ESAT INNOVATION SCHOOL | ESCUELA SUPERIOR DE ARTE Y TECNOLOGÍA

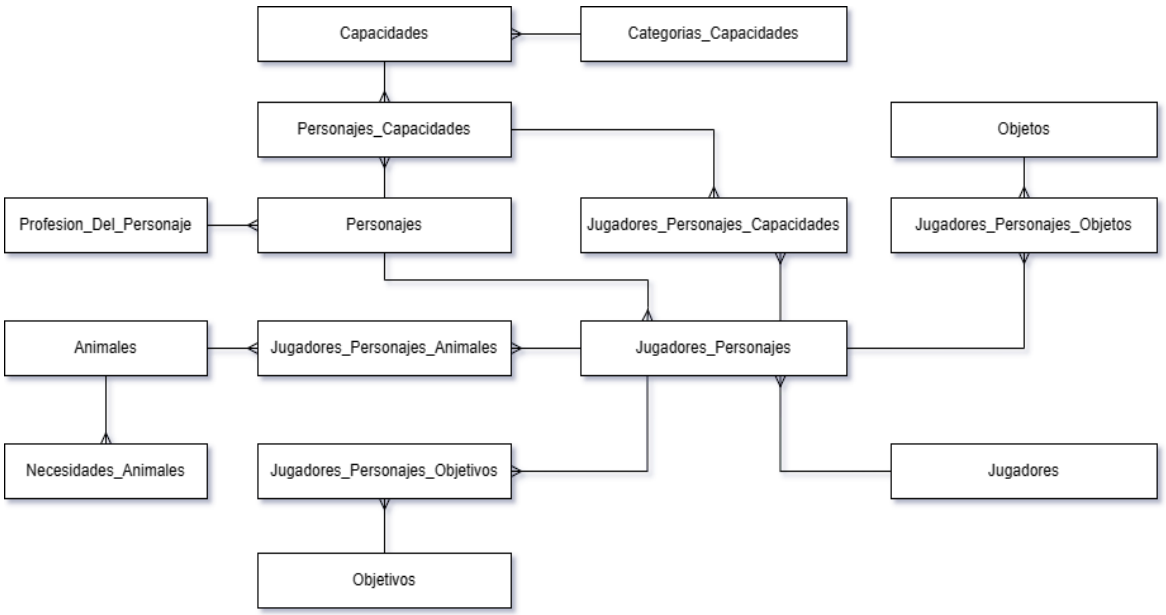
# DATABASE MANAGER

Subject: Advanced Programming
Module: 2nd VGP
Professor: Gustavo Aranda
Academic Year: 2023/2024
Authors: Lucas Calatayud, Luis Miguel Jiménez & Carlos Mazcuñán

# Índice/Index

# 1.- Database

## Database design




For this program, we use a database created in SQLite Studio for a hypothetical farm game with 14 tables, of which the following are related to each other. The "Personajes" table is connected to "Personajes_Capacidades" and "Profesion del personaje" using foreign keys. In turn, "Personajes capacidades" is linked to "Capacidades," and this is related to "Categorias_Capacidades." "Jugadores_Personajes_Capacidades" is associated with "Personajes_Capacidades" and "Jugadores_Personajes." "Jugadores_Personajes" is associated with "Jugadores_Personajes_Objetos," which is connected to "Objetos." "Jugadores_Personajes_Animales" is related to "Animales," which is connected to "Necesidades_Animales." "Jugadores_Personajes_Objetivos" is related to "Objetivos," and with "Jugadores."

The design of this database is effective in that we created many foreign keys to test in different scenarios to ensure that values of foreign keys referenced from other tables cannot be modified. As

for the rest of the values, we verified that they can be accessed and edited correctly.
With our program, we can make queries that return correct results as long as the query syntax is correct.

## 2.- Personal Tasks and Workgroup Plan

- **Lucas Calatayud**

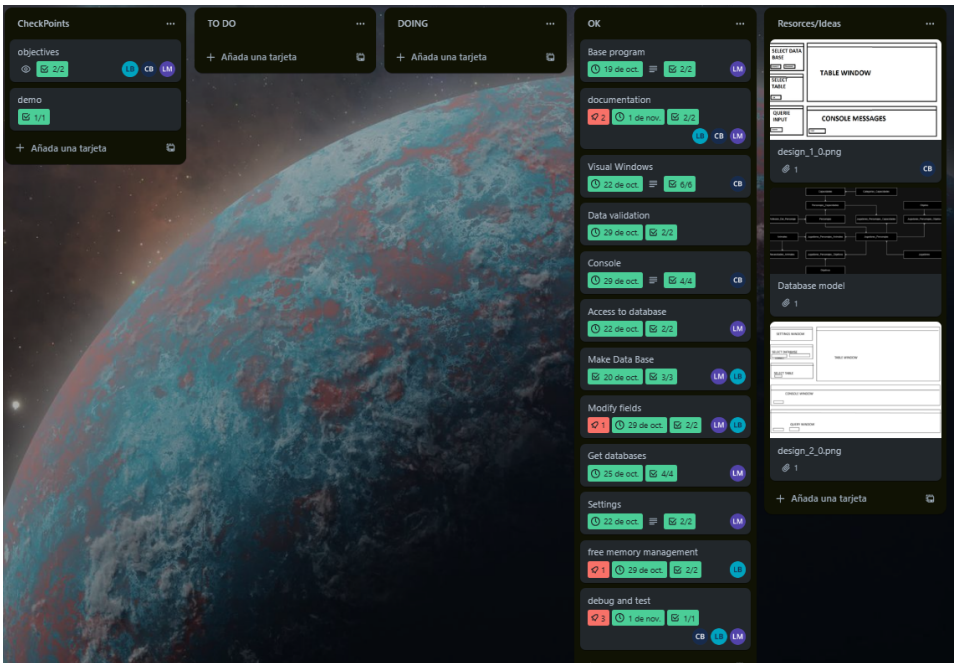
  As my personal contribution to the work, I mainly handle tasks related to research, debugging the program, testing the program, populating the databases, and documentation, creating both the sample database for testing the program and the configuration database.

