ERASMUS PROJECT:

Chemicals database, web page and android app

By Ariel Gómez, Pol Villar and Edgar Montero

ERASMUS, 2023

Table of contents

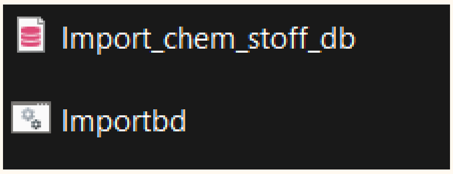
1. Database

by Pol Villar

Importing the database

In order to manage the database you first have to import the database you can choose between two ways:

Method 1: -Download the .bat file and the .sql file



Import_chem_stoff_db

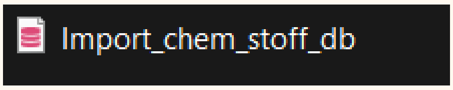
Importbd

-Execute the .bat file

-Now the database should be imported onto your mysql manager

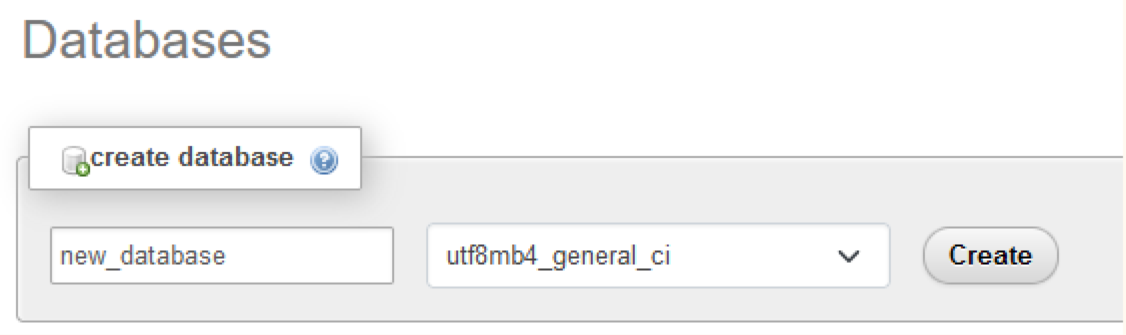
If you have phpMyAdmin you can also choose Method 2

Method 2: -Download the .sql file



Import_chem_stoff_db

-Go to phpMyAdmin and create a database



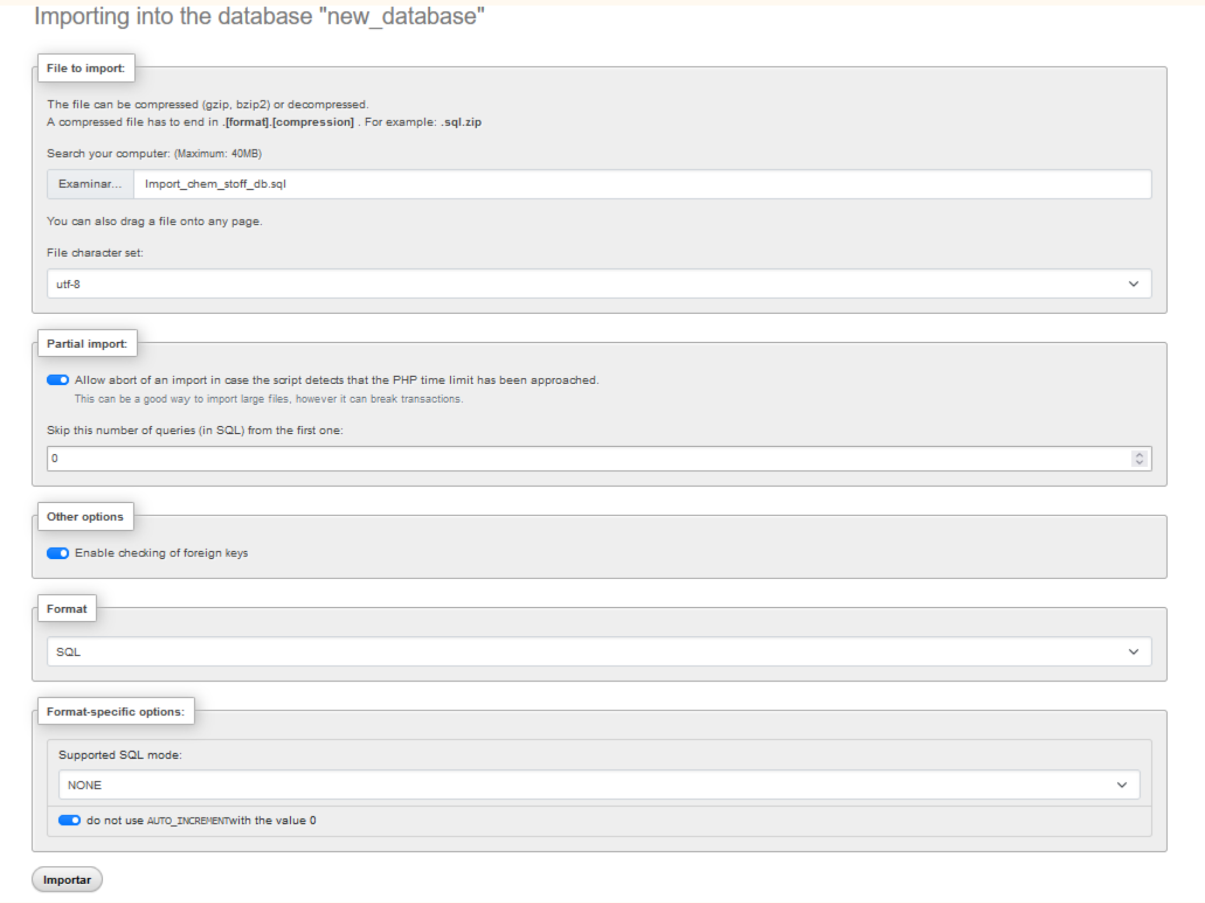
Databases

create database

new_database

utf8mb4_general_ci

-In that database click import and insert the .sql file



Importing into the database "new_database"

File to import:

The file can be compressed (gzip, bzip2) or decompressed.
A compressed file has to end in **.[format].[compression]** . For example: **.sql.zip**

Search your computer: (Maximum: 40MB)

Examinar... Import_chem_stoff_db.sql

You can also drag a file onto any page.

File character set:

utf-8

Partial import:

● Allow abort of an import in case the script detects that the PHP time limit has been approached.
This can be a good way to import large files, however it can break transactions.

