|  |  |  |
| --- | --- | --- |
| **Objective** | **Explanation** | **Completed** |
| Server Database | | |
| 1. Store information of all facilities (Name, Type) | Tamarind Meeting Room, North Spine Lecture Theatre, etc. |  |
| 1. Store availability of facility over 7 days a week | Time represented in day/hour/minute format, enum/integer/integer format. |  |
| 1. Store bookings made by users | Each facility requires exclusive use. |  |
| Server Services | | |
| 1. Allow query of availability of a facility | Specify facility name and the range of days. |  |
| 1. Allow booking of a facility for a period of time | Specify facility name and the start and end times of booking. Store the period. |  |
| Must return a unique confirmation ID to client, linked to facility, start and end times of booking. |  |
| Update the availability of facility at server. |  |
| 1. Allow change in booking of a facility | Specify confirmation ID of booking & offset for changing (assume 30 mins block). Does not modify the length of time period booked. |  |
| Successful change, send acknowledgement to client. |  |
| Update the availability of facility at server. |  |
| 1. Allow monitoring of availability of a facility | Client to provide facility name and length of monitoring period. (Tamarind Meeting Room, Tuesday, 09h, 30min to Thursday, 09h, 30min) |  |
| Internet address and port number of clients are recorded by the server. |  |
| If changes made (new booking / change booking timing) to availability of facility during the monitoring period, notify the client by sending updated  availability of the facility. |  |
| After expiration of the monitoring period, remove the client’s record (internet address and port number) from server, stop sending updates. |  |
| Each user can only send one monitoring request, but the server must allow multiple clients to monitor availability. |  |
| 1. Idempotent service | TBC |  |
| 1. Non-idempotent service | TBC |  |
| Client Interface | | |
| 1. Client should know the server address and port number | Server address and port number specified as arguments in the command that starts the client. |  |
| 1. Server should only know the client’s address after receiving client request | Client should attach its address in the request. Server only handles one request at any time. |  |
| 1. Text-based interface for the client, to pick options 1 to 6 and terminate the client | While loop, print statements, accept user’s inputs. |  |
| Additional Requirements | | |
| 1. Proper error message for mistakes in user’s inputs | [S1, S2, S4] Wrong facility name |  |
| [S2, S3] Unavailable facility |  |
| [S2] Bad start / end booking times |  |
| [S3] Bad confirmation ID |  |
| [S3] Bad offset to advance / postpone booking |  |
| [S4] Bad monitoring period |  |
| Consider for Service 5 & 6 as well |  |
| 1. Design request-reply message formats | TBC |  |
| 1. Marshalling / Unmarshalling | You may use a byte array to store the marshalled data. Do NOT use  any existing RMI, RPC, CORBA, Java object serialization facilities and  input/output stream classes in Java.   * Encode (Byte, Byte[ ]) |  |
| 1. Fault-tolerance measures | Retransmit request message |  |
| Duplicate filtering |  |
| Re-execute method or re-transmit reply |  |
| 1. Facility information storage | Does not need to be persistent but should maintain information during execution of server program. |  |
| 1. All messages transmitted in the form of a sequence of bytes. | Marshall integer, strings before transmission and unmarshall then upon receipt. Consider adding length information of the string in marshalled messages. |  |
| 1. Implement the system with 2 different invocation semantics, at-least-once & at-most-once | Which semantics used should be specified as an argument in command that starts the client / server.  At-least-once:  Retransmit request message, but no filtering of duplicate requests, re-execute method upon request.  At-most-once:  Retransmit request message, filter duplicate requests, retransmit stored reply messages upon duplicate request. |  |
| Possibly implement timeouts, filtering duplicates request messages, maintaining histories.  Possibly assign request identifier to each request to detecting duplicates. |  |
| Stimulate the loss of request and reply messages, show that at-least-once invocation semantics can lead to wrong results, whereas at-most-once is always correct. |  |
| 1. Optional – Implement the server and client in 2 different languages. | Server in C  Client in Java |  |