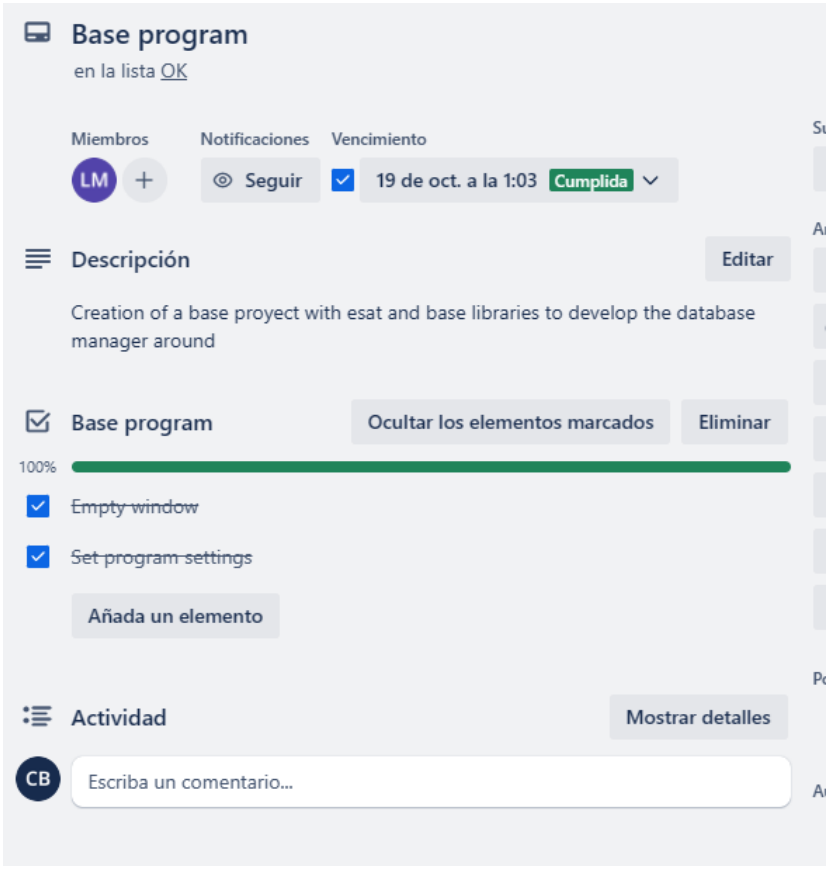
- **Luis Miguel Jiménez**

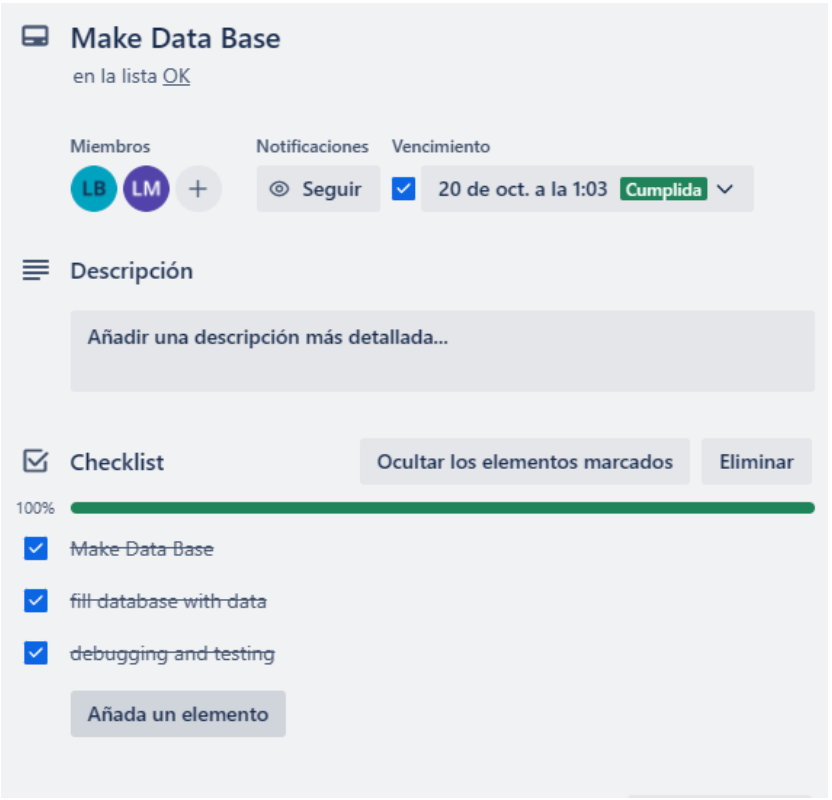
  My work on this project focused on creating the program's foundation, coding all database access and management, co-creating both the sample database for testing the program and the configuration database that stores the program's data, conducting functionality and performance testing, and eliminating all code-related errors and bugs.

