University of Glasgow | School of Computing Science

# CSC3004: Cloud and Distributed Computing
# Lab Assessment: 2021-2022

| | |
|---|---|
| *Date issued* | *Friday, 3 June 2022* |
| *Final submission deadline* | *23:59 (11:59pm) Monday, 27 June 2022* |

This is the lab assessment for the Cloud and Distributed Computing module; it is worth 10% of the assessment for the module.

You are asked to *design* and *implement* a simple *"SafeEntry"* system in an effort to combat COVID-19 in Singapore using **gRPC Python**. The *SafeEntry* system should consist of a server and client programs, as described below:

The client program(s) should enable users to perform check-in and check-out (specifying name(s), NRIC number(s), location (building, shopping mall, restaurant, etc), check-in and check-out time; returning success or failure). Note that a user can also perform a *group check-in / check-out* for family members. Procedures (Functions) for listing the history of all *SafeEntry* locations should also be implemented. If the users were at a place visited by a COVID-19 case in the past 14 days, the users should be notified that there was a possible exposure and that they must monitor their health for 14 days from the date of visit. The notification should include information such as location(s) of the exposure and the corresponding check-in and check-out date/time.

A special (remote) access for Ministry of Health (MOH) Officers should be implemented to enable them to declare that a location had been visited by a COVID-19 case. Such a declaration should include the visit date and time of the COVID-19 case. With this, a notification will then be sent to the affected users based on the *SafeEntry* records.

The *SafeEntry* server should be able to handle concurrent check-in and check-out, deal with requests from clients, and maintain the state of the *SafeEntry records*. Multiple clients are expected to be launched from machines in the network and invoke the appropriate methods on the server. User's *SafeEntry* information must be retained and queryable by a client application. The server should also be able to save/restore *SafeEntry* state and information to/from permanent storage (e.g., write to a file and no database is required for this exercise). The latter feature should be used for bootstrapping the server with some initial *SafeEntry* records (for system testing purposes).

Your *SafeEntry* system should deal with issues arising from concurrent client access to *SafeEntry* records, as well as from the implicit requirement to keep state for clients and *SafeEntry records*. Note that you are not expected to provide a sophisticated user interface for any part of the system (a simple text based interface will suffice).

You are asked to *write a report* (of no more than 3000 words) on your *SafeEntry* system discussing your overall design, and how your implementation meets the design goals and requirements. You should make clear (a) which parts of the specification you have actually implemented, (b) whether your solution works as advertised, and (c) how you have tested it.

The report should include a discussion of potential extensions that could improve the fault tolerance and availability of your system (you are not expected to implement these extensions!). Finally, you should include a section describing how you would expect me to test your system that you have submitted. You can also package your system into a Docker container if you wish.

**Important**: This is a pair exercise (two students in a group) and must not be discussed with other classmates not belonging to your group or anyone else. However, you can discuss the non-assessed code presented in class and any examples of gRPC Python programming that you find on the Internet.

## What to Submit

You should submit your solution electronically via the LMS Dropbox Submission link.

Submissions should consist of two files:

1. A single file (e.g., a .tar, .zip, etc.) with your source code listings, and instructions on how to extract, compile, test your code.
2. A single .PDF file of your report

Both filenames should be called NNNNNNN.suffix, where NNNNNNN is your SIT ID, suffixed by the appropriate file extension.



**END/**