Skip this number of queries (in SQL) from the first one:

0

Other options

● Enable checking of foreign keys

Format

SQL

Format-specific options:

Supported SQL mode:

NONE

● do not use AUTO_INCREMENTwith the value 0

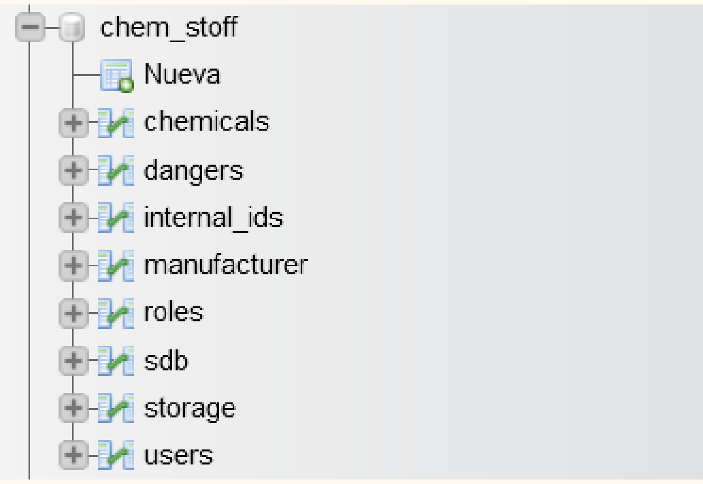
-You have now imported the database

Managing the database

Now that you have the database imported on your system you can manage the database. The following examples will be represented by the phpMyAdmin manager, but can be also replicated in Mariadb or other mysql managers.

Database explanation

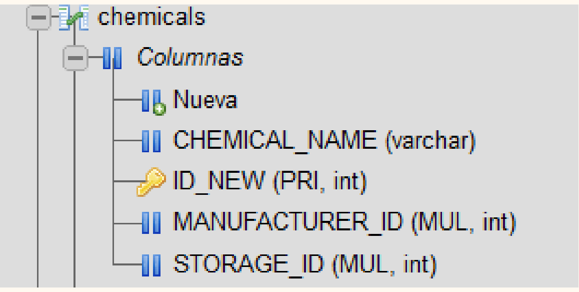
The database has six related tables for the chemical, storage and manufacturer data and all their related data; and another two related tables for the user data.



chem_stoff
Nueva
chemicals
dangers
internal_ids
manufacturer
roles
sdb
storage
users

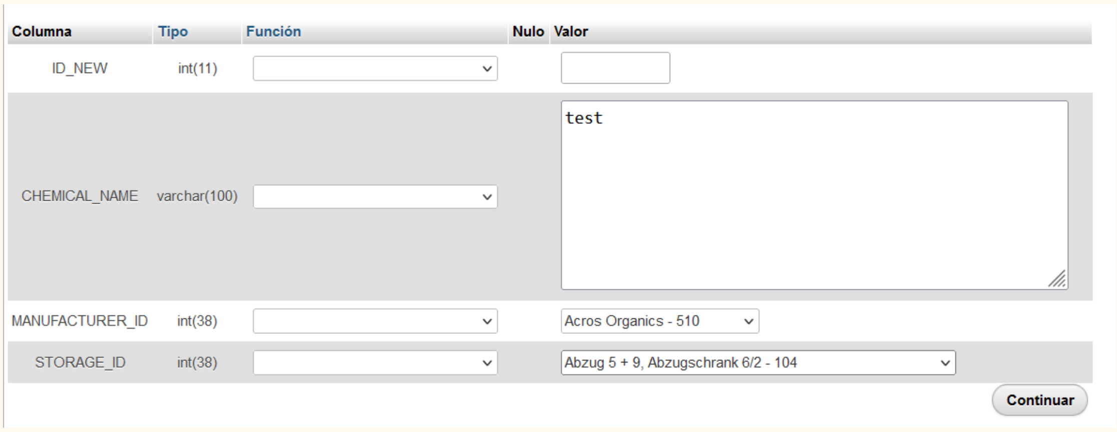
Chemical management

The chemicals have a main table **(Chemicals)** and three related tables with extra data **(dangers, internal_ids, sdb)**. The chemical table displays the information of the chemical name and its id, which ties it to the three related tables. It also has two columns **(storage_id, manufacturer_id)** that connect it to the manufacturers table and the storage table.



chemicals
Columnas
Nueva
CHEMICAL_NAME (varchar)
ID_NEW (PRI, int)
MANUFACTURER_ID (MUL, int)
STORAGE_ID (MUL, int)

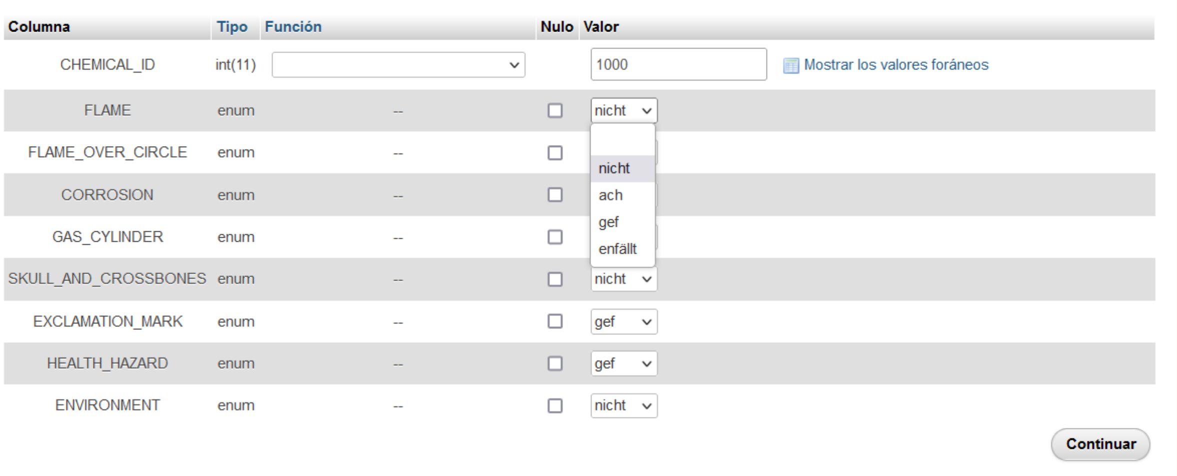
To insert a new chemical the mandatory fields are the chemical_name, manufacturer_id and storage_id; you can leave id_new blank as it's an autoincrement, and will assign a new value automatically. The manufacturer_id and the storage_id are related to the storage and manufacturer table, and so you can only input values that also exist on those tables.



