Project and Seminar

2024/2025 - 2nd Semester

Bachelor in Computer Engineering and Informatics

Database Documentation

Ângelo Azevedo, n.º 50565, e-mail: a50565@alunos.isel.pt
António Alves, n.º 50539, e-mail: a50539@alunos.isel.pt

Advisor: Pedro Matutino, e-mail: pedro.miguens@isel.pt

March 2025

Introduction

This document provides an overview of the database, entities, attributes, and relationships. It also includes implementation decisions.

Database overview

The database has been modeled using an Entity-Relationship (ER) approach. This approach allows for a better understanding of the relationships between entities. The following section shows a figure with the ER model.

The database is implemented with PostgreSQL and tested using fake information in a Docker container.

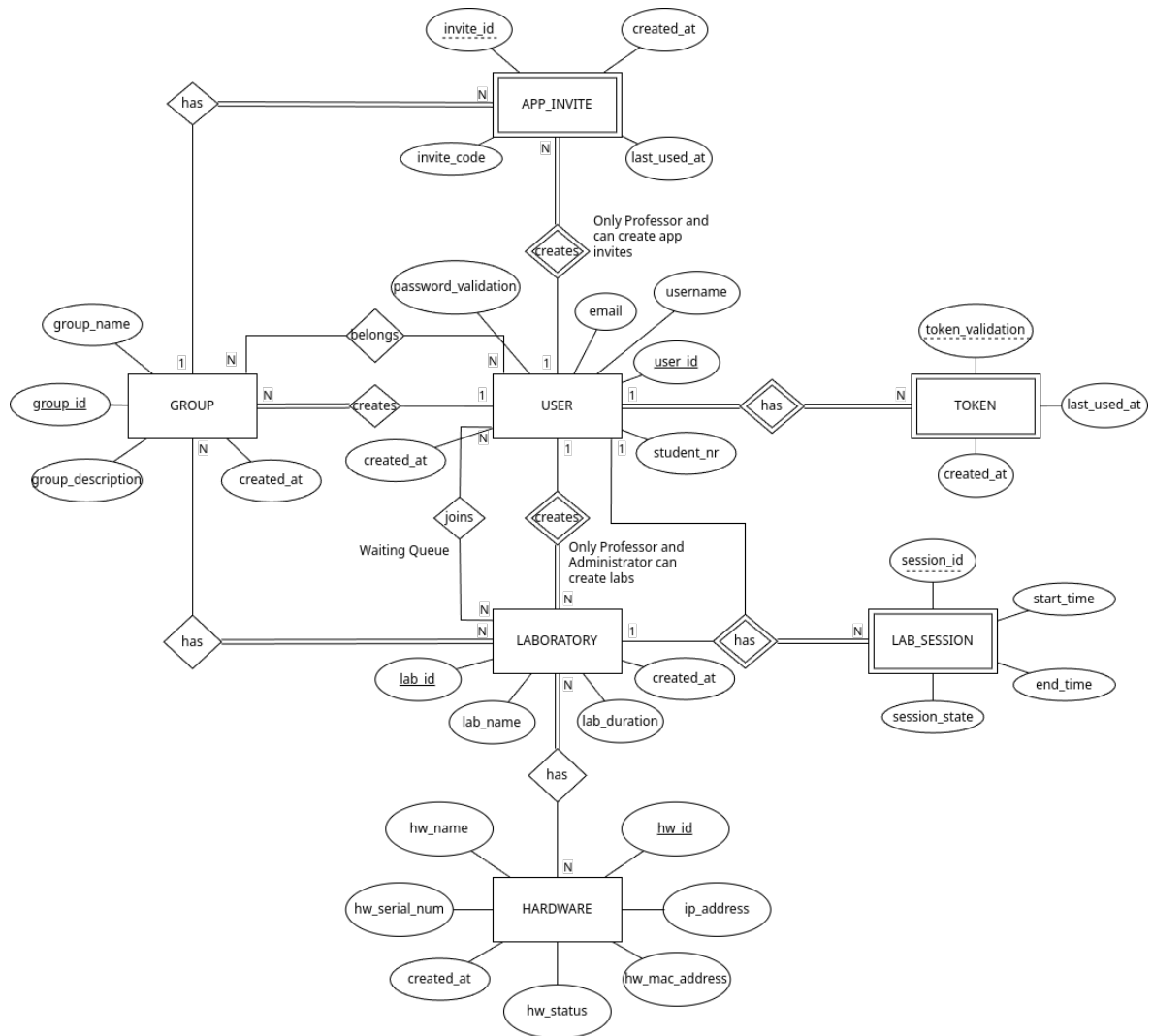The following section will address the ER model.

