Overview:

Your task is to create a web application front-end and back-end for viewing and editing working shifts. This application will be deployed to Google Cloud Platform (GCP) and available on the internet.

Back-end:

The back-end service should be a Python application that has 3 endpoints.

The first endpoint will read and write a setting for a preferred IANA timezone string.

The second endpoint will be CRUD for workers, where each worker has a name.

The third endpoint will be CRUD for working shifts, where each shift contains the following properties:

- associated worker
- start datetime localized to preferred timezone and represented as ISO 8601 datetime string
- end datetime localized to preferred timezone and represented as ISO 8601 datetime string
- read-only computed duration between start and end times, in floating point hours

Shifts should be validated on the back-end that they do not overlap with each other and do not exceed 12 hours. Multiple shifts can be added per day per worker. The timezone should be able to be updated, and automatically apply to existing saved shifts.

The application should perform well with massive amounts of data.

This application will deploy to Google Cloud Run using Google Cloud Datastore (officially known as "Cloud Firestore in **Datastore Mode**") as a persistence store (i.e., database).

Front-end:

The front-end should be a Vue.js application for viewing and editing the back-end endpoints described above.

Datetimes should be displayed according to the preferred configured timezone.

It should display nicely on all screen sizes.

It should be deployed to GCP's Firebase Hosting.

Additional instructions:

We encourage you to use any third party libraries or frameworks to complete the task.

The GCP SDK contains emulators you can use for local development and testing. There should be automated tests for both the front-end and back-end.

We have set up a special GCP project for you to use for this task, and granted your email address access to this project to manage the necessary resources.

If you are having trouble or difficulty on how to start the task or what frameworks/technologies/concepts to try, send us a follow-up email with any questions.

We will notice attention to detail and any extra cool things added.

Submission:

Please commit all your code to a Github repository containing a README file with any special instructions or documentation on how to run, build, test, deploy, etc. the project.

Also provide the url of the front-end where it can be run.

Good luck!