Columna	Tipo	Función	Nulo	Valor
ID_NEW	int(11)			
CHEMICAL_NAME	varchar(100)			test
MANUFACTURER_ID	int(38)			Acros Organics - 510
STORAGE_ID	int(38)			Abzug 5 + 9, Abzugschrank 6/2 - 104

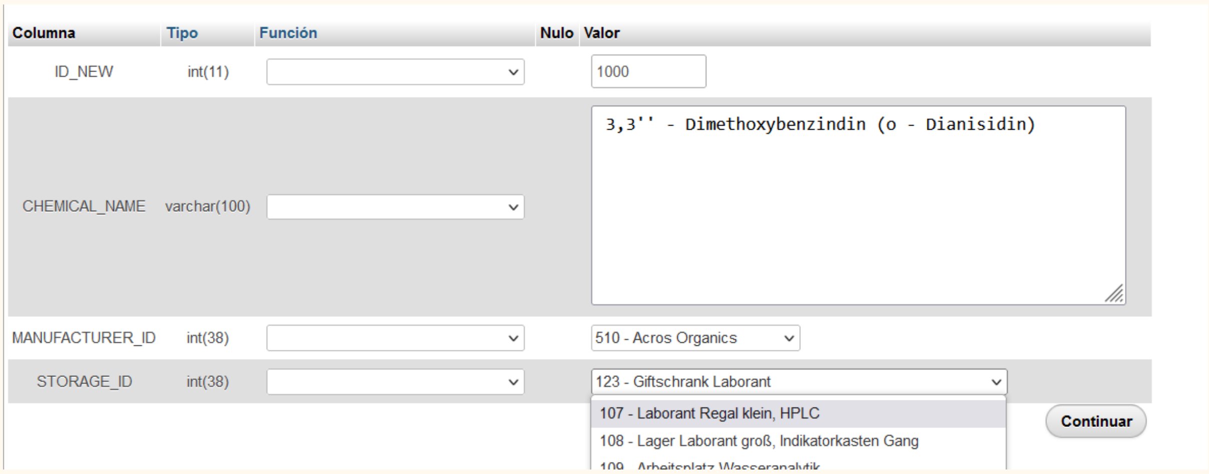
Now that we have created a new chemical, we can also fill the extra information on the three related tables: **dangers, sdb** and **internal_ids**.

Now, in order to modify an existing chemical, that's more tricky. You can only modify the values that do not interfere with the **new_id,** so the new_id (that is purely for the web page and the app to work easier with) must remain the same as when created. Other than that, you can modify every value of every table related to the chemicals, with a few exceptions: in the danger values of the **dangers** table, you can only input **gef, ach, nicht** or **enfällt.** And in the sdb table you can only insert date values into the **date** column.



Columna	Tipo	Función	Nulo	Valor
CHEMICAL_ID	int(11)			1000 Mostrar los valores foráneos
FLAME	enum	--	□	nicht nicht ach gef enfällt
FLAME_OVER_CIRCLE	enum	--	□	
CORROSION	enum	--	□	
GAS_CYLINDER	enum	--	□	
SKULL_AND_CROSSBONES	enum	--	□	nicht
EXCLAMATION_MARK	enum	--	□	gef
HEALTH_HAZARD	enum	--	□	gef
ENVIRONMENT	enum	--	□	nicht

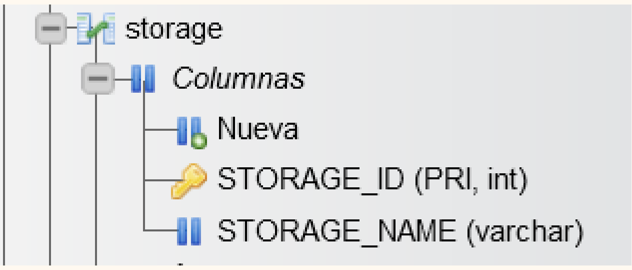
And obviously, as when inserting new chemicals, you can only input values that are preexisting for the **manufacturer_id** and the **storage_id.**



Columna	Tipo	Función	Nulo	Valor
ID_NEW	int(11)			1000
CHEMICAL_NAME	varchar(100)			3,3'' - Dimethoxybenzindin (o - Dianisidin)
MANUFACTURER_ID	int(38)			510 - Acros Organics
STORAGE_ID	int(38)			123 - Giftschrank Laborant 107 - Laborant Regal klein, HPLC 108 - Lager Laborant groß, Indikatorkasten Gang 109 - Arbeitsplatz Wasseranalytik

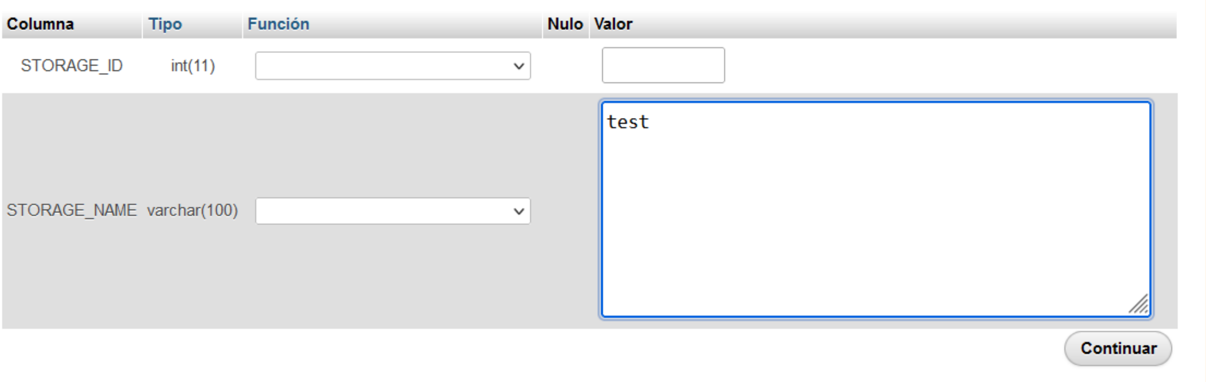
Storage management

The storages have a single table **(storage)** where the name of the storage **(storage_name)** and its id are stored **(storage_id).**



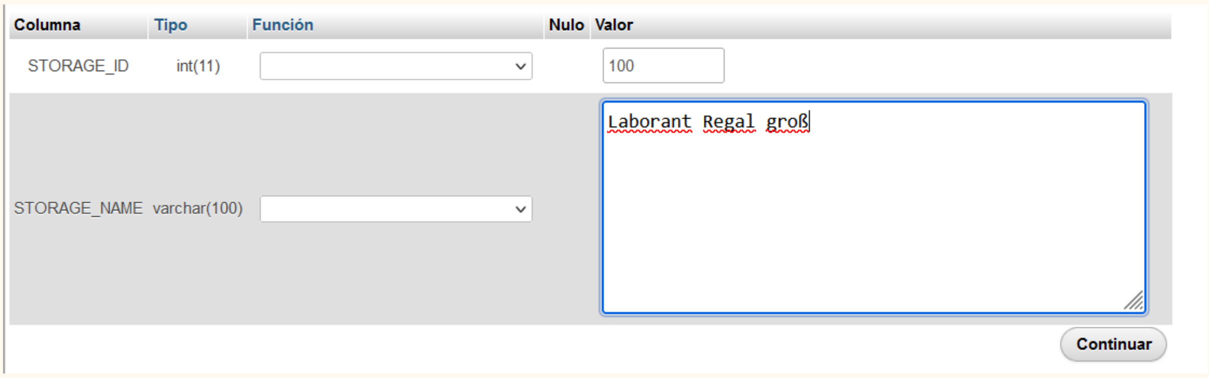
- storage
  - Columnas
    - Nueva
    - STORAGE_ID (PRI, int)
    - STORAGE_NAME (varchar)

To insert a new storage the only required value is the name **(storage_name)** as the **storage_id** is an auto increment value that will be filled automatically.



