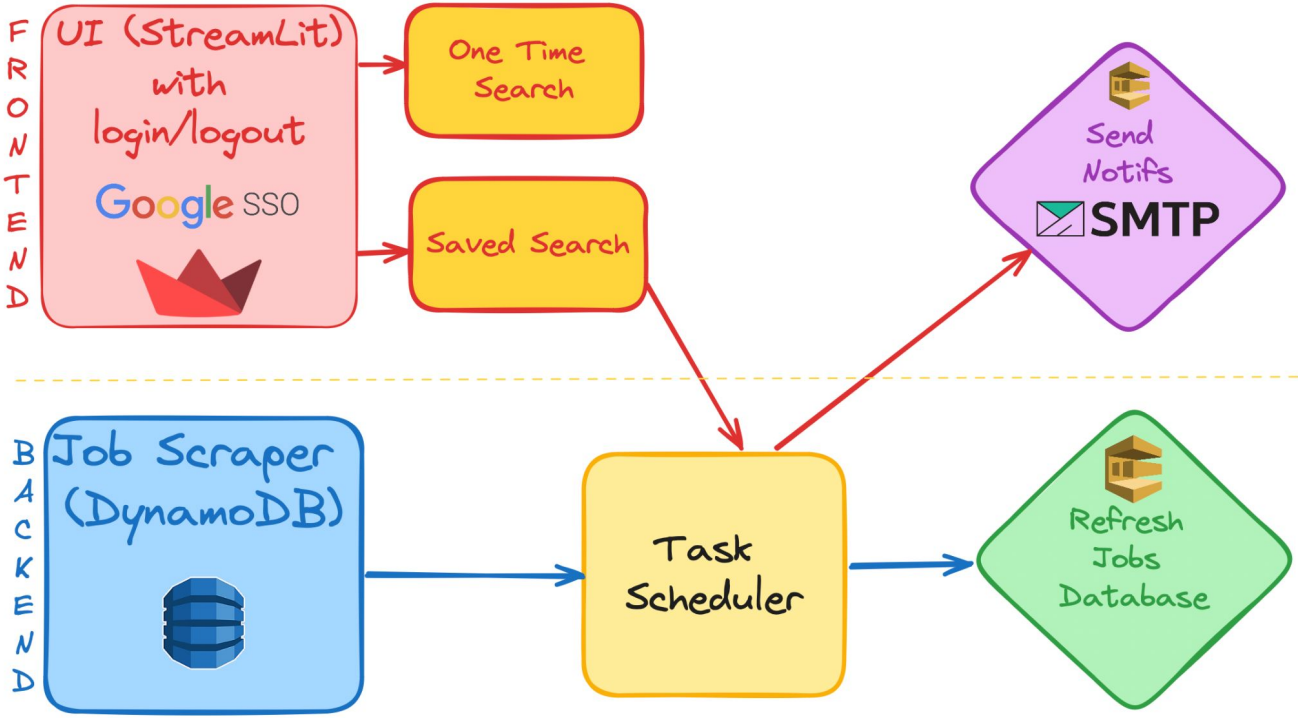
Solution: A website that aggregates jobs from various websites into one database and sends notifications with new job postings based on a user's job preferences.



Tools and Frameworks:

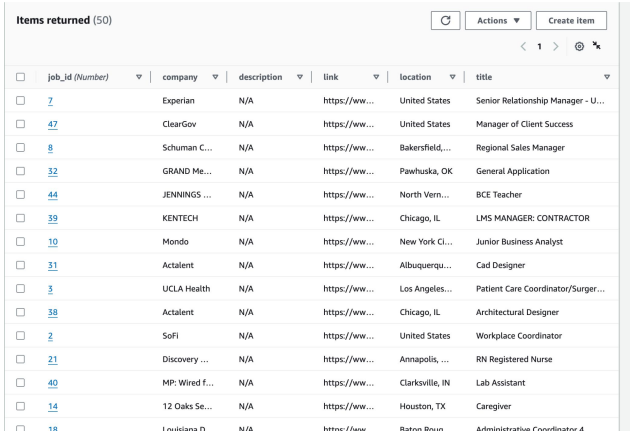
- **Python** for backend code development
- **Streamlit** for UI
- **FastAPI** for UI endpoints
- **AWS DynamoDB** for job database and task scheduling
- **Boto3** for Python interacting with DynamoDB
- **AWS SQS** for execution services
- **AWS EC2** for deployment
- **Flask** for Google login
- **SMTP** for email notification

