1 2  9 0 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE D COIMBRA

BD Intermediate Report

João Esteves, Pedro Lima, Sofia Leitão

LifeLink

Bachelor's in Informatics Engineering
Faculty of Sciences and Technology of the University of Coimbra

March 22, 2024

Contents

1 Introduction

This report documents the progress and work done by the group for the intermediate review stage of the Databases course project.

The project name chosen is LifeLink, joining the concept of Healthcare and Databases through the chosen words: *"link"* - as in the relations between entities in a database, and *"life"* - self explanatory.

Members and contacts:

- João Esteves (2022230550@student.dei.uc.pt)
- Pedro Lima (pmlima@student.dei.uc.pt)
- Sofia Leitão (2022214134@student.dei.uc.pt)

2 Project Description

LifeLink is a simplified application for managing and storing processes for a Healthcare institution. It aims to provide different users a set of features specific to their role (patient, doctor, nurse or assistant) through an HTTP web server, allowing them to review and change information within the institution's database through simple HTTP requests.

The application layer in between the user and the database not only streamlines the functions from an end user perspective, but also allows a good developer the opportunity to protect the database, maintaining data integrity while maximizing the efficiency of its storing and accessing.

3 Transactions

3.1 Add Patient, Doctor, Nurse, and Assistant

Create a new individual, inserting the data required by the data model. This will add a new entry to the database.

3.2 Schedule Appointment

Create a new appointment, inserting the data required by the data model. Only a patient can use this endpoint. Remember that for each new appointment a bill should also be created using triggers.

3.3 Schedule Surgery

Schedule a new surgery, inserting the data required by the data model. If hospitalization_id is provided, associate with existing hospitalization, otherwise create a new hospitalization. Remember that for each new appointment a bill should also be created (or updated if the hospitalization already exists) using triggers.

3.4 Get Prescriptions

Get the list of prescriptions and respective details for a patient.

3.5 Add Prescriptions

When an appointment or hospitalization takes place, a prescription might be necessary.

3.6 Execute Payment

Pay existing bill. This will create a payment entity in the database.

4 Concurrency Conflicts

Given the nature of the project and the way it is constructed, there can be cases where concurrency conflicts may be destructive to the database. Here are the ones we have identified:

4.1 Assigning employees to overlapping tasks simultaneously

While checking in appointments and surgeries if a doctor/nurse is already assigned at that hour, two transactions could simultaneously find that entity to be "free", creating the issue of "same person, in different places, at the same time".

This can be avoided by locking the task table we are modifying, ensuring that other transactions don't simultaneously modify the same record.

4.2 Accessing bills, prescriptions and hospitalization information that has been altered but not yet committed, resulting in dirty Reads

By choosing an appropriate isolation level, we can achieve the required data consistency and integrity we need to avoid Dirty Reads.

5 Entity Relation Model

The group submits alongside this report an Entity Relation Model to modulate the database as appropriately as possible in the planning stage, being still open to further alterations.

The group assumes the heritage relationships between Client and Employee and Person are complete and with overlap, meaning no Client can exist without being either an Employee or a Person, and Employees can be Clients. Within Employee, the heritage relations of Nurse, Doctor and Assistant are disjointed (no overlap) and complete.

6 Development Plan

The group planned the development of the project more strictly for the next subsequent weeks, leaving later tasks more roughly outlined, already assuming the vision for these tasks and the group members' preferences will consolidate upon completion of the first round of tasks. Regardless, the following table aims to demonstrate the planning done.

Timeline

#	Week	Task Description	Group member
1	1	Start the database setup in PGAdmin	João Esteves
2	1	Explore Postman's professor example	Sofia Leitão
3	2	Start the application code (Rest API) in Python	Pedro Lima
4	3	Develop Endpoints in the application	Pedro Lima
5	3	Postman Collections	Sofia Leitão
6	4-7	Develop transactions/queries and database adjustment	Every member
7	8	Final adjustments and tests	Every member
9	8	Report conclusion and User Manual	Every member

The report, user manual, as well as any adjustments to the diagrams are to be done continuously throughout the development of the project.

7 Conclusion

With this report, the group aims to consolidate the planning of the project, and represent that work to the docent, as well as demonstrate the preliminary tasks of segmenting the different entities, relations, transactions and potential concurrency problems with the conceptualized system. The group accepts that this is an important part of developing database applications and software as a whole, and concludes that the project is in a good state for completion during this intermediate review.