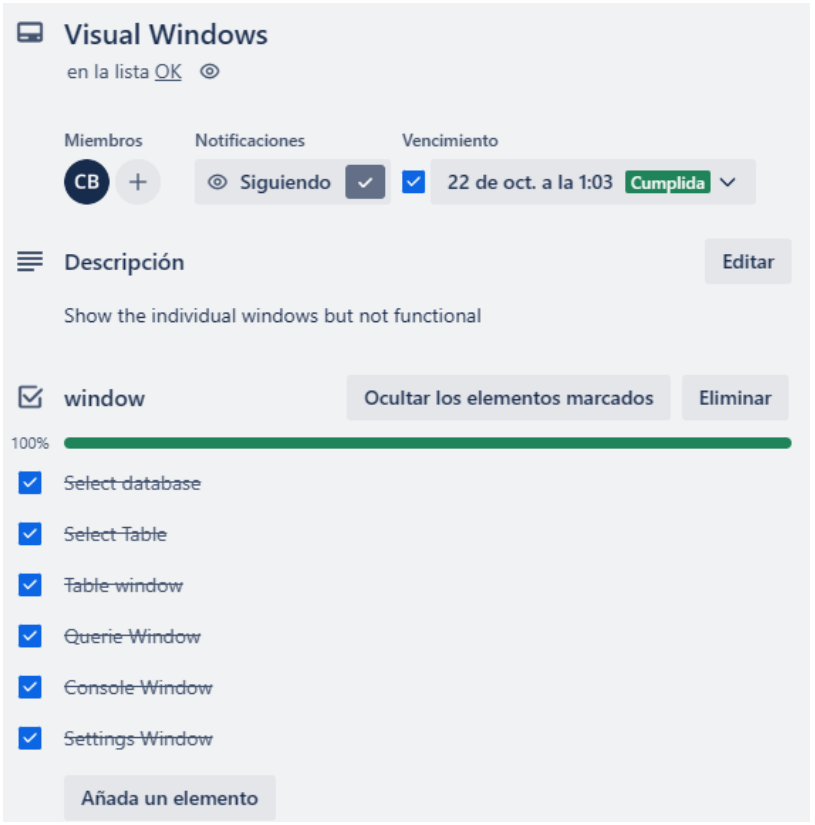
- **Carlos Mazcuñán**

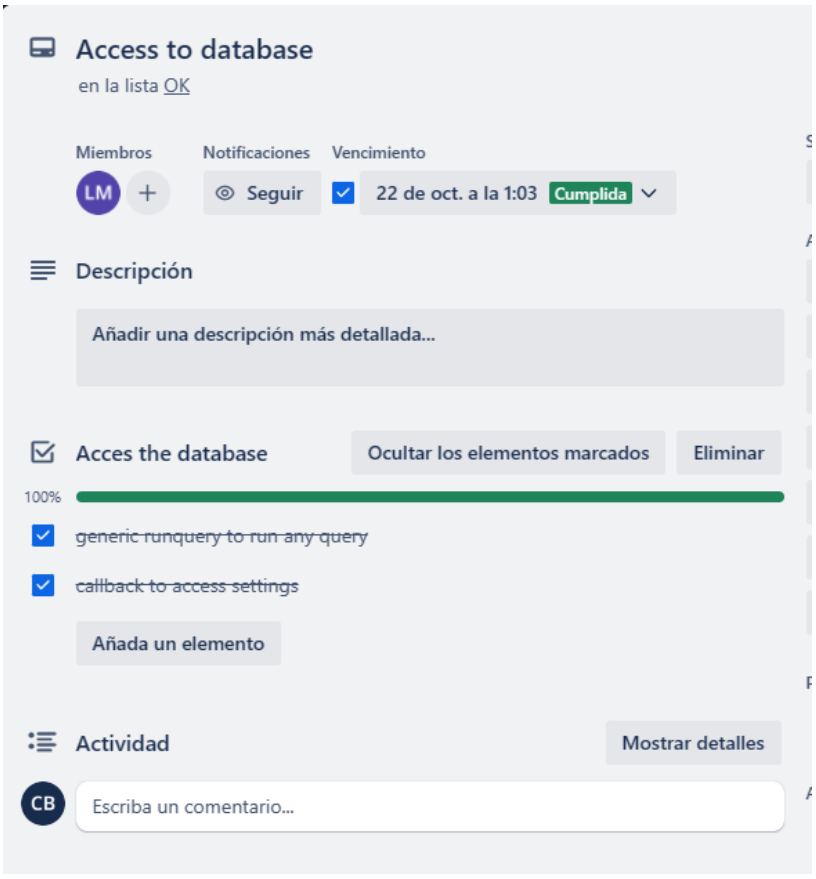
  My personal contribution to this project has been creating the visual section using the Dear ImGui Toolkit. I've designed six windows through which the user can select databases, tables, read, edit, and delete information from them, and perform queries. I've also handled the management of console messages. Following that, I performed testing and debugging task.

