**SOA4 Project**

**(20% of final grade)**

**Due date: Sunday 30th March.**

**Overview of Project:**

1. Create two **new** Services (**not** Student/Course or anything we used in class) of your choice. Use JPA for the databases.
2. Connect Service A (consumer) to Service B (producer) using Request-Response (Asynchronous Nonblocking) style of communication.
3. Research how to use ETags in Spring Boot to implement HTTP **caching**, and implement it in the project. Use caching on the collection only.
4. Design a HTML/JS client to showcase the communication and caching in action.

**[20 Marks]**

Note: You’ll have to demonstrate the project in week 12 (31st March). You should be able to explain the code and answer any questions relating to your project.

**Note on the JavaScript Client:**

Client should show the attributes of Service A in suitable format (e.g. table)

Client should be able to refresh the entities and indicate whether the request was a 200 or 304.

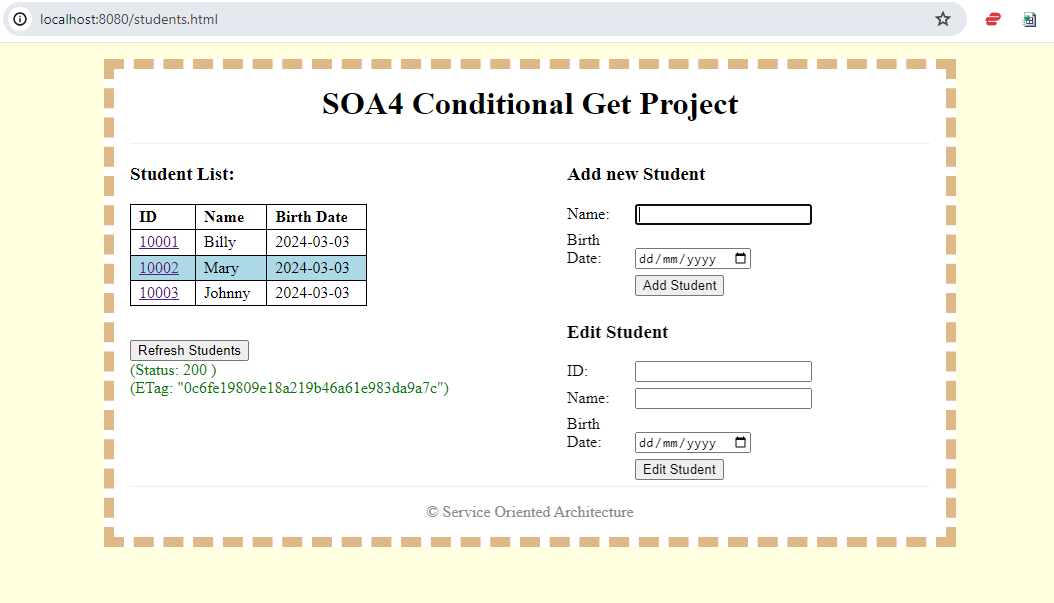
Client should be able to add a new entity (to Service A, the consumer).

Client should be able to edit an entity (on Service A, the consumer).

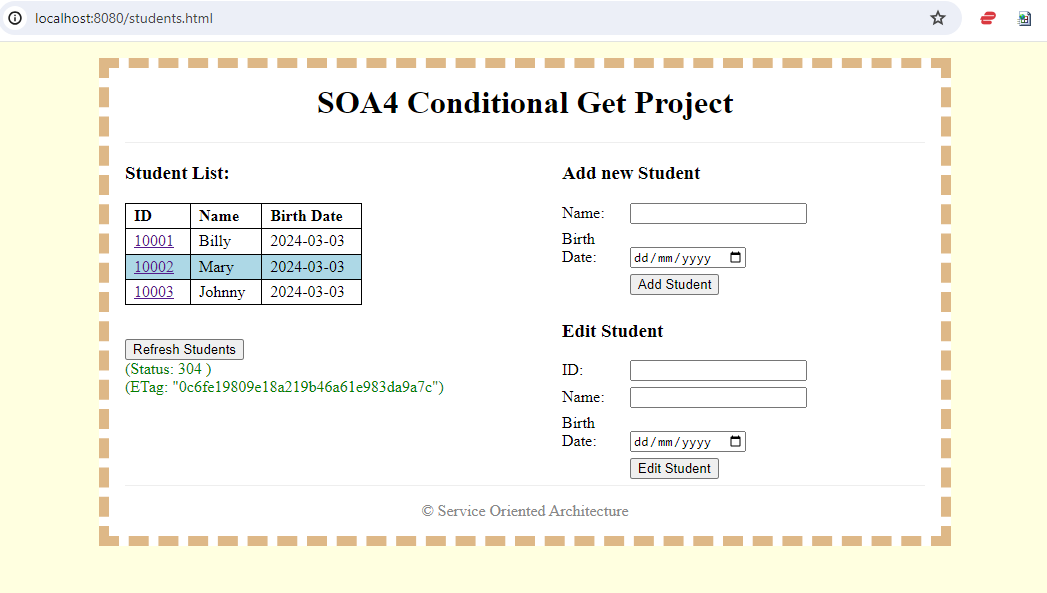
**Sample to show the caching aspect of the project:**

(Note: You should not use the Student example from class. Also, this example doesn’t connect to another service – you’ll have to add to the interface to show this.)

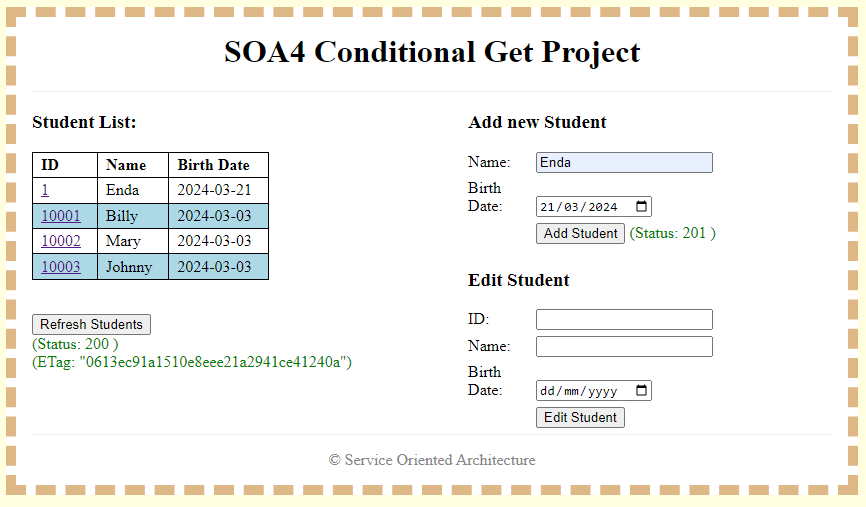
Initial Screen:



After clicking ‘Refresh Students’:



After adding ‘Enda’ and clicking on ‘Refresh Employees’:



**Notes:**

* You can add the HTML/CSS/JS files to the **src/main/resources/static** folder of your project.
* The JS Fetch API can be used for the GET, POST and PUT requests.

**Submission:**

* Both Spring-Boot projects.
* A document with screenshots to clearly show the caching in action, as well as demonstrating the functions to Add and Edit.
* The document should also highlight and explain
  + the code used to connect the two services
  + the code used to implement the caching