CS5208 - Distributed Systems

Assignment 1 (10 Marks)
Deadline: 21st March 11:59PM

Please do not copy.

1. - Read about REST (https://restfulapi.net/)

    - Read about net/http package in Go (https://golang.org/doc/articles/wiki/)

    - As part of the assignment, you are required to develop a meeting scheduler.

        - A server that accepts requests from faculty. Assume there are 10 faculty interacting with the server. Faculty names can be F1, F2, ... F10.

        - A faculty will specify the timeslot and date when he/she wants to block his/her calendar. All timeslots are of 1 hour granularity. For example, faculty named F1 POSTS a blockCalendar message to the server with time 14:00 on 23/Mar/2021.

        - A faculty will specify the timeslot and date when he/she wants to schedule a meeting with a group of faculty. All timeslots are of 1 hour granularity. For example, faculty named F2 POSTS a scheduleMeeting message to the server with time 16:00 on 23/Mar/2021 and specifies the group as F3, F1, and F5. He/She also specifies a meeting title.

        - blockCalendar and scheduleMeeting POST messages fail if the calendar time for any faculty in the group is already occupied by a different meeting or is blocked previously.

        - A faculty can retrieve his/her day's schedule. For example, faculty named F3 GETS from the server the schedule for day 23/Mar/2021. All the meeting/blockCalendar slots should be displayed for him/her.

        - A faculty can post PUT/PATCH/DELETE for an existing timeslot.

        - A faculty can interact with the server using curl (https://curl.se/) or using a client program written in Go.

        - HOD (say F10) can GET from server a summary of all the meetings that happened in the department for one whole month (not the blockCalendar slots of individual faculty). The output can just be meeting title, time and the participating members.

        - A faculty specifies in the message to the server his/her identity (F1, F2, .. F10). There is no need to authenticate the same.

        - Use a datastructure of Go to store all data (In realworld it would have been a database).

- The server is expected to handle concurrent requests, so ensure that your datastructures are safe from race conditions.

- All interactions can be through command line. You can choose have a webpage displayed.