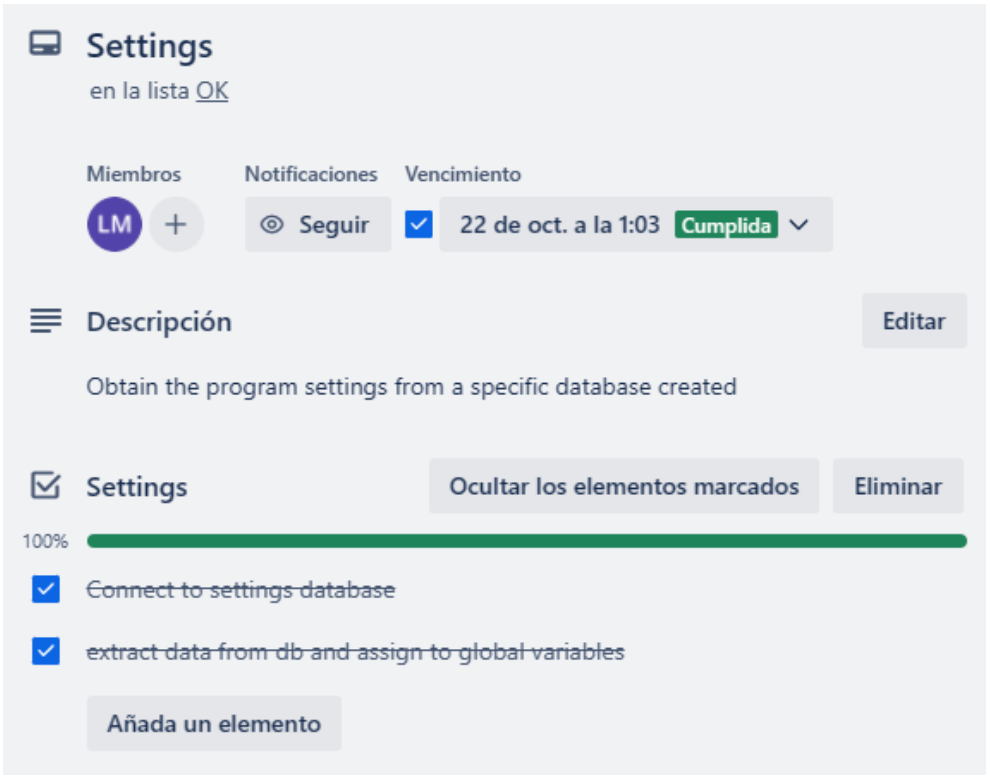


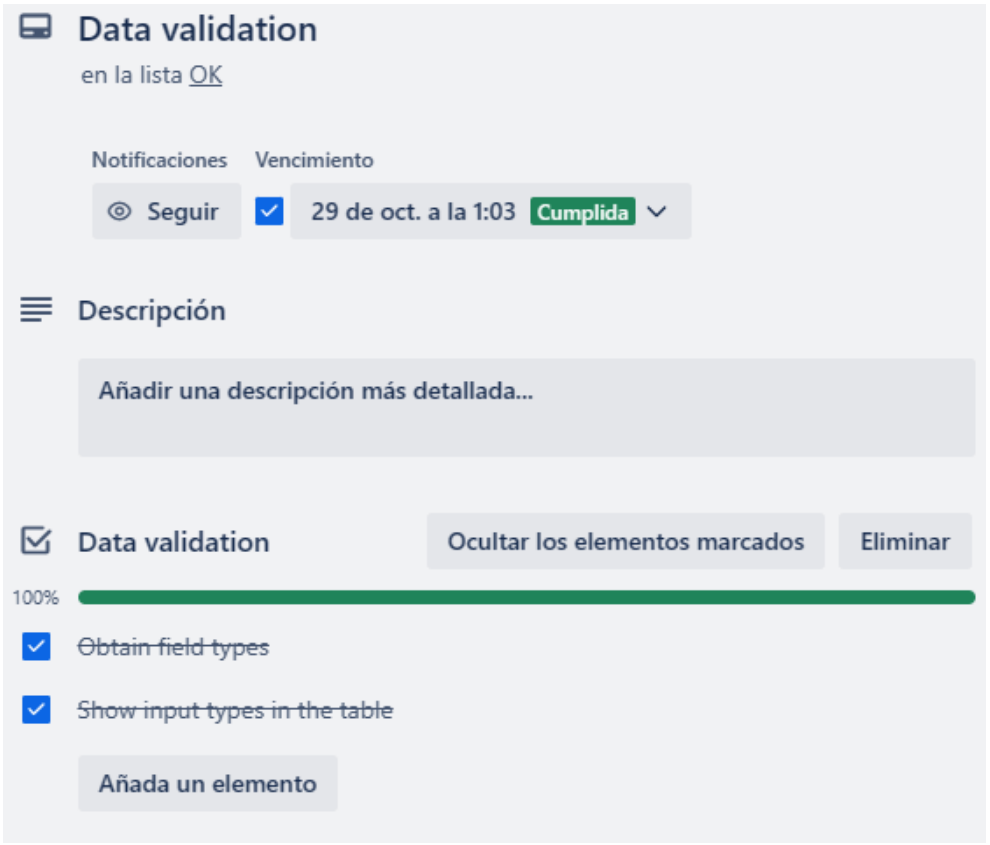

Base program
en la lista OK
Miembros
Notificaciones
Vencimiento
LM
Seguir
19 de oct. a la 1:03
Cumplida
Descripción
Editar
Creation of a base proyect with esat and base libraries to develop the database manager around
Base program
Ocultar los elementos marcados
Eliminar
100%
Empty window
Set program settings
Añada un elemento
Actividad
Mostrar detalles
CB
Escriba un comentario...

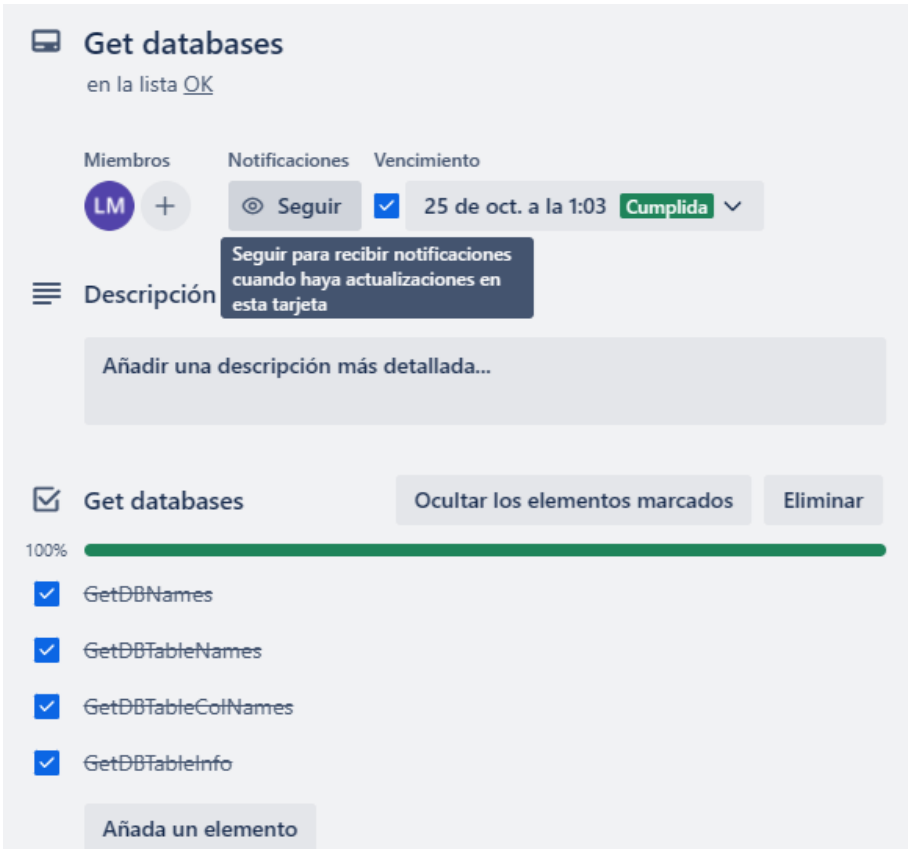

Make Data Base
en la lista OK
Miembros
Notificaciones
Vencimiento
LB
LM
Seguir
20 de oct. a la 1:03
Cumplida
Descripción
Añadir una descripción más detallada...
Checklist
Ocultar los elementos marcados
Eliminar
100%
Make Data Base
fill database with data
debugging and testing
Añada un elemento

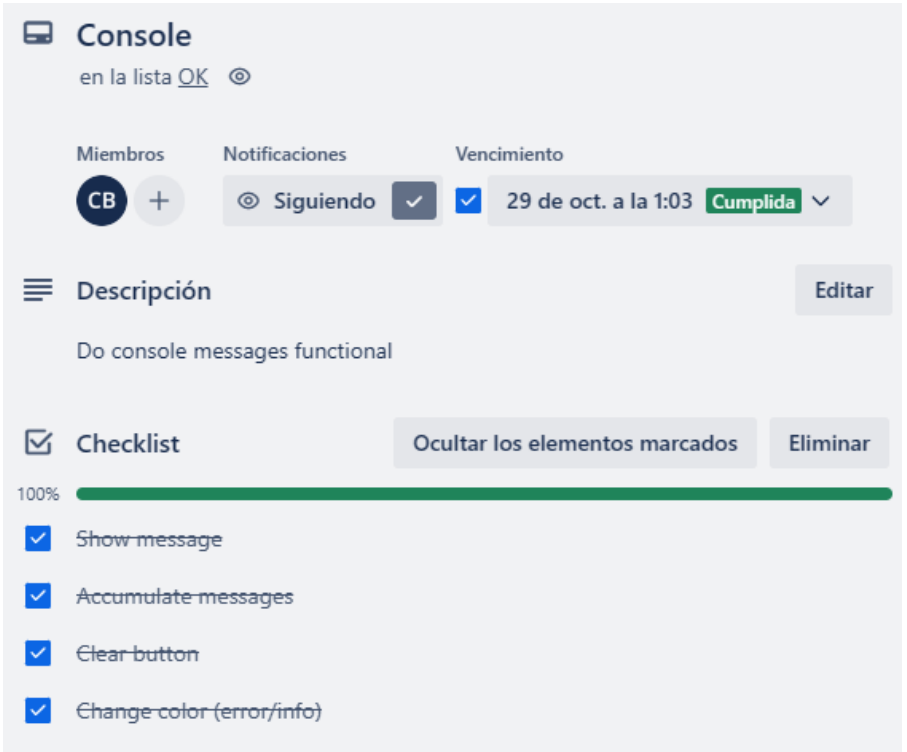

Visual Windows
en la lista OK
Miembros
Notificaciones
Vencimiento
CB
Siguiendo
22 de oct. a la 1:03
Cumplida
Descripción
Editar
Show the individual windows but not functional
window
Ocultar los elementos marcados
Eliminar
100%
Select database
Select Table
Table window
Querie Window
Console Window
Settings Window
Añada un elemento


