

# Fraunhofer product tracking system



This file is a guide for anyone who means to make changes to the website for the Fraunhofer product tracking system, big or small.

The system was made for storing and tracking products for Fraunhofer CCD. The system was written using the following programming languages:

- [PHP](#)
- [HTML](#)
- [JavaScript](#)
  - [Good resource for JS](#)
- [jQuery](#)
- [CSS](#)
  - [Online tutorials for all of the above](#)
- [MySQL](#)
  - [SQL Free introduction class](#)

## Table of contents

- [Fraunhofer product tracking system](#)
- [Table of contents](#)
- [Install](#)
  - [MySQL workbench](#)
  - [MySQL community server](#)
  - [MAMP](#)
  - [Sublime](#)
  - [Grunt and NPM](#)
- [Getting started](#)
  - [Accessing the code](#)
  - [Database architecture](#)
- [Changing the code](#)
  - [Fixing minor errors](#)
  - [Changing database tables/procedures etc.](#)
  - [Changing Javascript or jQuery functions](#)
- [Guides](#)
  - [MySQL setup guide](#)
  - [MAMP setup guide](#)
  - [Searching for POs, runs or tools](#)
  - [A new PO. From adding to shipping.](#)

# Install

Below is a list of the programs that need to be installed on a new computer to start working on the system. More thorough guide on how to get the programs working can be found at the end of this documentation. Note that the server computer should have all of this already installed and you can make changes to the live database without having to download anything.

## MySQL workbench

MySQL workbench is a [IDE](#) used to write MySQL code. Free download from this [link](#).

## MySQL community server

MySQL community server is a free download that ables you to use MySQL on your computer. Download it from this [link](#).

A guide to get started can be found [here](#).

## MAMP

To have a running website connected to a MySQL database use MAMP. You can download it [here](#) for free. MAMP setup guide can be found [here](#).

## Jshint and Uglify

Jshint was used to detect errors and potential problems in the JavaScript code. Uglify was used in this project to minify all JavaScript files. This is done so that the website is as quick as it can possibly be, since the computer in the lab is outdated and the internet connection might get slow this is very important. While developing it is better to include the JavaScript file you are making changes on in the HTML/PHP views, since compiling with uglify takes a few seconds and if you are making a lot of small changes and testing them those seconds add up quickly. A more detailed guide on how to make changes on JavaScript code can be found [here](#). To run those tools we used Grunt.

# Getting started

## Accessing the code

The source code is stored on the shared folder under Eysteinn Gunnlaugsson. If changes are made to this code they will appear on the website. A source code management system called GitHub was used during development of this project. It is not necessary that you continue using it but I found it very helpful to look through my changes and it also serves as a backup for the code. At the time writing this file the code is stored [here](#) however this is most likely outdated but it might be useful to look through the commit history. If you want to continue using GitHub just make a new project on your account and push the code from the shared folder on to it.

# Database architecture

The entity relation diagram for the database was designed using [draw.io](https://draw.io) and can be found on the shared folder under Eysteinn Gunnlaugsson. I recommend changing the ER-diagram right after you make changes to the database. To change the ER-diagram just go to [draw.io](https://draw.io) and open `FraunhoferERNew.xml` that is stored on Eystein's shared folder. If you are not sure how to read a ER-diagram [this documentation](#) is a pretty good source on relationship patterns in SQL. There are many ways to show relations in a ER-diagram, I decided to use the following :

- ----- is a zero-to-many relationship.
- -----> is a zero-to-one relationship.
- ===== is a one-to-many relationship.
- =====> is a one-to-one relationship.

## Changing the code

### Fixing minor errors

To fix minor things like spelling errors, incorrect grammar etc. Follow these steps:

1. Go to the server computer
2. Open Sublime Text 2
3. Find the source code. At the time writing the code is stored on `CCL/Shared/Eysteinn Gunnlaugsson`
4. Drag the folder called `FraunhoferDBWebsite` in to Sublime Text 2
  - **Now you should have access to every single line of code in the project so make sure to not change anything else!**
5. Press `⌘-Shift-f` to search all the files for the text you want to change
  - Type in the search string you need to change
  - After pressing `Enter` a new file will open up showing **every single instance** of the text searched for in the project
  - To search only this file press `⌘-f`. If you can make your search more specific you will have less files to look through
6. When you find the text you want to change double click the yellow string above it, this is the file name containing the text.
7. Now the correct file should be open. Search this file for the error that needs to be fixed, you can do this either manually or by pressing `⌘-f` and typing the text again.
8. Fix the error and save the file, `File->Save` or `⌘-s`
9. Check the website and make sure that the changes are correct

