Scenario:

There is a registration request from a patient.

One Clinic Center administrator approves and the other administrator rejects the same registration request at nearly the same time.

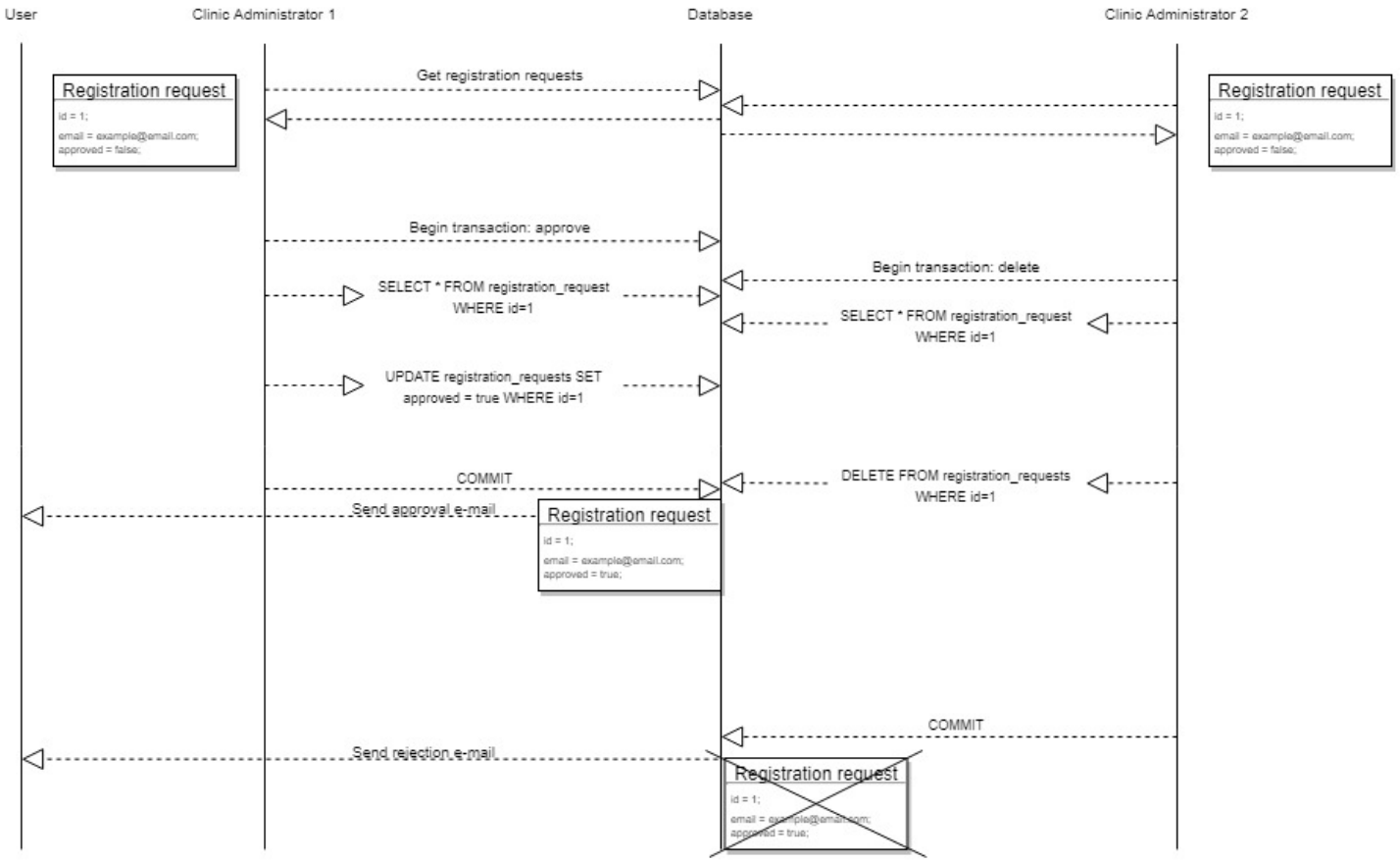
Problem:

The server gets one request to approve the registration request and one to reject (delete) it.

Let's say the method that should delete the request from the database gets there first, and successfully reads the data.

The second method is supposed to approve the same request, which it does – while the first method is still performing some checks. The second method then saves the, now approved, request to the database and sends an e-mail to the user. The first method finished all the checks and performs a delete command.

Now we are in a situation where one request has been both approved and deleted. The user receives two e-mails, one for approval and one for rejection.



