Planning

Fictional Scenario

The client, LittleLives, is an educational technology company. They use the online platform GitHub to collaborate in code design and implementation. GitHub allows multiple users to work on the same repository of code independently without conflict. To track bugs and errors in any repository, GitHub has a built-in feature called "Issues". Issues allow any collaborator to log bugs or mistakes that need to be addressed. As a project manager, the "Issues" tab contains information regarding bugs that he doesn't need to know; hence, he wishes only the relevant data to be presented to him via a spreadsheet.

For data to be easily shared and accessed, the client requested the application to *"create GitHub Issues on Google Sheets"*. LittleLives has further stated that each "Issue" should have the following headers for easy tracking: *"Number, Title, URL, Username, State"*. Moreover, each "Issue" should only have one row pertaining to its information; as the Issue is edited, its respective row on Google Sheets should reflect this without the need for a new row of data. Lastly, LittleLives has stated that this task needs to be "*finished within seven days"*.

Rationale

The proposed solution is to build a web service that'll act as an API endpoint. We can then set up an adjoining GitHub repository with a Webhook subscribed to its "Issues" such that whenever an "Issue" is raised, it'll send a "POST" request to our API. Our API will, after authentication, process the data and retrieve the relevant information stated by the client. After that, we'll update the spreadsheet with the needed information by using a Google Sheets library along with the Google Cloud API.

I'll create this web service in NodeJS because Webhook's data is given in JSON (JavaScript Object Notation). Hence, working in NodeJS allows me to access and process the data. Furthermore, I've decided to build this as a "serverless application" utilising AWS (Amazon Web Services) Lambda functions integrated with AWS API Gateway. This is for three main reasons; firstly, serverless applications increase development speed, allowing me to meet the deadline. Secondly, serverless applications are easier to maintain in the long run as there is no physical server required. Lastly, by building my API on AWS, I protect my home network as there is no need to port forward my computer to give anyone access to my API.

Success Criteria

- Any GitHub repository with the correct authentication and API URL should be able to connect to the server and update the Google Sheets.
- The Google Sheets document should have the ability to contain multiple GitHub repository "Issues" logs. Every sheet within a document should represent one repository log.
- To test whether the implementation of the Webhooks API works, the Webhooks "Ping" command should automatically initiate a new sheet for the repository, given that it doesn't already exist.
- New issues should get their own row on the Google Sheet. However, updates to an old issue should be modified on an already-existing row.
- Issues should be sorted by "Sheets", with each "Sheet" representing a repository.
- The Google Sheets will contain the headers with the correct information for *"Number, Title, URL, Username, State".*