The pictures display how to fix a spelling error of the word Employee

Step 5 

Step 5.3 



## Changing database tables/procedures etc.

Interacting directly with MySQL should only be needed if you need to do any of the following

1. Add a new table to the database
2. Add a new field to a table
3. Adding/changing procedures, triggers or functions
4. Testing queries before adding the to the website

To make changes to the database tables, procedures, triggers or functions it is required to have some background on databases so this guide will focus on explaining how and where the code is stored. The folder containing all MySQL code, at the time writing this, is stored in the shared folder under `Eysteinn Gunnlaugsson/SQL/WebPageDB/`. The file names are descriptive. The file containing SQL functions is `functions.sql` and procedures are stored in `procedures.sql` etc. To make changes to the live database you have to go to the server computer, open MySQL workbench and connect to the server called `Live Fraunhofer Database`. When a page called `Query 1` opens type in `Use Fraunhofer;` in the editor and you are ready to start making changes.

I highly recommend trying all changes first on a mock database! A guide to set up a mock database can be found [here](#).

To change SQL code for other things like inserting, deleting, updateing or selecting this code is stored in the PHP files in the project folder and can be changed from there, but again, you should test the changes on a test website before making changes to the source code.

## Changing Javascript or jQuery functions

To edit the js files you need to run a 'grunt' task after you change the files. You do that by navigating to the project folder location in the computers terminal and typing in `grunt`.

To run a Grunt task you need Node: [Node.js](#)

Npm: [Npm beginners guide](#)

And Grunt: [Grunt - Getting started](#)

Within the project folder, you need to run 'npm install' and that should install grunt and every grunt dependency you need. It does that because we have specified those dependencies in the `package.json` file.

After that you should be all set to simply run 'grunt' in the terminal and that should compress all the javascript files in the `js` folder to the minified js file in the 'dest' folder. As well as checking if your JavaScript code is 'lint free' with the Jshint tool.

Instead of running the grunt command in your terminal after each javascript file modification, you

can write 'grunt watch'. After that, every time you save your changes in a javascript file, the 'grunt' command is executed automatically.

The server computer should have everything installed for you to simply run the grunt command in the terminal after you edit the js files

## Guides









### MySQL setup guide

After both the MySQL community server and MySQL workbench have been installed you can set up a connection to a new database.

To allow MySQL connections on OSX, go to system preferences and search for MySQL. From there it is possible to start a connection. To fill this database with the same data as is on the real running Fraunhofer database take the newest database dump (at the time writing it is stored on the shared drive under Eysteinn Gunnlaugsson. File name 'Dump + date created') and import it to a new server. You might need to add `USE Fraunhofer;` to the top of the file if you get an error saying that this database does not exist (see picture 5). You can now play around with this database to get to know the system without it having any effect on the real running database.

**While developing I highly recommend testing everything on a mock database before saving the changes to the source code for the project.**

Here is a picture guide on how to create a copy of the real database.

1. Create a new connection in MySQL. 
2. Click test connection to make sure everything is OK. 
3. Click Data Import/Restore. 
4. Select Import from self contained file and pick the newest dump file. 
5. If there are no errors you should get a message like this. 
  - If you get an error saying that this database does not exist do the following changes to the dump file.
    - Before 
    - After (see line 23): 
6. Try running queries to see if you have data in your database. 