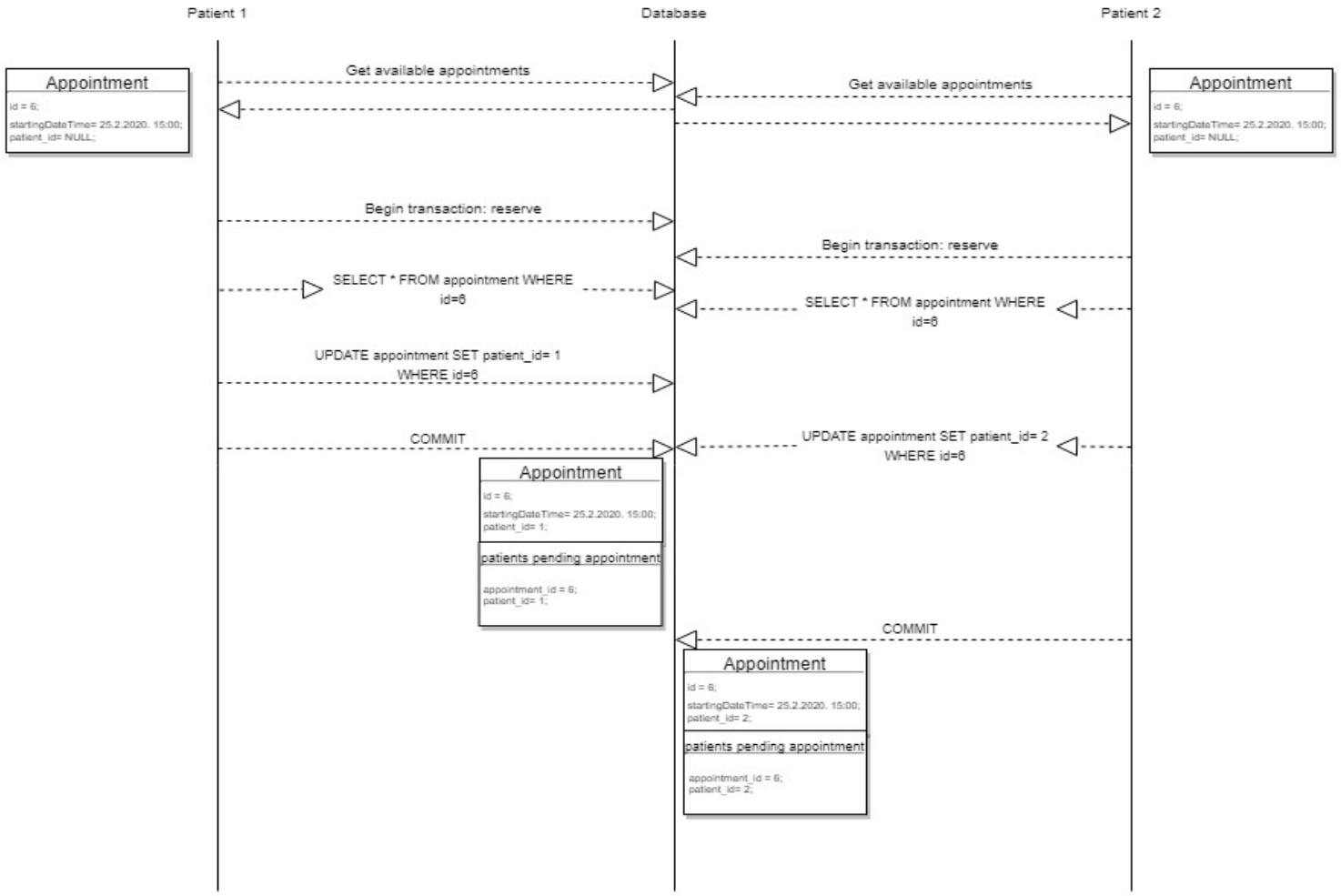
Solution:

Lock the registration request optimistically, so when a method tries to delete the registration request it has to check for the version, which will now be wrong and it will stop executing. The administrator will be notified accordingly.

Similar would happen if two administrators approved the same request at the same time. While this would not be a great problem, because the data would remain the same, it would be uncomfortable for a user (two e-mails about being approved), and/or it would create an issue if there should be an information on which administrator approved the request.

Scenario:
Two patients reserve the same predefined appointment at the same time.

Problem:
Predefined appointment can be reserved by only one patient. If clicked at the same time, without protection, the system would allow one predefined appointment to be reserved by two different patients, and the same appointment would appear as a pending appointment for both of patients.



Solution:

Lock the appointment pessimistically while it is being changed, so it can be changed/reserved by only one patient. This way if two requests try to get the appointment from the database in order to change it, only one will actually get it. The other one will be prompted to try again and/or choose another appointment.