Access to database
en la lista OK
Miembros
Notificaciones
Vencimiento
LM
Seguir
22 de oct. a la 1:03
Cumplida
Descripción
Añadir una descripción más detallada...
Acces the database
Ocultar los elementos marcados
Eliminar
100%
generic runquery to run any query
callback to access settings
Añada un elemento
Actividad
Mostrar detalles
CB
Escriba un comentario...

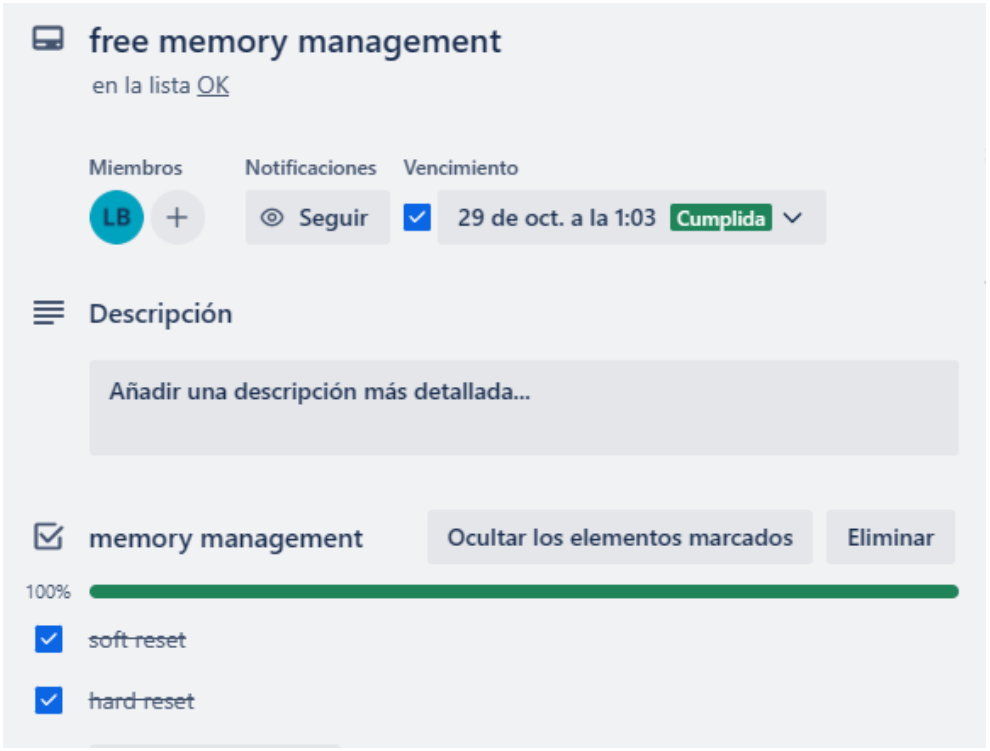

Settings
en la lista OK
Miembros
Notificaciones
Vencimiento
LM
Seguir
22 de oct. a la 1:03
Cumplida
Descripción
Editar
Obtain the program settings from a specific database created
Settings
Ocultar los elementos marcados
Eliminar
100%
Connect to settings database
extract data from db and assign to global variables
Añada un elemento

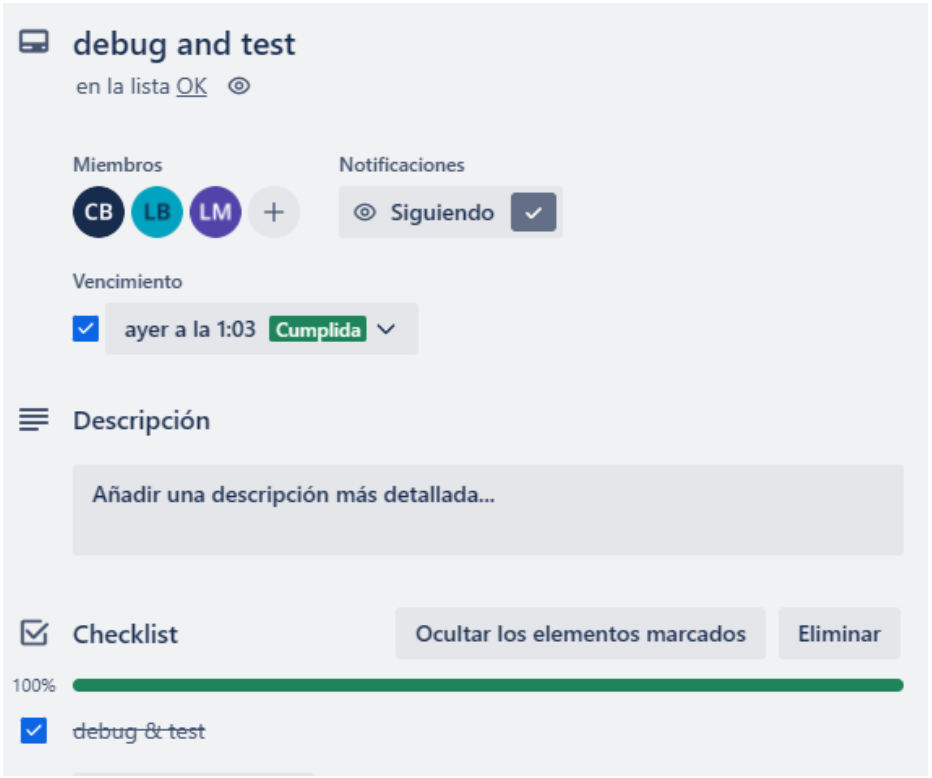

Data validation
en la lista OK
Notificaciones
Vencimiento
Seguir
29 de oct. a la 1:03
Cumplida
Descripción
Añadir una descripción más detallada...
Data validation
Ocultar los elementos marcados
Eliminar
100%
Obtain field types
Show input types in the table
Añada un elemento

