This paper is provided in response to the request of writing a “one page writeup on how you went about doing this project, any research you did, new tools learned, etc.”

A. How I went about doing this project

To start with, I am going over the one-page request for this document. Fortunately, the additional content does not represent a large additional level of effort to consume.

Regarding my approach to this project, the following is summary of the project’s software architecture:

- Infrastructure
  - GitHub
  - vi
  - WebStorm
- Languages
  - HTML
  - JavaScript
  - Node
- Platform
  - AWS (see diagram)
- Webserver
  - Apache

Considering the project's problem statement: convenient, free, simple, the selections noted above make sense considering those requirements.

For the “bonus” part of the assignment, what was delivered was not tested fully. Since I do not have the private key, I could not generate a token solely based upon the public key for eventual validation of the signature. For testing purposes, I created my own keypair, generated a token from my keypair, and validated the signature. Results of the row in the table turning red (or not) are available as a video uploaded to GitHub.

B. Research I did

I initially struggled with parsing a user string to see if the input was a token. Where I went wrong was using the jsonwebtoken library. I initially missed the nuance of the provided PGP key being an ES256 key where jsonwebtoken has limited ECDSA support. In the end, I used a library called jose.

For what the deliverable is, I did not use much AI -- and for the AI I attempted to use, the results were of varying quality. For example, when I asked AI a question about Google’s OAuth:

1. AI directed importing a Google script that has been deprecated;

2. AI did not know the current state of Google Identity Services (GIS) offerings;
3. AI did not have a current understanding of the layout of the Google Developer Console and pointed me to controls/services that did not exist or have been changed in the current version of the UI.

Other resources used for research include Reddit and Stack Overflow.

C. Did I enjoy building it?

I enjoyed it. This was an opportunity to dive into some hands-on keyboard development work. Engineering and security are evolving fields so it makes sense to do work like this to keep one's skills relevant.

I also needed to be disciplined in managing this project. The project had a deadline, it required some research and I lost about 6 hours due to troubleshooting multiple problems with AWS (including not being able to login). I had to be mindful of not chasing minutiae and remind myself of the requirements in the problem statement so I could deliver on time. With more time and resources, my deliverable would have been much cooler. But then again, every practitioner says that.

1. What could I have done better?

   As I write this paper, I'm having thoughts of things I could have done better. Considerations I have but that did not get implemented in the project include:

   a. I should have deployed this webpage in a container. But I got too focused on getting results quickly rather than considering design.

   b. I wanted to run SonarQube and provide a SAST report. But at the time of this writing, I'm experiencing problems with Sonar and GitHub integration. I'm going to post some questions online about that later.

   c. Logging. I really only logged the Node server starting and JWT parsing.

   d. Google Identity Service integration is clunky. Frankly, I'm going to blame Google for some of that. Vendor documentation does not seem to be aligned with the current state. The Google Developer Console also does not support OAuth authorized origins that don't end with a public top-level domain. That can be a challenge for homework such as this. My project meets the requirement of logging in with Google to be able to submit data to the table but I do not like the esoteric 403/4 errors I see in the browser's developer console (F12).

   e. DNS. The project could have a friendlier URL.

f. The website needs a certificate. I thought about self-signing but that really is not security -- it just allows a connection to pass.