HeyGen Documentation:

Short Description of how to use Client Library:

The JobStatusChecker client library is written in Python to interact with the Job Management Server. It enables users to start multiple new jobs on the server, track the progress of existing jobs, and handle responses based on the completion status.

Note: Make sure the port you are using (I used port 5000) is not running any sort of job. If there is any job running on the port, kill that job or use a different port number.

Step 1:

Installations:

1. Install the **requests** package**:**
    - `pip install requests`

Step 2:

Running the Client Library:

- Simply run the following command to run the client side:
    - `python3 client.py`

Step 3:

Creating Jobs

- A user can create a job using the following curl command:
    - `curl -X POST 'http://127.0.0.1:5000/start_job'`

  A user will see something like below if a job is created successfully:

```
{
  "job_id": "1730329958865"
}
```

Step 4:

Checking the status of the job:

- The library is designed in such a way that it can handle multiple jobs at the same time. The jobs are stored in a dictionary which is handled on the server side.

- To check the status of a job, simply use the following command:
    - `python3 client.py job_id`

- If only `python3 client.py` is used, the library creates and executes a new job.

Short Description of how to use Server Library:

The client side interacts with the server side. I have utilized flask which provides endpoints for starting and monitoring async jobs. Each job has a 10% chance of failure, simulating a realistic process with potential errors. The server is designed in such a way that it processes the job in the background and provides a simple API for job creation and checking the status.

Requirements:

- Python 3 or higher
- Flask Library

Install Flask using the following command if not already installed:

- pip install Flask
- pip install Flask requests

Then simply run the server using the following command:

- python3 server.py

How to use tests (Integration and Unit Tests)

I am utilizing unittest framework which is the default testing framework used in Python for unit testing.

- Simply run the following commands:

For Integration tests:

- python3 integration_test.py

For Client Unit tests:

- python3 unit_tests_client.py

For Server Unit tests:

- python3 unit_tests_server.py