**Team13 Library Database**

Hosted Site:

[**http://onethreelibrary.site/**](http://onethreelibrary.site/)

Link to files and readme:

[**https://drive.google.com/drive/folders/1nnmtJf7R45BZ9YpfuE35UvN-HWYW2V3D?usp=sharing**](https://drive.google.com/drive/folders/1nnmtJf7R45BZ9YpfuE35UvN-HWYW2V3D?usp=sharing)

*Description:*

**Library database**: The purpose of this project was to create a library database with several different functionalities. The library contains two types of users members and admins (employees). The members are split into students and faculty. A member may request/hold items. There is a different limit for how many items a faculty vs student can borrow or hold. Also, the number of days will be different in case of students and teachers. Each item has a unique ID, and may have a different number of copies.

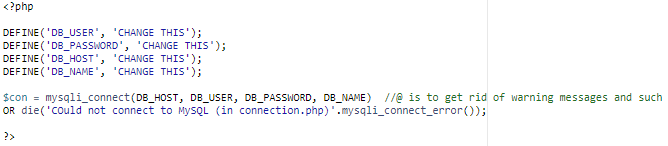
*Installing the files & Setting Up:*

The Library project.zip folder in the link above contains the database sql dump (onethree\_test.sql) and all of the files for the front end.

**Using WAMP (W**indows **A**pache **M**ySQL **P**HP**)**

After the installation of WAMP proceed with the following steps.

1. Import the database dump file (onethree\_test.sql) into MySQL Workbench.
2. Drag all php files from the Files Folder into the wamp project folder.
3. Edit the connection.php file and update root and password to local settings.



1. To keep track of days passed with triggers **cronjobs** were used in this project, they were installed using a feature of our stack, and for local purposes they are installed using linux commands. Since we did not use these commands, an alternative way to get the same result, regarding the days passed, can be done using an sql event instead. The code for the event is below

-------------------------------------------------------------------------------------------------------------------------------

CREATE DEFINER=`root`@`localhost` EVENT `DaysPassed` ON SCHEDULE EVERY 1 DAY STARTS '2021-04-09 21:46:31' ON COMPLETION NOT PRESERVE ENABLE DO UPDATE daysopened SET days\_passed= days\_passed+ 1

-------------------------------------------------------------------------------------------------------------------------------

The above code is to update the days passed once a day. If you would like to see this happen at a much faster rate, then the code below will make a second equal to a day passed in the system

-------------------------------------------------------------------------------------------------------------------------------

CREATE DEFINER=`root`@`localhost` EVENT `DaysPassed` ON SCHEDULE EVERY 1 SECOND STARTS '2021-04-09 21:46:31' ON COMPLETION NOT PRESERVE ENABLE DO UPDATE daysopened SET days\_passed= days\_passed+ 1

-------------------------------------------------------------------------------------------------------------------------------

To view events in my sql workbench use

SHOW EVENTS FROM test;

Test can be replaced with the database name. To show the code the event has use

SHOW CREATE EVENT DaysPassed;

To enable events or turn them off use

SET GLOBAL event\_scheduler = ON;

SET GLOBAL event\_scheduler = OFF;

To drop the event if you no longer want to use it use

DROP EVENT IF EXISTS DaysPassed;

\*Do note that the fines in this system run off an integer count to calculate fines. For reports, they are based on timestamps and not this count. Additionally if you wish to see this count manually, go to the page with projectfoler/cronjob.php (replace projectfolder to local project name) to increase the days by one per refresh. Another note is that faculty members get 20 days before fines are accumulated while students get 14 days.

Unfortunately for the cronjob regarding emails, this is done using our stack and we can not figure out a way to have this working automatically on local host. To run this manually follow the instructions below.

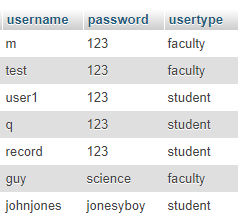
In order to run the email script, edits need to be done to php.ini and sendmail.ini

1. Go to the php.ini file and search for [mail function]. Comment out SMTP=localhost and smtp\_port=25 using ‘;’
2. Set “sendemail\_from =” whatever email you prefer using or any dummy emails
3. Set “sendmail\_path = C:\xampp\sendmail\sendmail.exe -t” or whatever path your sendmail.exe file lies in.

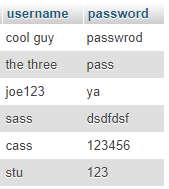
Now go to the sendmail.ini file

1. set “smtp\_server=smtp.gmail.com” for gmail or “smtp.mail.yahoo.com” if you are using yahoo
2. set “smtp\_port=465” and “smtp\_ssl = ssl”
3. auth\_username= user email that you created for this test
4. auth\_password = your email’s password
5. If apache server is still running, restart it so changes go into effect.
6. Create a cronjob to run the script everyday. The purpose is to email the user, who still has items borrowed, with a list of the items that are due in 3 days. This can be done on windows using a task scheduler or in linux using $crontab -e.

Member side login information

****

Admin login information

****

*Triggers:*

There are 6 triggers in this project and 4 of them are used to help calculate fines. Within the schema there are 2 triggers attached to DaysPassed. DaysPassed is a table dedicated to just counting days. The first trigger is an after update trigger used to update the day passed count in fines.

