Documentation

Most of the files were taken from the lab handout, and only requirements.txt, main.py were edited. I have added an additional requirement for pydantic to use as a structure for adding students.

Also, there are a few more .http files created to test different variations of GET,POST and DELETE requests.
Functionality includes:

1. DELETE requests which can handle
   1.1 Batch deletes if gpa is lower than a certain value
   1.2 Singular deletes based on student id.

2. GET requests with combinations of
   2.1 sortBy,count and offset
   2.2 No query parameters
   2.3 However, the dataset is small,so offsets are best tested when limited to 1 or 2

3. POST requests that can
   3.1 Add a new student into the list of data
   3.2 This new student can be reflected and used in the above requests

4. A route returning a content-type which is not plaintext, and application/json instead.
5. Error messages and response codes if invalid value is given
   5.1 GET and DELETE requests are complete
   5.2 Only POST requests to create students with invalid name and student id were handled, trying to detect a missing grade was harder

6. Idempotent routes will be the GET requests, as they only change the view of the datasets temporarily, and will be undone when we move back to localhost:8000/students

HOW TO RUN THE CODE
As usual, we would need docker to run in the background, and run "docker compose up" in the terminal with current working directory as the lab directory. Ensure that the requirements.txt has also been modified to include pydantic.

I used a REST client extension for vscode to test out these requests as it was easier to click on a button, however, the requests can also be tested by typing the url as given in the .http files.

I have created different files for different situations, like delete non-existing data, combinations of query parameters and posting students with invalid data as mentioned.

Expected outputs are documented in the http files for easier comparison after running the requests.