Entity-Relationship Model

Figure 1: Entity-Relationship Model

Entities and Attributes

This section provides a comprehensive description of each entity and their attributes.

User

The first important entity is the **User**. This entity represents a user in the database.

It has a **user_id** as primary key and identity column. An identity column is a special column that is generated automatically from an implicit sequence. So, whenever a user is inserted in the database it will generate an id. The user_id is an int data type.

It also has, as a char sequence, **username**, **email**, and **password_validation**. All of them are not null and email is unique. The password_validation attribute stores the hash of the user's password.

Finally, it has a **student_nr** (student number) as an int type and unique and **created_at** as a timestamp type and not null. The student number can be null. This allows users, like professors or others, to be a user entity in the DB. It is worth mentioning that there is not a discriminator attribute because this distinction will be implemented in the Role-Based Access Control (RBAC) system.

Token

Token is a weak entity because it cannot be uniquely identified by its attributes alone. This means that a token needs a user_id to be identified. This way, a token requires the user to exist.

Since it is a weak entity, it must have a partial key. This partial key is **token_validation**. The token_validation attribute is a randomly generated hash. It is a char sequence and not null.

The last attributes are **created_at** and **last_used_at**. Both are timestamp types and not null.

App Invite

App Invite is a weak entity. For the same reason as the token, this entity requires the user to exist and cannot be uniquely identified. For example, an app invite should not exist if the user is deleted.

As a partial key, the **invite_id** with user_id uniquely identifies an app invite. This invite_id is always generated as identity and is an int type.

It has an **invite_code** attribute as a char sequence and not null. This invite code can take up to 255 bytes, but the actual length is determined by the application domain.

It also has a **created_at** and **last_used_at** like the token entity.

Group

The **Group** entity represents a group. This group can be a class of students, a work group, a professor's group, etc...

It has a **group_id** attribute as primary key and uniquely identifies a group. It is always generated as identity and is an int type as well.

Also has a **group_name** as a char sequence and not null. This attribute represents the group name chosen by the user.

It has a **group_description** as text type. This is an attribute where a user can write something about the group that they like. It can be null and is a text type. A text type has an unlimited length, so that the user does not have a limit.

Lastly, it has a **created_at**. This attribute has the same characteristics as the token's created_at.

Laboratory

Laboratory, as the name indicates, represents a laboratory.

This entity has a **lab_id** as primary key. This is the unique identifier and indicates the laboratory identifier. It is always generated as identity and is an int type.

It has a **lab_name** attribute to represent the laboratory name. It is a char sequence and not null.

Finally, it has a **lab_duration** attribute to represent the duration of each lab session. It is an int type and is not null. And it has a **created_at** attribute similar to the group's created_at.

Lab Session

The **Lab Session** is a weak entity. This session should only exist if associated with a user and a laboratory. It represents a lab session, that is, a session where a user can manipulate something in the laboratory.

As attributes, it has **session_id** as a partial key. With user id and lab id, a lab session can be uniquely identified. It is always generated as identity and is an int type.

It has **start_time** and **end_time** attributes to represent the date and hour when a session starts and ends. Both are timestamps and not null.

Lastly, it has a **state** attribute to indicate the session state, that is, if it is active, inactive, or scheduled. This attribute is a char sequence and is not null. If necessary, additional states can be defined for this attribute. If more states are added, this document should be updated.

Hardware

The last entity is **Hardware**. This entity can represent any hardware. For example, it can represent a computer or an FPGA.

As primary key, it has a **hw_id**. It is always generated as identity and is an int type. This attribute uniquely identifies the hardware.

It has a **hw_name** attribute for the hardware name. It is a char sequence and not null.

It has a **hw_serial_num** attribute that represents the hardware serial number. It is a char sequence and not null.

It also has an **ip_address** and **hw_mac_address**. Both are char sequences and depending on the type of hardware, these attributes can be null.

It has a **hw_status** attribute as a char sequence and not null. This attribute indicates the status of the hardware, such as available, occupied, etc.

Finally, it has a **created_at** attribute with the same characteristics as the laboratory's created_at.

References

[1] PostgreSQL Global Development Group. Character types. `https://www.postgresql.org/docs/current/datatype-character.html`.

[2] PostgreSQL Global Development Group. Identity columns. `https://www.postgresql.org/docs/current/ddl-identity-columns.html`.