-------------------------------------------------------------------------------------------------------------------------------

CREATE TRIGGER `UpdateDays` AFTER UPDATE ON `DaysOpened`

FOR EACH ROW UPDATE fines SET days\_passed = days\_passed + 1 WHERE paid = 0

-------------------------------------------------------------------------------------------------------------------------------

The second is also an after update trigger to update the fine amount in fines if its days passed is greater than 14 days

-------------------------------------------------------------------------------------------------------------------------------

CREATE TRIGGER `UpdateFine` AFTER UPDATE ON `DaysOpened`

FOR EACH ROW UPDATE fines SET Fine\_Amount = Fine\_Amount + 0.5 where paid = 0 AND Days\_Passed >= 14

-------------------------------------------------------------------------------------------------------------------------------

NOTE: The starting fine amount for faculty is -3 to give them an additional 6 days. The fines don't show up on the front end until this amount is past 0.

Then in the fines table there are 2 triggers that give permission and revoke permission to rent. The first trigger removes permission

-------------------------------------------------------------------------------------------------------------------------------

CREATE TRIGGER `cancheckout` BEFORE UPDATE ON `fines`

FOR EACH ROW BEGIN

UPDATE members SET members.isallowedtorent = 0 WHERE members.cardnumber = NEW.cardnumber AND NEW.paid = 0 AND NEW.fine\_amount>0;

END

-------------------------------------------------------------------------------------------------------------------------------

And the second trigger does the opposite of this and gives back permission to rent for the member.

-------------------------------------------------------------------------------------------------------------------------------

CREATE TRIGGER `givepermission` BEFORE UPDATE ON `fines`

FOR EACH ROW BEGIN

UPDATE members SET members.isallowedtorent = 1 WHERE members.cardnumber = NEW.cardnumber AND NEW.paid = 1;

END

-------------------------------------------------------------------------------------------------------------------------------

The last 2 triggers are on the admin side and are activated when creating an employee or updating. These will stop any insert or update of an employee that is under the age of 18. The code for the two is below

-------------------------------------------------------------------------------------------------------------------------------

CREATE TRIGGER `check\_age` BEFORE INSERT ON `employees`

FOR EACH ROW BEGIN

IF (unix\_timestamp() - unix\_timestamp(new.bdate) < 60\* 60 \*24 \* 365\*18) THEN

SIGNAL SQLSTATE '02000' SET MESSAGE\_TEXT = 'Warning: Cannot hire an employee younger than 18 years old!';

END IF;

END

------------------------------------------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------------------------------------------

BEGIN

IF (unix\_timestamp() - unix\_timestamp(new.bdate) < 60\* 60 \*24 \* 365\*18) THEN

SIGNAL SQLSTATE '02000' SET MESSAGE\_TEXT = 'Warning: Employee cannot be youger than 18 years old!';

END IF;

END

-------------------------------------------------------------------------------------------------------------------------------

*Features of the project:*

Initial page consists of a login page for the member and an employee (admin.php).

Successful login into the member page directs user to the home page where they are welcomed to the library and a few blue links are shown below that can direct the user to a certain page.

Clicking on the browse library link directs user to the library database where they are presented with a table that views all the items in the library. Items have an ID, a name, a type, a status, quantity, and a more information link that will direct them to more info about the item. They are given a few options(or blue links) in which the user can: check out the item if the item is available, hold the item if the item is unavailable, or continue browsing(which just sends the user back to the library database page. Checking out an item becomes successful if the user doesn’t have any fines on his/her/their account. The return items link directs the user to a page consisting of items they can return and items that the user has On Hold.

Clicking on the Account report in the home page to view a report on the user’s checkout history as well as the fine history. User can choose a time frame from the time dropdown menu and generate the report; option are 1 week, 2 weeks, 3 weeks, or all time. Checkout history views the items borrowed with their corresponding date issued, due date, and returned date; if not returned then it shows “Still out”.

Furthermore, the user can also click on the check fines link in the home page to view any fines that exists with the option to pay for it. Paying off any debt with the library gives the user liberty of checking out items once again since fines obligate users from checking out.

Finally the logout tab redirects user back to the login page where another member/ employee can login to the library.

Successful login into the admin page directs user into the home page where they are able to add a new item(book, media, device, or a journal). The admin has a few tabs on top of the website to choose from(Inventory, employees, members, New items report, overdue items report, and logout).

Inventory page shows tables of each category of item with the choice to edit or delete them. Clicking on the employees tab directs user ot the employee tab where it initially shows a search bar to identify a specific employee using name or ID. Just clicking search will load all the employees onto the page with options to edit or delete each entry or add a new entry.

New items report shows all the new items added to the library in the last 30 days. You can specify item type and genre of books.

Overdue items report shows all the items overdue in the library at the moment generating the report. Results can be ordered by due date and specified how long it’s been due(more than two weeks). Total amount and late days are calculated.

*Credits:*

**Collaborators:**

Yury Ionov

Daniel Bonilla

Lingwei Kong

Ahmed Diefalla

Andres Sanchez

Made with the help of these tutorials:

HTML filter data:

<https://www.youtube.com/watch?v=4ZBTeqSuBrk>

Crud Tutorials:

<https://www.youtube.com/watch?v=3xRMUDC74Cw&t=61s>

<https://www.youtube.com/watch?v=JZdMXUIMdQw>

WAMP tutorial:

<https://www.youtube.com/watch?v=Afi7vdPsK6A>

Login Tutorial:

<https://www.youtube.com/watch?v=WYufSGgaCZ8>