### MAMP setup guide

Setting up MAMP is pretty straight forward. It is a good idea to set MAMP up to a mock database so you can test code there instead of the live database. After you download MAMP and setup MySQL there are a few settings that need to be configured:

- Set apache port to any available port. This will be the number you type in the URL to connect to the database. 8889 or 8887 should be free
  - It might be a good idea to pick a port number very different from the actual database so you do not accidentally mess with the live database. I used 3306 for my mock database. The live

one is 8888

- Set MySQL port to the same port as your mock database
- Set PHP version to 5.6.2
- Set Web Server to Apache
- Set the document root to the folder that your code is stored in

The URL needed to access the webserver is a combination of three things:

- Your computers name or IP-address
  - You can find your computer name by going to `System Preferences/Sharing`. At the top you can see the computers name and an `Edit` button. Click the edit button and there you can see your computers webserver name. The one I use is `eysteinn.local`
  - You can see your IP-address [here](#)
- The port number you picked for MAMP, not the MySQL port number
- The filepath from the project root folder to the file you want to open

The URL should look something like this `http://35.9.146.192:8888/Login/login.php` or `http://eysteinn.local:8888/Login/login.php`.

You should now have a connection to the database you just made through google chrome. You can only have one active web server running per computer so do not change this if you are working on the server computer since it will remove access for other computers to the main database!

## Search for POs, runs or tools

All the search functionality of the webpage is found if you click `Tooling overview` on the front page. You can do a basic search by adding information to the filters on the page. There are two symbols you can use to narrow your search even more. You can use `%` and `_` as wildcard characters.

`%` replaces zero or more characters. `_` replaces exactly one character. Here are a few examples where this can be useful.

- `%rplcmt%` searches for all POs that contain the string `rplcmt`.
- `%-%` searches for all POs that contain a dash.
- `k21506__2` will show all runs in K2 in June 2015 that were the second run of the day.
- `_____1` will show first run of the day for all machines.
- `%drill%` will show all tools that contain the string `drill` in their tool ID

## A new PO. From adding to shipping.

The following videos show the full life span of a PO in the database, from adding it to shipping it. Click the videos to enlarge them, you can also zoom in on your browser.

## Adding a new PO to the database



- You can also use the shortcut button `Add PO` in the header of the website.
- The initial inspection should be ok or OK if everything is good, if not write a short description of what is wrong. I.e. "missing tools" or "broken box on 5 tools rest OK".
- Number of lines is the number of different tools on this PO.
- Make sure you click `Add PO` before you click `Add tools to PO`.

## Adding tools to a PO



- Fill out the information as it is on the PO received from the customer.
- Inserting a diameter and length generates the price for each customer.
- Having coating as DLC or checking `Double ended` will double the price.
- You can edit the auto generated price.
- If the tool does not show up right away in the table you can click the refresh button.
- You can't have the same `Line on PO` more than once on each PO but you can have multiple lines with the same `Tool ID Number`.
- Clicking the red cross next to the tool will delete that tool from this PO.

## Viewing the general overview page



- After adding all the tools from the PO make sure all the information is correct.
- Click `Print general information sheet` for a printable version of this information.
- Press `CTRL-p` or `⌘-p` to print this page.
- Make sure to choose `Landscape` as a layout.

## Adding a scan of the PO



- Use the scanner and the computer in the lab to save a scanned picture of the PO.
- Ask your supervisor about what to name the scan and where to save it.
- Make sure you have the right image before uploading it
- **Try to keep the file-size as small as possible.**
- Click the image to open a bigger version of it in another tab.

## Adding a run



- Make sure you have the correct PO chosen.
- After filling out the form click the plus to store the run.
- If the run does not show up right away in the table you can click the refresh button.
- The Run ID is auto generated. The format is `Machine+Date+Run` for machine.

## A quick way to add runs and edit them.



- The recently added runs list shows the 6 newest runs in the database.
- Select the run and click `Add run`. Now this run is linked to this PO.
- To edit information about a run click the right `Run ID`.
- The fields in the edit pop-up display the information as it is so only edit fields that are wrong.

## Assign tools to run



- You can add the same line item to multiple runs
- You can edit the information displayed in the table by clicking the right `Line item #`.
- Clicking the red cross next to the table will delete that entry from the table.
- If the line item does not show up right away in the table you can click the refresh button.

## Adding tools error



- You can not add the same line item twice to the same run.
- If you want to change the number of tools in a run, delete the entry and add it again with correct information.
- If you add more tools to runs than you received on your PO you will get a warning. This can either be re-runs or an input error.

## Shipping the PO



- After all tools have been coated you can ship the PO back.
- Make sure all information is correct. You can change the tools in shipment if it is wrong for some reason.
- Add a comment and a shipping date and click `Save`
- **The customer we are sending this shipment too will see this comment.**