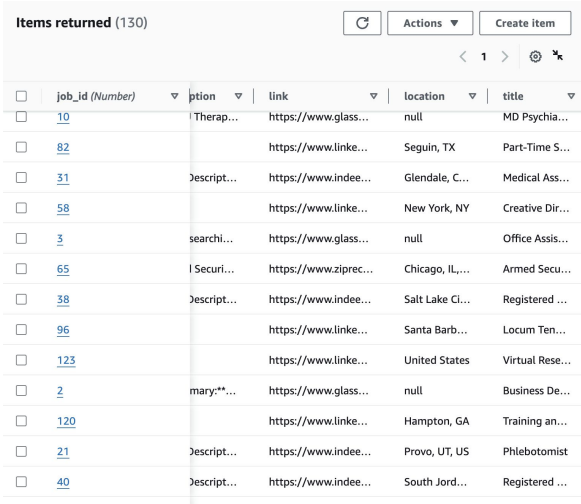
Software Design



Job Scraper

- Created 2 working job scrapers:
  - Linkedin (Primary Job Scraper)
  - JobSpy (LinkedIn, Indeed, ZipRecruiter, Glassdoor)
- Challenges:
  - Deploying the JobSpy scraper
  - Need for third-party subscriptions





Distributed Task Scheduler

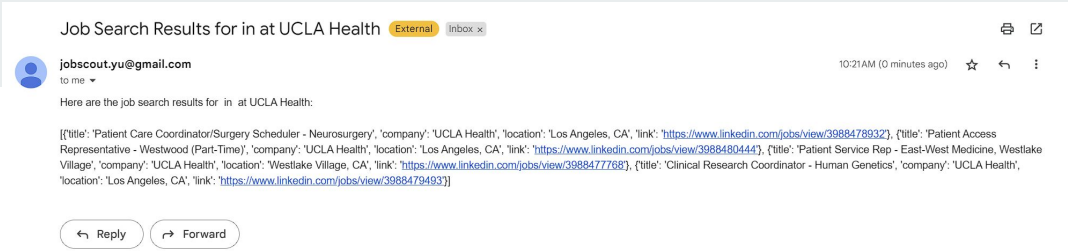
- Handles two types of tasks:
  - Notifications
  - Refresh
- Checks every minute for executable tasks and executes them by sending Notifications to Notif SQS Queue and sending Refresh to Refresh SQS Queue.
- Utilizes multiple executor objects in order to distribute executions

Tasks Table:

task_id...	company	created	interval	location	title	type	user_id
0		1722380640	PT6H			refresh	
2	Yeshivas Ra...	1722380640	P1D	New York	Talmid	notif	107130786...
1	Google	1722380640	P1D	New York	Software E...	notif	107130786...

Executions Table:

task_id *(Number)*	segment *(Number)*	next_exec_time
2	3	1722467040
1	2	1722467040
0	1	1722402240



SQS Listeners and Email Service

Refresh Listener:

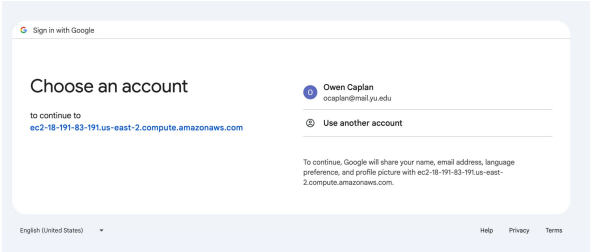
- SQS Listener that runs the scraper at an interval decided when booting up the system, and refreshes our jobs database.

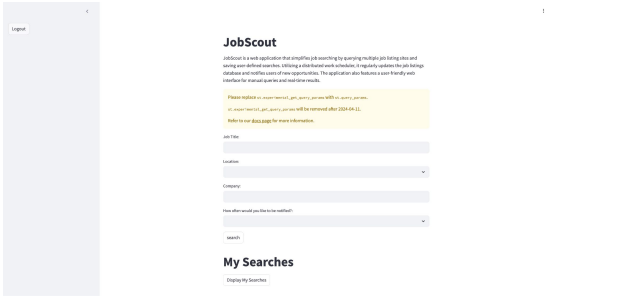
Notification Listener:

- SQS Listener that listens for executable tasks each minute and then passes the saved search preferences (such as location, company, job title) and User ID to our email service.
- Created a notification sender with Gmail SMTP that retrieves the user's email address that they used when logging in and sends the results to that address.

UI and Login

- Created a straightforward user interface with Streamlit:
  - Includes a page where logged-out users can execute instant searches
  - Includes a pages where logged-in where users can execute AND save searches
- Used Flask for login-logout logic:
  - The app includes a login button that redirects to a Google login page and a logout button to return to the standard page.
  - Logged-in users are assigned an ID which allows them to save searches





Software Development Skills Gained:

- Git/GitHub
- Stand-up meetings
- Working on a project as a team
- Local to Deployment
- AWS - IAM, EC2, SQS, DynamoDB

What's next

- Load Balancer
- Deploy the newer job scraper
- Improve website and email formatting
- Search suggestions
- More jobs scraped
- Delete/update searches

Demo

https://youtu.be/f4eIl1FLbGo