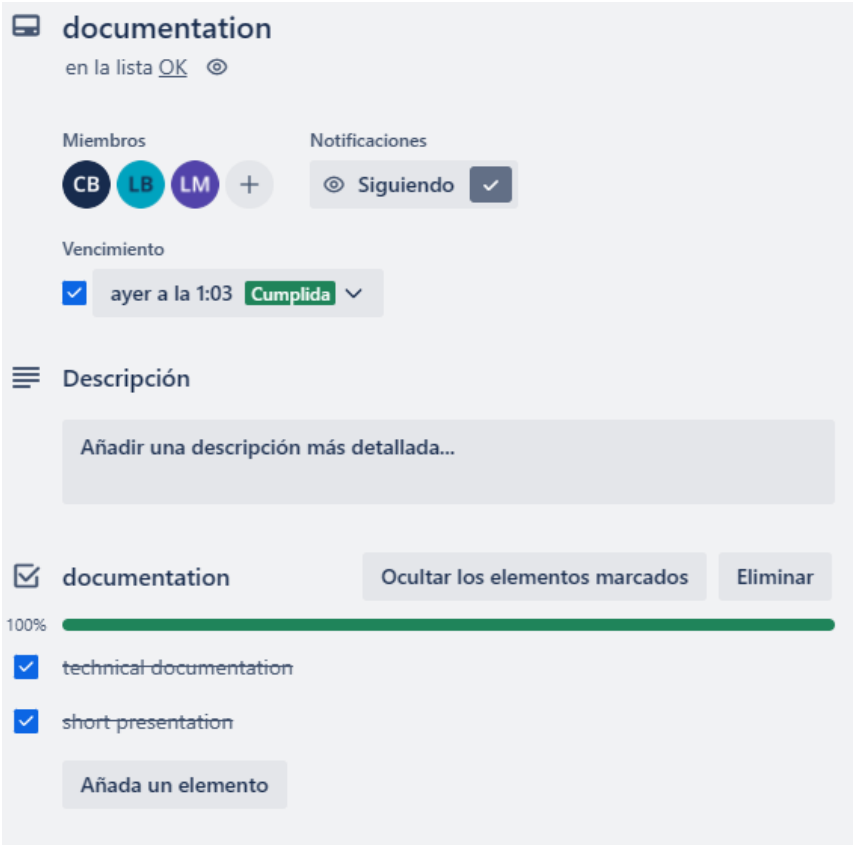

Get databases
en la lista OK
Miembros
Notificaciones
Vencimiento
LM
Seguir
25 de oct. a la 1:03
Cumplida
Seguir para recibir notificaciones cuando haya actualizaciones en esta tarjeta
Descripción
Añadir una descripción más detallada...
Get databases
Ocultar los elementos marcados
Eliminar
100%
GetDBNames
GetDBTableNames
GetDBTableColNames
GetDBTableInfo
Añada un elemento


Console
en la lista OK
Miembros
Notificaciones
Vencimiento
CB
Siguiendo
29 de oct. a la 1:03
Cumplida
Descripción
Editar
Do console messages functional
Checklist
Ocultar los elementos marcados
Eliminar
100%
Show message
Accumulate messages
Clear button
Change color (error/info)


Modify fields
en la lista OK
Miembros
LB
LM
Notificaciones
Seguir
Vencimiento
29 de oct. a la 1:03
Cumplida
Descripción
Añadir una descripción más detallada...
Modify fields
Ocultar los elementos marcados
Eliminar
100%
Modify field
Update fields in BBDD


free memory management
en la lista OK
Miembros
LB
Notificaciones
Seguir
Vencimiento
29 de oct. a la 1:03
Cumplida
Descripción
Añadir una descripción más detallada...
memory management
Ocultar los elementos marcados
Eliminar
100%
soft reset
hard reset


debug and test
en la lista OK
Miembros
CB LB LM
Notificaciones
Siguiendo
Vencimiento
ayer a la 1:03 Cumplida
Descripción
Añadir una descripción más detallada...
Checklist
Ocultar los elementos marcados
Eliminar
100%
debug & test


documentation
en la lista OK
Miembros
CB LB LM
Notificaciones
Siguiendo
Vencimiento
ayer a la 1:03 Cumplida
Descripción
Añadir una descripción más detallada...
documentation
Ocultar los elementos marcados
Eliminar
100%
technical documentation
short presentation
Añada un elemento

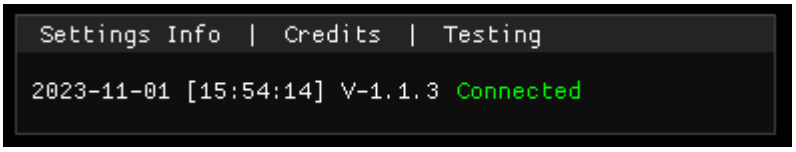
# 3.- User Manual

## Software Overview

This software application is what is usually called a database management system, the central idea is to read any database file stored in a specific folder (including every table) and present the information either navigating through the program with the different buttons at the user's disposal or making queries directly in the query window. This way the users can easily search, edit or remove information among the databases also swapping between them at will.

