Part 1: SWAPI

Using SWAPI, figure out the complete URLs (including params and queries) that you need in order to reach the following data:

1. **The height of Darth Vader**
   https://swapi.dev/api/people/?search=darth
   https://swapi.dev/api/people/4

2. **The population of the planet Alderaan**
   https://swapi.dev/api/planets/?search=alderaan
   https://swapi.dev/api/planets/2

3. **The name of the manufacturer of the Millennium Falcon**
   https://swapi.dev/api/starships/10

4. **The name of the species that C-3PO belongs to (multiple URLs)**
   https://swapi.dev/api/people/?search=C-3PO
   https://swapi.dev/api/people/2
   https://swapi.dev/api/species/2/
   https://swapi.dev/api/species/?search=droid

5. **The title of each film that Obi-Wan Kenobi is in (multiple URLs)**
   https://swapi.dev/api/people/?search=obi-wan
   https://swapi.dev/api/people/10
   https://swapi.dev/api/films/1
   https://swapi.dev/api/films/2
   https://swapi.dev/api/films/3
   https://swapi.dev/api/films/4
   https://swapi.dev/api/films/5
   https://swapi.dev/api/films/6

6. **Use the search query (the how to on the search query is at the bottom of the Getting Started section of the documentation) to get the information about the Millennium Falcon, it's a starship**
   https://swapi.dev/api/starships/?search=millennium

Part 2: Social Mountain
In this section, you'll be looking through the documentation for the Social Mountain API and answering questions. You'll also be making requests and recording the URLs and some information about the responses. Run the requests in Postman. **Note: this API is live and viewable by your classmates and staff. Keep things appropriate for class.**

The base URL of your requests is: https://practiceapi.devmountain.com/api (make sure to have the "s" in "https")

1. **Check if the POST request accepts params, queries, and/or a body. Which one(s) and what information is it expecting to be sent?**
   POST requests take in a body of "text". It accepts parameters of the three different keys: "id", "text", and "date".

2. **What data type does the GET request return?**
   GET request returns an array of all posts/objects.

3. **What would the URL look like for deleting the post with the id 555? (This post does not exist anymore, but the syntax is the same for existing posts)**
   ```
   https://practiceapi.devmountain.com/api/posts?id=555
   ```

4. **List the possible response codes from the GET request at '/posts/filter'**
   200
   409

5. **Create a post whose text is your name, record the URL and body here:**
   ```
   {
           "id": 299,
           "text": "Nicki Boothman",
           "date": "01 Dec 2021"
    },
   ```
   https://practiceapi.devmountain.com/api/posts/filter?text=nicki

6. **What would the URL and body object be to update the post you just made to contain your favorite color instead of your name?**
   https://practiceapi.devmountain.com/api/posts?id=299

   ```
    }
           "text": "yellow"
    }
   ```

7. **What is the URL to get posts that contain the text "blue"?**
   https://practiceapi.devmountain.com/api/posts/filter?text=blue

8. **Make a request to GET all the posts. What are the content type and charset of the response? (Hint: look on the Headers)**

Content-type is application/json. Charset UTF-8.

9. **What would cause a PUT request to return a 409 status?**
   If the request was missing req.query.id or req.body.text

10. **What happens if you try to send a query in the GET request URL? Why do you get that response?**
    I searched for the id 299 and received all posts as a response. The GET request does not accept parameters so it would not accept queries with parameters. To narrow the search and use queries, you have to use /posts/filter then the query.