Outfitter code challenge

1. What I covered in current release:

* Customer management: Implemented getAll, getById, add(incorrect but can be used also as update) and remove.
* Stylist management: Implemented add, readyForStyle and requestForLeave.
* Reservation management: Implemented getAvailableTimeSlot, makeReservation, updateReservation, makeManyReservation

1. Customer Management: Implemented regular CRUD operations, nothing to say more.
2. Stylist Management:

* Add: Adds new Stylist with state Rookie.
* readyForStyle: update stylist state to “Ready to Style”
* requestForLeave: send stylist for temp or permanent leave. In current implementation divided into 2 parts. Rookie and Ready to Style does not require action, Sick, on holiday and Offboarded requires. In temp leave (sick, on holiday) stylist go vacation or sick leave in a period. For these actions only, reservations from this period affected (move to other stylists) and for Offboarded all reservations. Not implemented in this version: no time slot available reservation which is needed to change stylist.

1. Reservation management:

* getAvailableTimeSlot: calculate free time slots and return string list of time slots.
* makeReservation: each customer can have one active booking.
* updateReservation: used for updating reservation date, time slot and may be stylist.
* makeManyReservation: used to insert bulk of reservations. Returns in response 2 list: processed and failed reservations. Planned to add failed reservations to a different table for implementing different options, but not reached for current release.

Unfortunately, all rest API`s implemented only for positive scenarios.

What is missing:

1. correct implementation of rest API`s
2. input validations
3. logging
4. test cases
5. missing reservation list API

After implementing all missing part what I would like to improve: in current implementation API`s getAvailableTimeSlots and makeReservation aren`t implemented in a read-intensive way. Because of this API`s use query with many joins. As an optimization, I planned to cache these queries results in a different table and use this table for getAvailableTimeSlots, makeReservation and avoid every time invoking queries with many joins. Because this app not used only in one country there can be added timezone support as an improvement.