Columna	Tipo	Función	Nulo	Valor
STORAGE_ID	int(11)			
STORAGE_NAME	varchar(100)			test

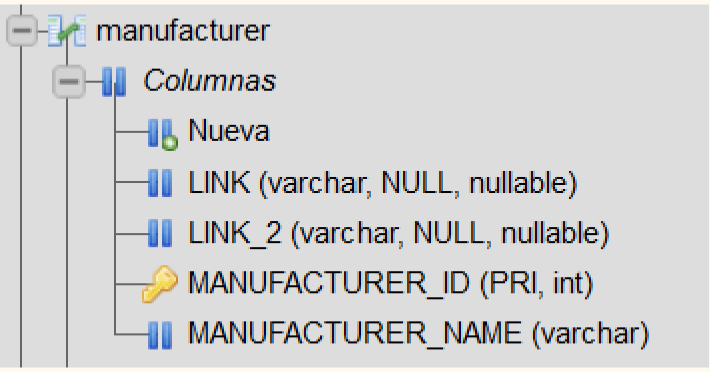
The only thing that can be modified from **storage** is the **storage_name** as the id interferes with the foreign key.



Columna	Tipo	Función	Nulo	Valor
STORAGE_ID	int(11)			100
STORAGE_NAME	varchar(100)			Laborant Regal groß

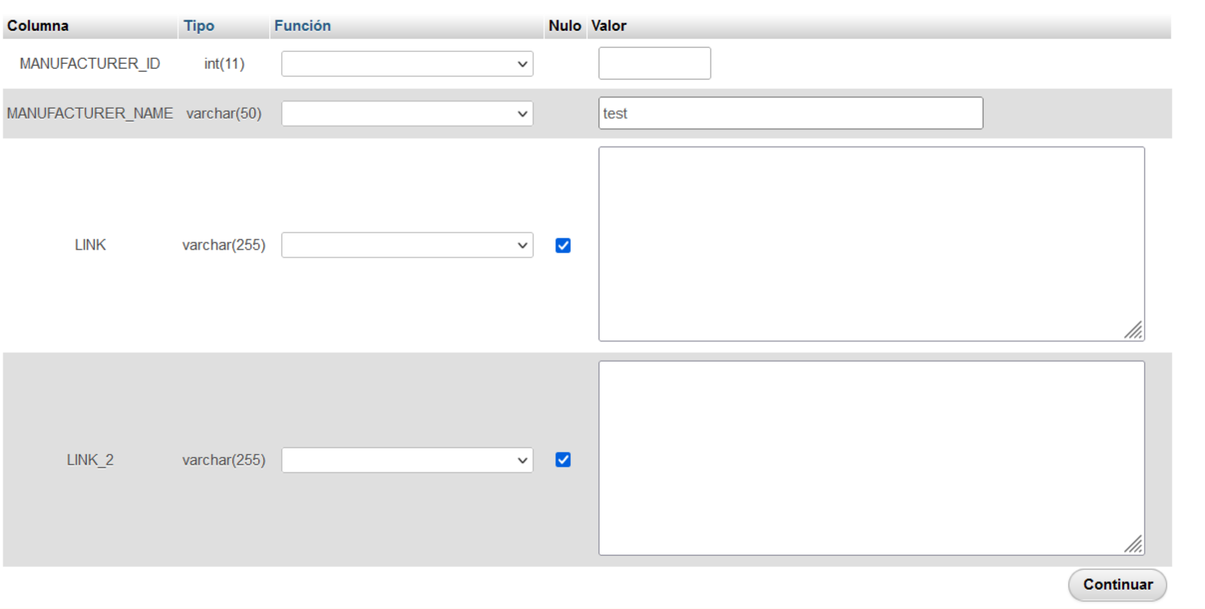
Manufacturer management

The manufacturers have a single table **(manufacturer)** where the name of the manufacturer **(manufacturer_name),** its id **(manufacturer_id)** and its links are stored **(link, link_2).**



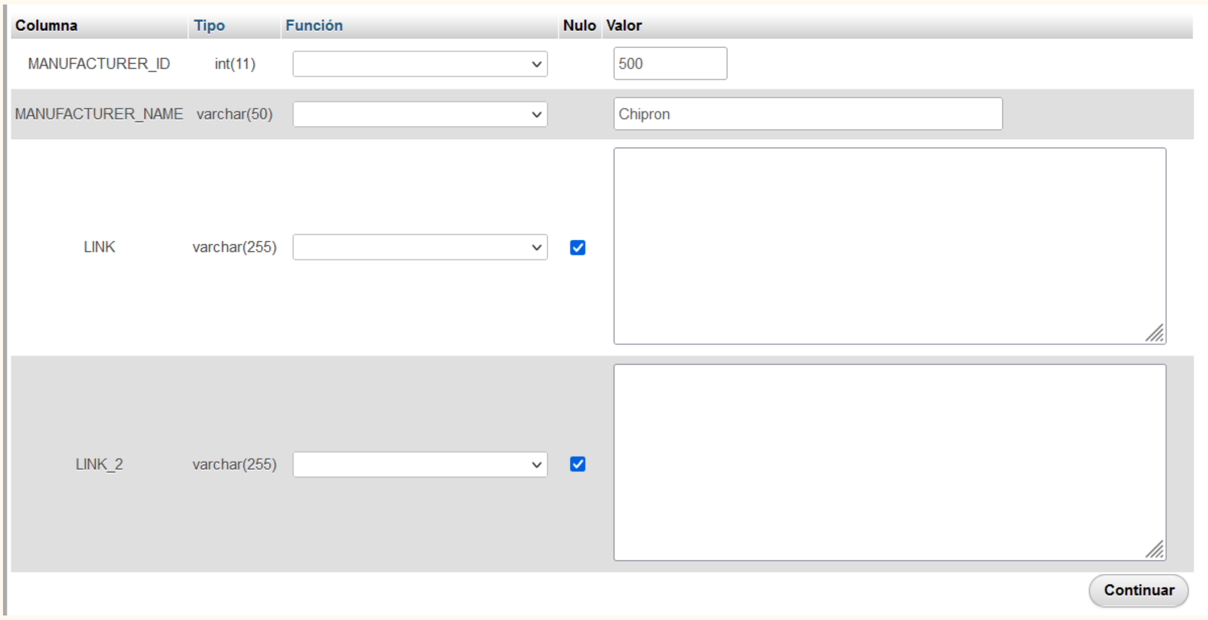
manufacturer
Columnas
Nueva
LINK (varchar, NULL, nullable)
LINK_2 (varchar, NULL, nullable)
MANUFACTURER_ID (PRI, int)
MANUFACTURER_NAME (varchar)

To insert a new manufacturer the only mandatory field is **manufacturer_name** as the links are optional and the **manufacturer_id** is an auto increment value that will be filled automatically.