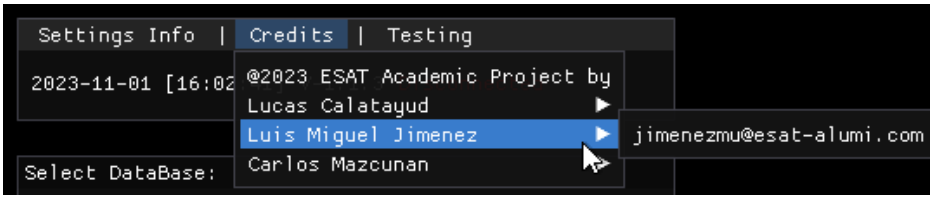
## Windows

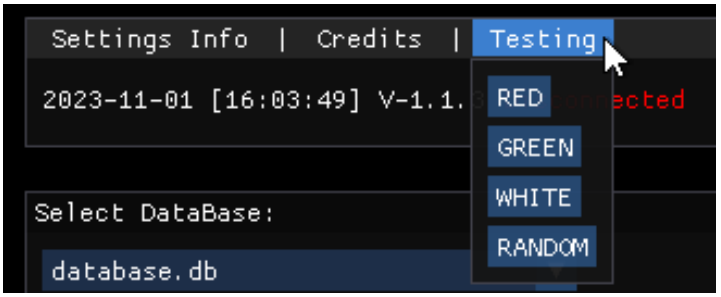
- **Settings Window**



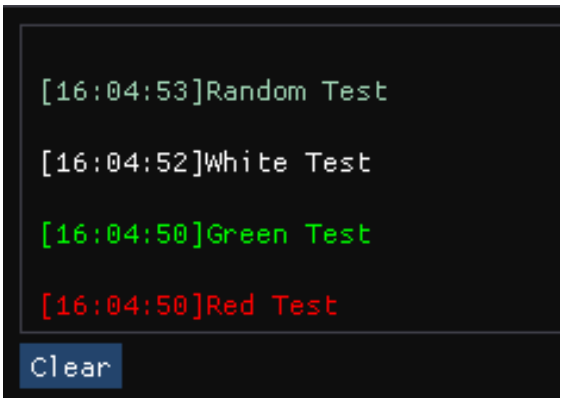

In this window, we have information about the date, the real-time clock, the application version, and a brief message indicating whether we are connected to a database or not.


At the top, we have a menu with additional information, the credits, and some testing buttons for the console.



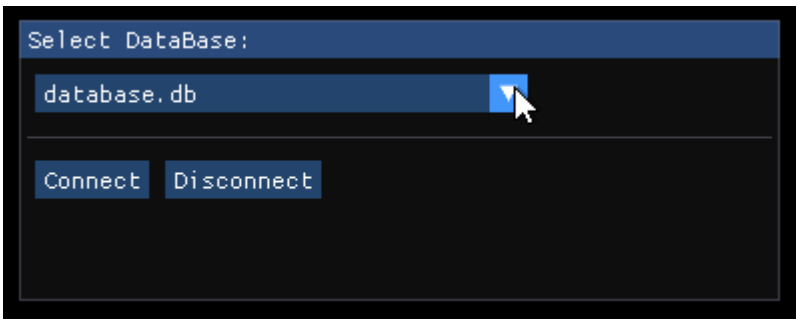




The testing buttons display the following messages through the console:

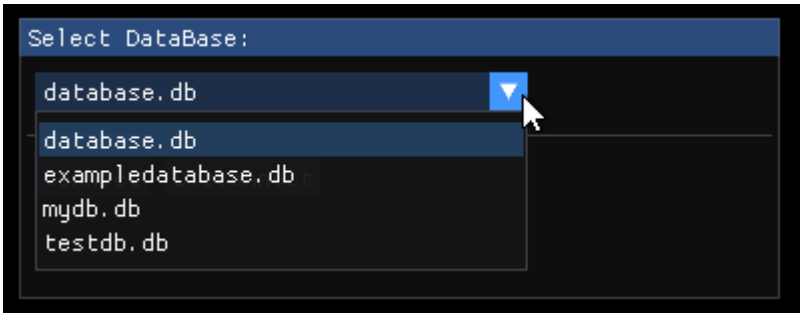


- **Database Window**

Further down, we have the window to select the database, with a dropdown that shows all the available databases, and two buttons, one for connecting and another for disconnecting.



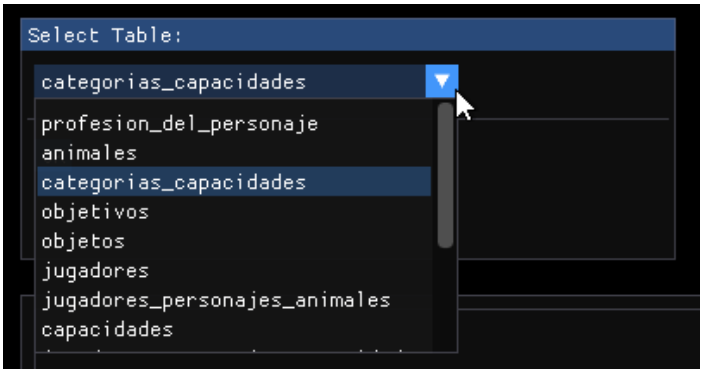

We decided to implement a procedure that uses the dirent.h library to save all the names from the data/databases folder ending with ".db" in a char pointer variable so that way we could access any database inside that folder.



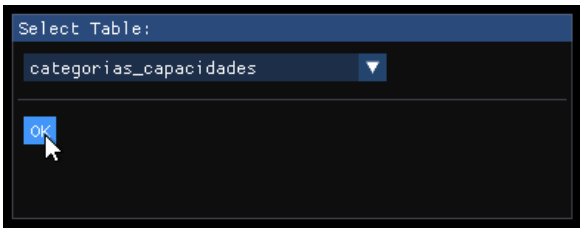

Once the database is selected in the combo and the sqlite_open function is executed we save the result code to check if everything is ok, so that if it isn't we immediately call sqlite3_close. But if it is, a message is sent to the console window

At the same time this connection occurs, we load the database table names into a variable of the program to use it both in display to select and in several other queries during the program.

- **Select table window**



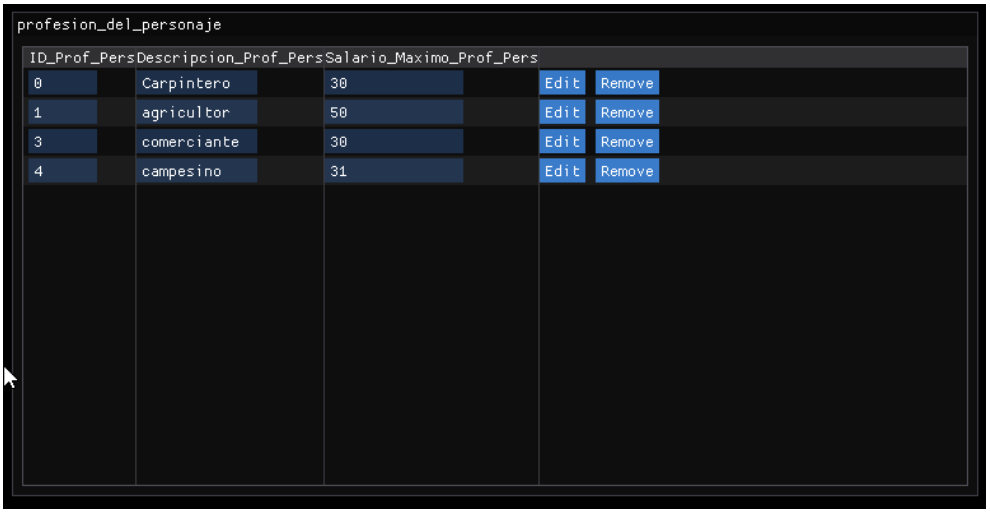

The window for selecting a table follows the same concept as the previous one: a small dropdown that displays all the tables and an "OK" button.