Columna	Tipo	Función	Nulo	Valor
MANUFACTURER_ID	int(11)			
MANUFACTURER_NAME	varchar(50)			test
LINK	varchar(255)		☑	
LINK_2	varchar(255)		☑	

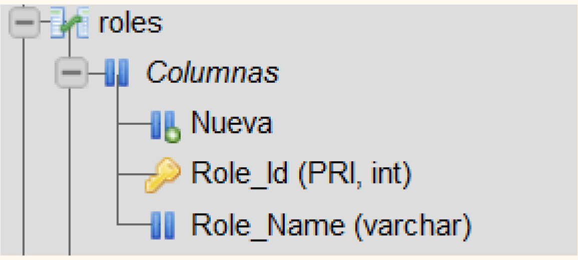
The modifiable fields of the manufacturer table are the **manufacturer_name** and the links. The **manufacturer_id** cannot be modified as the id interferes with the foreign key.



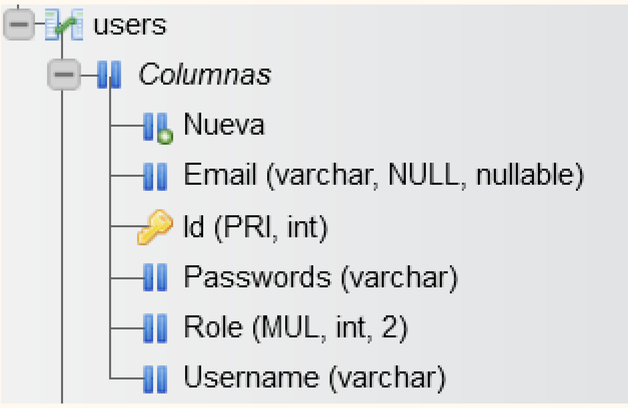
Columna	Tipo	Función	Nulo	Valor
MANUFACTURER_ID	int(11)			500
MANUFACTURER_NAME	varchar(50)			Chipron
LINK	varchar(255)		☑	
LINK_2	varchar(255)		☑	

User and role management

The database uses two tables in order to store the users and its roles: the table **users** with the fields **username, passwords, id, email** and **role** and the table **roles** with the fields **role_name** and **role_id**.



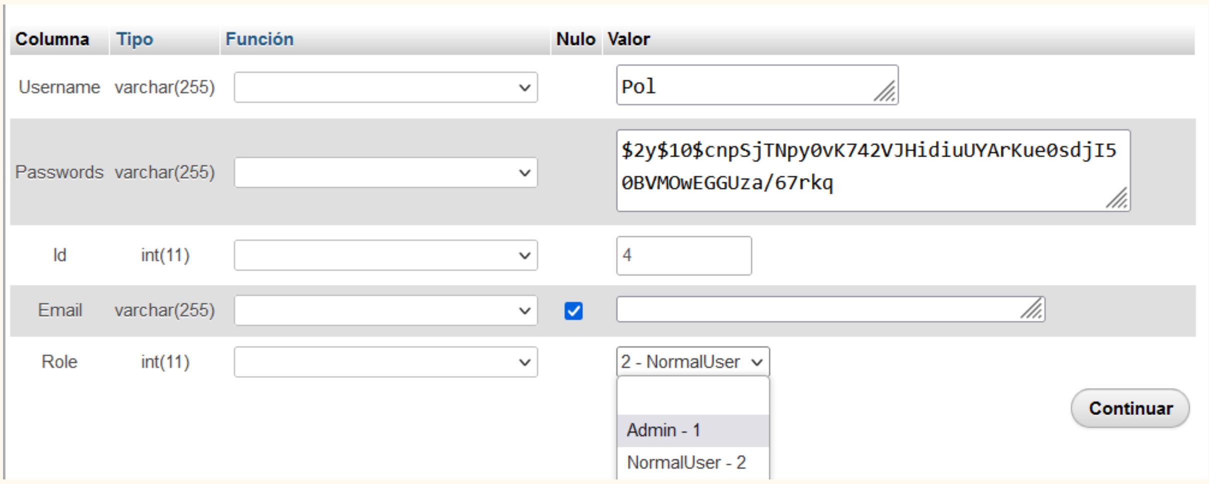
- roles
  - *Columnas*
    - Nueva
    - Role_Id (PRI, int)
    - Role_Name (varchar)



- users
  - *Columnas*
    - Nueva
    - Email (varchar, NULL, nullable)
    - Id (PRI, int)
    - Passwords (varchar)
    - Role (MUL, int, 2)
    - Username (varchar)

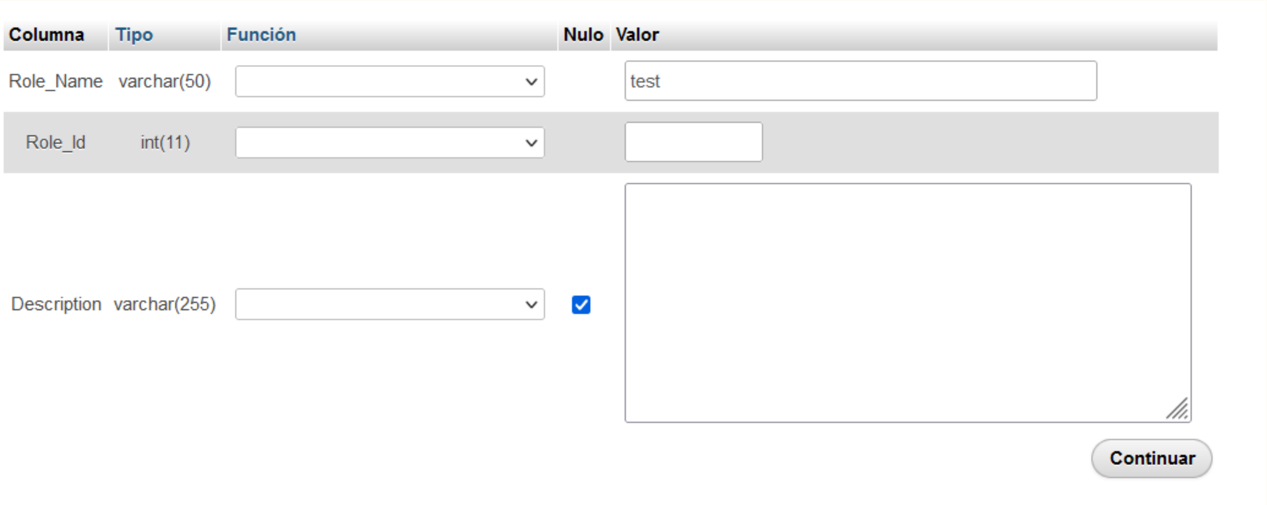
A new user cannot be created successfully using only the database directly, as the password of the new user will not be hashed by the system, and the login method will not compare the stored, unhashed password with the password typed by the user (which will be automatically hashed before checking) successfully. Instead the new users always have to be registered via the web page, and modified afterwards taking into account the following considerations:

·The newly created users are always assigned the role NormalUser by default. This means that in order to create an admin user you have to create the user via the web page and afterwards modify the **role** field on the **users** table



Columna	Tipo	Función	Nulo	Valor
Username	varchar(255)			Pol
Passwords	varchar(255)			$2y$10$cnpSjTNpy0vK742VJHidiuUYArKue0sdjI50BVMOwEGGUza/67rkq
Id	int(11)			4
Email	varchar(255)		☑	
Role	int(11)			2 - NormalUser Admin - 1 NormalUser - 2

·A new role can be created by inserting a new **role_name** into the roles table, but in order for it to have any real use, it has to be coded onto the webpage, to restrict or allow users with the role to do different functions or access certain pages.. The id is not mandatory because it is an autoincrement value that will automatically be filled.



Columna	Tipo	Función	Nulo	Valor
Role_Name	varchar(50)			test
Role_Id	int(11)			
Description	varchar(255)		☑	

·The **role_id** cannot be modified as it would interfere with the foreign key. The description field is not mandatory but helpful.

·The only mandatory fields in the users database are **username, passwords, id** and **role.** Neither of those fields can be deleted or left blank.

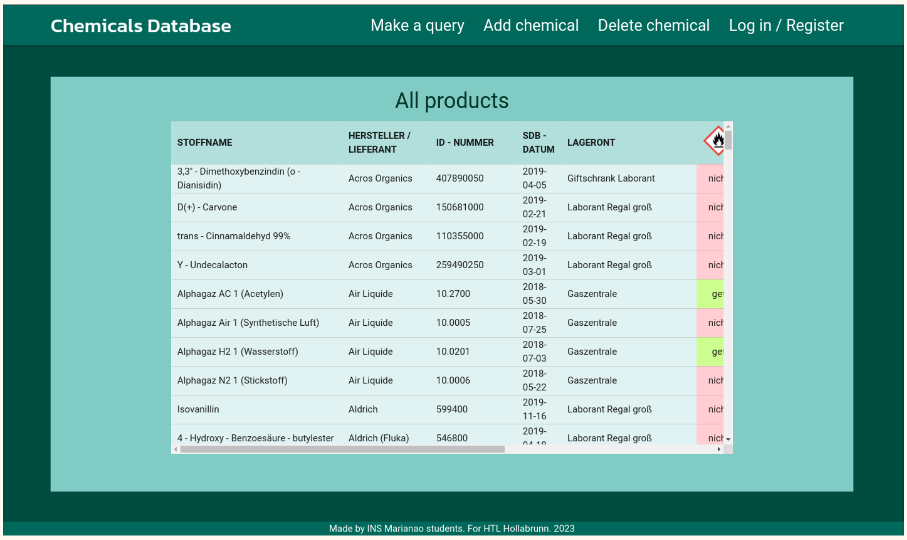
2. Web page –Chemicals Database–

by Ariel Gómez

Web page usage

「Index」page

When accessing the webpage, the index.php is opened. In it, the current database is displayed, with the relevant information of each product. On the left of the nav-bar is the name of the webpage (Chemical database), and pressing it redirects to this very file. On the left of the nav-bar are the different buttons that redirect to the functions of the page (namely making a query, adding a product, deleting a product, and logging in/registering a new user).



All products

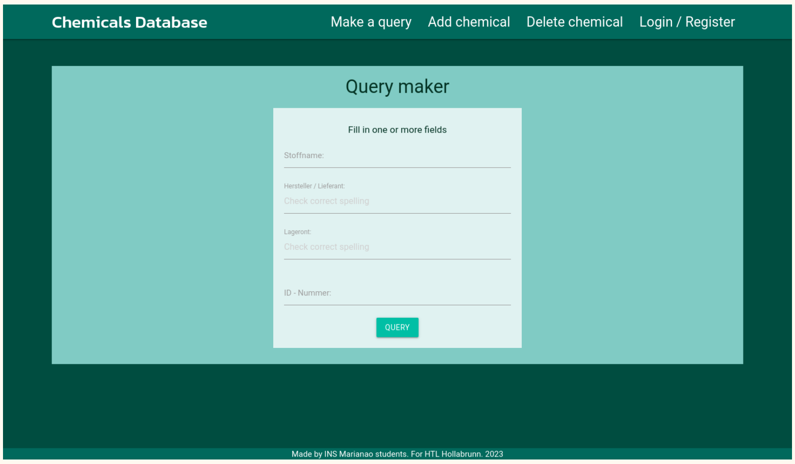
STOFFNAME	HERSTELLER / LIEFERANT	ID - NUMMER	SDB - DATUM	LAGERONT	🔥
3,3" - Dimethoxybenzindin (o - Dianisidin)	Acros Organics	407890050	2019-04-05	Giftschrank Laborant	nich
D(+) - Carvone	Acros Organics	150681000	2019-02-21	Laborant Regal groß	nich
trans - Cinnamaldehyd 99%	Acros Organics	110355000	2019-02-19	Laborant Regal groß	nich
Y - Undecalacton	Acros Organics	259490250	2019-03-01	Laborant Regal groß	nich
Alphagaz AC 1 (Acetylen)	Air Liquide	10.2700	2018-05-30	Gaszentrale	get
Alphagaz Air 1 (Synthetische Luft)	Air Liquide	10.0005	2018-07-25	Gaszentrale	nich
Alphagaz H2 1 (Wasserstoff)	Air Liquide	10.0201	2018-07-03	Gaszentrale	get
Alphagaz N2 1 (Stickstoff)	Air Liquide	10.0006	2018-05-22	Gaszentrale	nich
Isovanillin	Aldrich	599400	2019-11-16	Laborant Regal groß	nich
4 - Hydroxy - Benzoesäure - butylester	Aldrich (Fluka)	546800	2019-04-18	Laborant Regal groß	nich

Made by INS Marianao students. For HTL Hollabrunn. 2023

Image 1: index.php

「Make a query」page

The query page displays a form with four different fields. A query can be made by filling any field. It is also possible to fill more than one field, further refining the query results. Note that while the given values can be partial, they need to have the correct spelling, otherwise the query will fail.



Query maker

Fill in one or more fields

Stoffname:

Hersteller / Lieferant:
Check correct spelling

Lageront:
Check correct spelling

ID - Nummer:

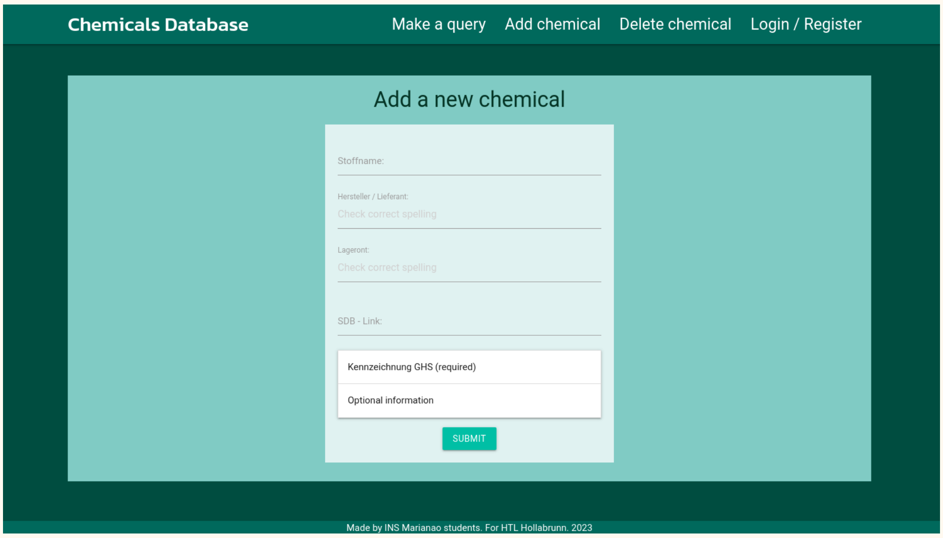
Made by INS Marianao students. For HTL Hollabrunn. 2023