To connect to the table there is a requirement that it must be at least 1 column in size in order to prevent errors in connection. So if it doesn't the table would not show at all.

If it does connect, then the program runs an sqlite3 query to load the table informatión into a variable and it's displayed on the table window while a message is sent to the console window.

- **Table Window**




This window shows all the information saved in a specific variable once the user is connected to a table or has made a query through the query window. Every column represents every field that the query has returned and every row is a different entry of this query.

To modify any field a left-click with the mouse over the text input will allow the user to rewrite the entry.

Additionally, as the last column of every row there are two buttons:

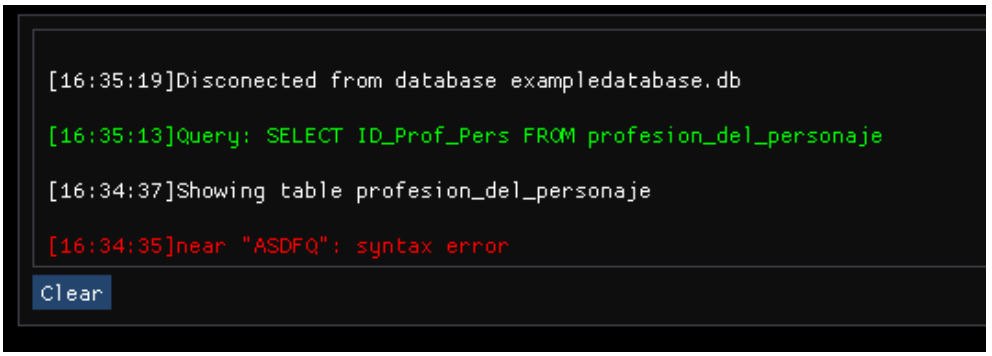
The Edit button that allows the user to upload all the modified data into the database.
If the user tries to change a field with a foreign key to another table this action will not happen but the console will not alert about it.

The remove button that will erase the entry completely. Warning: clicking this button will immediately delete the entry without question so the user must be sure of this action.

For the actions of editing and removing, we encountered an issue when trying to find the query information in the database. Therefore, we decided to create a parallel variable that would contain the values of the table without being altered by the user, so that once these queries were executed, the search would be performed with the previous information.

Additionally, both in the input of data into the table and in the output to the database, when updating the information, the data type of the table is checked to allow only certain values or characters to be permitted.

- **Console Window**



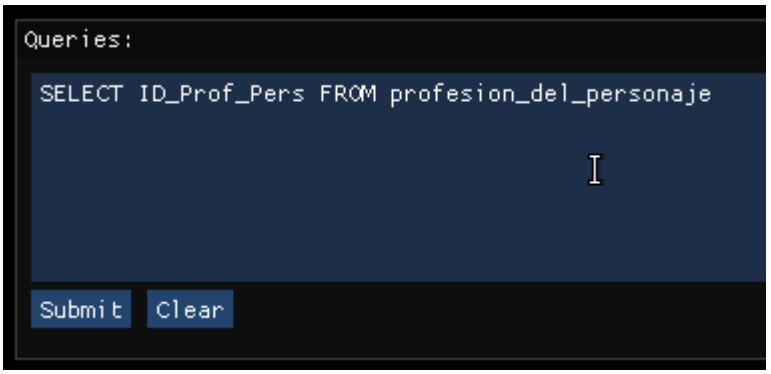

This window displays a console that provides informative messages about many actions within the application, along with the exact time of each one. At the bottom, there is a "Clear" button that will clear all the messages in the console.

- **Query Window**

Finally we have a window to process queries directly. It has two buttons:

Submit, that will send the query to the sqlite_exec function.
Clear, that will empty the input content.

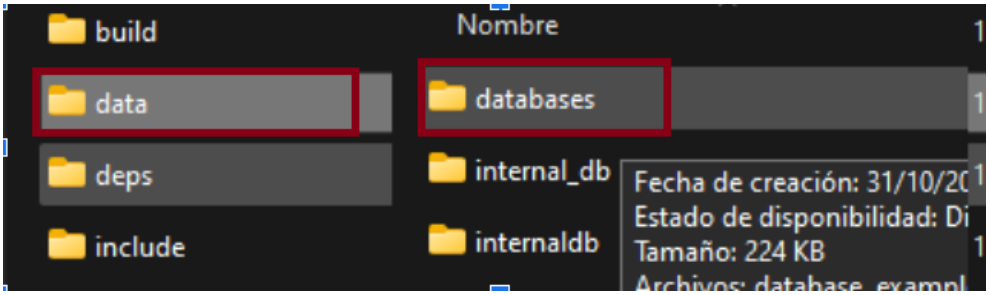
Submitting queries will send an alert of what happened to the console window in many cases, either if it went well or not and will show the result table in the table window (as long as the query meets the requirements).



## Import Database

The program is designed to display and edit data from any database. To import any database, you should place it in this path:

**.\Database_manager\data\databases**



## 4.-Conclusions and Future Work

Possible program improvements could include:

- Indication that you cannot edit a field with a foreign key.
- A wider variety of console messages, both error and informational.
- An options section where you can edit the properties of each window, manipulating the ImGui "Flags".
- Improve data validation assessing that the entries to be uploaded to the database are more limited to the exact characters in certain fields.
- Transform repeated code fragments in the program into specific functions for better clarity.
- Finalize and implement the procedure InsertDataLine() with a button and a pop up screen that allows the user to insert a new row of data into the database.

In this project the organizing has been challenging due to the differing time schedules of each team member. However, the moments when at least two group members could work together have been the most productive and efficient times for working.

We would like to express our special thanks to our classmate Carlos García Roig for the help he provided, both in contributing ideas and in assisting with error identification.

# 5. BIBLIOGRAFÍA

External libraries:

- **Imgui**: ocornut/imgui: Dear ImGui: Bloat-free Graphical User interface for C++ with minimal dependencies (github.com)
- **dirent**: (documentation) <dirent.h> (opengroup.org) (code) dirent/include/dirent.h at master · tronkko/dirent (github.com)
- **sqlite3:** libsqlite3/sqlite3.h at master · LuaDist/libsqlite3 (github.com)
- Database model made in draw.io (diagrams.net)
- Database created in SQLite Studio
- Trello: Gestiona los proyectos de tu equipo desde cualquier lugar | Trello