Image 2: queries.php

If the search is successful, the form will disappear, and all products that match the search criteria will be displayed on a table. The displayed information is the same as the information seen on the index page. Under the table, a “new query” button will appear, letting the user make more queries. If there are no products that match the criteria, an error message will be displayed under the form, instead of it disappearing. This lets the user make a new query automatically,

「Add chemical」 page

The add chemical page has a form with all the different fields that make up each product. The first four fields are required in order to be able to successfully add the product (a product name, the manufacturer name, the storage name, and the link to the safety data sheet (SDB in german), as is choosing a signal word for each hazard pictogram. The fields under the “optional information” are optional, and include the ID used by the manufacturers, the date of the last revision of the SDB, a link to the product on the manufacturer webpage, and a second, alternate link.



Add a new chemical

Stoffname:

Hersteller / Lieferant:
Check correct spelling

Lagerort:
Check correct spelling

SDB - Link:

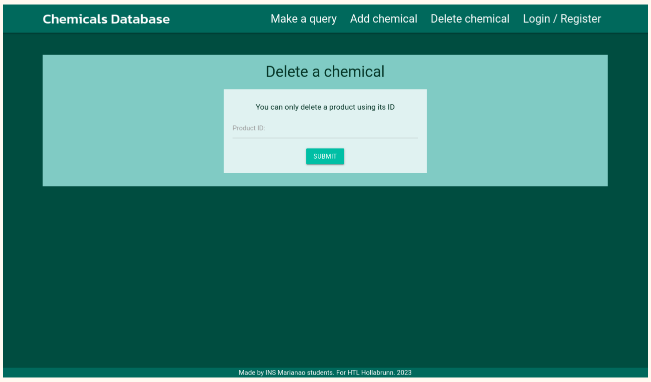
Kennzeichnung GHS (required)

Optional information

Made by INS Marianao students. For HTL Hollabrunn. 2023

Image 3: add.php

When submitting the form, the page will check each field, and inform if there is any missing critical field. In that case, an error message will be displayed. It is worth mentioning that the same product can be listed multiple times as long as they have different manufacturers. The manufacturer and storage names inputted can be new, and doing so will add them into their respective tables inside the database. Only an admin can make changes to the database.



Delete a chemical

You can only delete a product using its ID

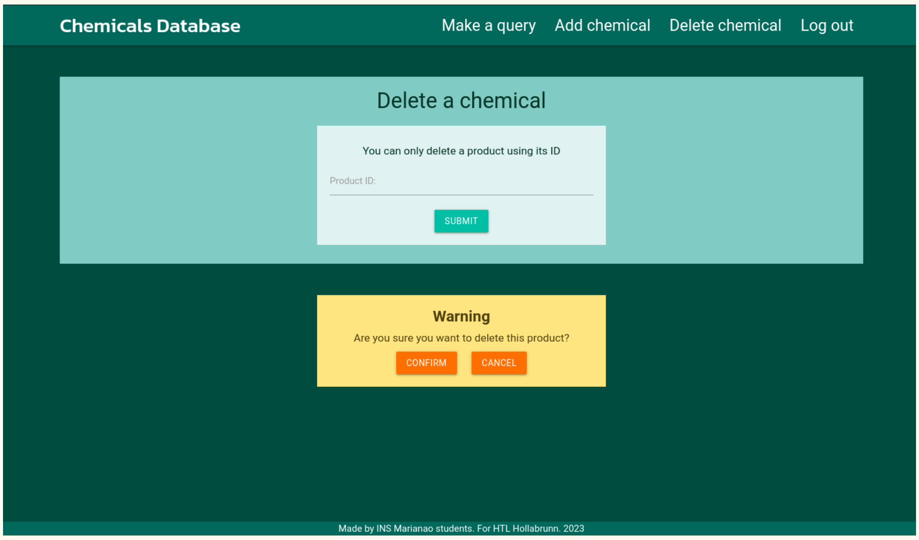
Product ID:

Made by INS Marianao students. For HTL Hollabrunn. 2023

Image 4: delete.php

「Delete chemical」page

In the delete chemical page, there is a form with a single field and a submit button. In order to delete a chemical, the chemical's ID must be used. After submitting a chemical ID, a warning message will be displayed. This is to ensure that the deletion is deliberate, and not a mistake. If the deletion is successful, a green message will be displayed. If the ID does not exist, or there was no ID submitted, the corresponding error message will be displayed. ONly an admin can make changes to the database.



Delete a chemical

You can only delete a product using its ID

Product ID:

Warning

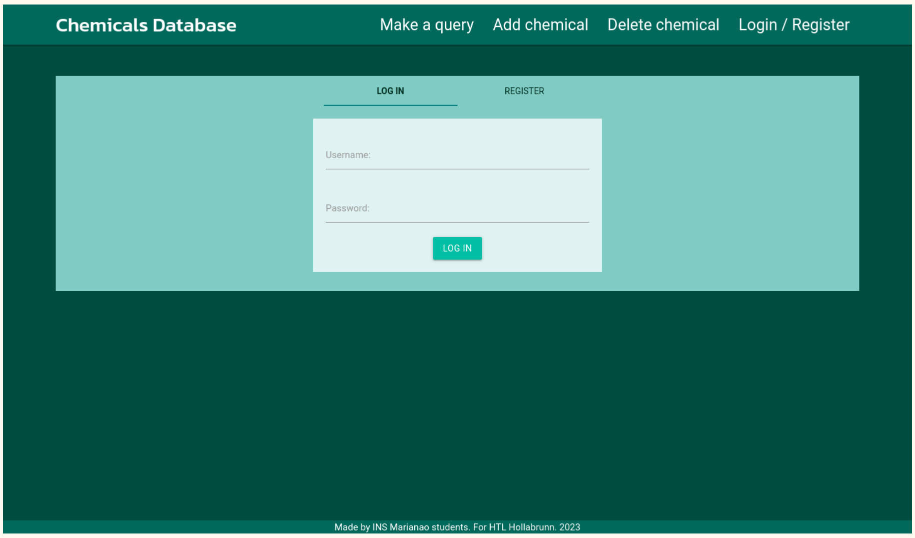
Are you sure you want to delete this product?

Made by INS Marianao students. For HTL Hollabrunn. 2023

Image 5: delete.php (with warning message)

「Login / Register」page

This page shows two different tabs: A "login" tab and a "register" tab. The login tab lets a user log into their account. a form with the username and the password fields will be displayed. If the credentials don't match, or the username doesn't exist, the message "wrong username or password" will be displayed. If the login is successful, there will be a green message displayed.



Username:

Password:

Made by INS Marianao students. For HTL Hollabrunn. 2023

Image 6: login.php

The register tab is to be used by a new user. It will let them create an account. As of now, only a username and password is needed to register. When registering a new user, the user will be given the “normal user” role by default. For now, this default role works the same as a “guest”. Both have access to all webpages, but can not modify the database through the “add chemical” or “delete chemical”. If they were to try, an error message saying “you are not authorized to add/delete a chemical” will be displayed.

Right now, only the “admin” role has enough clearance to submit both forms, and only the default “Admin/admin” user has this role assigned. To assign the “admin” role to a new user, it must be done by editing the users table on the database.

Managing php files

The current webpage has the following files and folders:

```
☒ Project (root folder)
├──☒ danger_img
│  └──🖼 (GHS pictograms)
├──☒ img
│  ├──🖼 logo.png
│  └──🖼 favicon.ico
├──☒ templates
│  ├──☒ db_conn.php
│  ├──☒ header.php
│  ├──☒ footer.php
│  └──☒ styles.css
├──☒ index.php
├──☒ add.php
├──☒ delete.php
├──☒ queries.php
├──☒ login.php
├──☒ logout.php
└──☒ functions.php
```

db_conn.php

This is the cornerstone of the webpage. This file is the one with all the credentials to connect the webpage to the database. Right now the webpage is linked to the database through a localhost (XAMPP). This will be the first file that needs to be edited to ensure the correct operation of the webpage.

header.php

This file comprises the links to the materialize CSS framework, different google fonts, the custom CSS stylesheet, the link to the Jquery library, the JavaScript library, the PC navigation bar of the webpage, and the mobile navigation bar (only one navigation bar will be displayed at a time, depending on the screen size). This file is included on all other files, ensuring the integrity of the nav-bar.

footer.php

This file consists of the footer section of the webpage, along with all the Jquery scripts needed to animate and display the UI elements correctly. This file is also included on all other files, so as to have the scripts always loaded and keep the footer integrity.

index.php, queries.php

These files are quite similar. Both make use of prepared statements to fetch all relevant data from the database. The index.php file uses a select query to fetch all products, and displays them on a table. The queries.php file, instead, prepares a default select query to fetch all products, and adds multiple where clauses based on the user inputs. Then the query result is displayed on a table, if there is at least one row that matches the search criteria. Otherwise, an error message will be displayed.

To enhance readability for the user, both tables feature a sticky table header and chemical name column. This lets the user navigate through the table while always having clear visibility of the data in each table cell and its corresponding chemical.

add.php, delete.php

These files also operate in a similar manner, although in reverse order. They both first check whether or not the user is logged in, and if yes, if the user has the “admin” role. If the user does not have the “admin” role, the message “you are not authorized to add/delete a chemical”. If the user has enough clearance, then the page will prepare, bind and execute a statement based on the inputs given by the user, after sanitizing and validating the inputs. if the “insert” or the “delete” query is successful, a green message will be displayed. Otherwise, an error message detailing what went wrong will appear.

logout.php

The logout.php file handles the destruction of the session, if a user is logged in. As this file won’t appear on screen, it has no CSS styling. This file is associated with the Logout button on the navbar that only appears after the user has successfully logged in, and the button disappears upon logging out. Whether the session is successfully destroyed or not, it redirects to the index.php file.

login.php

The login.php file handles the logging in and the register form. After successfully inputting the user credentials, a session is created. This is used in order to check if a user has enough clearance to perform data-altering queries on the database. The register form checks that the desired username is not already registered on the database before committing to the database with the user's data. Both the login and the register forms hash and salt the passwords in order to ensure the security and integrity of the user's credentials. No one, not even the admin, has access to the decrypted passwords.

Ways of improvement

Currently, there is no way of editing existing data through the UI, as there is no php file that handles it. In the future, adding a new file (update.php) with a form that queries into the database, displays the information and then grabs the user-modified data to update the chemical information is strongly advised.

When registering a new user, only a username and a password is needed. Incorporating an email, name and surname fields could be advantageous, in order to keep better track of each user. In the future, different roles could have access to different webpages, and knowing who to give each role will be critical. Having an email linked to each account also makes it easier for the admin to reset a password, or the user to ask for a new one if they forget the current one.

Another improvement could be done to the add.php file. The user should be able to choose from the existing manufacturers and storages when adding a new chemical, or choose to make a new one if it doesn't exist. Currently, the user has to input manually the names of both manufacturer and storage space, and then the code checks if the given names exist on the database; if they don't exist, they are created instead. This way is prone to errors, as any misspelling will create a new manufacturer or storage, even if it was unintended. The database could become cluttered by similar-looking names, creating the necessity of cleaning and merging those names together, needing periodical maintenance.

Lastly, new php files could be added, files that are only accessible to users with the admin role. These files would give extra tools for admins to have an easier time managing both the webpage and the database, such as giving the admin role to users (or removing it), locking the database so no updates can be done to it, hiding certain products from normal users, etc.

3. Android app

by Edgar Montero

The Android app serves as a mobile interface for accessing and managing the chemicals database. It provides similar functionalities to the web page but in a mobile-friendly format. Here are the key features and functionalities of the Android app:

User Authentication: The app allows users to log in using their credentials, including a username and password. Only registered users can log in and access the app's features.

Chemical Search: The Android app provides a search functionality that allows users to query the chemicals database. The user can enter search criteria in the provided fields, such as chemical name, manufacturer name, or storage name. The app will then display the matching results based on the entered criteria.

Chemical Details: When a user selects a chemical from the search results, the app displays detailed information about the chemical, including its name, manufacturer, storage details, and safety data sheet (SDB) link. The app also shows any additional information available for that chemical.

Add Chemical: Users can add new chemicals to the database directly from the app. The app provides a form where users can enter the necessary information for a new chemical, such as the chemical name, manufacturer name, storage name, and SDB link. Users can also choose a signal word for each hazard pictogram associated with the chemical. The app validates the entered data before submitting it to the database.

Delete Chemical: The app allows users to delete chemicals from the database. Users need to enter the ID of the chemical they want to delete. After confirming the deletion, the app removes the chemical from the database.

Security and Error Handling: The Android app implements security measures to protect user data and prevent unauthorized access. It handles errors gracefully, providing meaningful error messages when necessary, such as displaying a message for incorrect login credentials or unsuccessful database operations.

The Android app complements the web page by offering a mobile solution for accessing and managing the chemicals database. It provides a convenient and user-friendly interface for users to search for chemicals, view their details, add new chemicals, and perform other database-related tasks on their Android devices.