* Database:
  + This part’s database is named bookmarks and contain two tables to help store user authentication information and their links.
  + The *users* table is formed of two columns, a *user\_name*, wich is also the primary key, and a *user\_pswrd,* allowing the user to verify their identity.
  + Given that the username is a primary key, it is included in the SignIn script to request for usernames to be unique.
  + Both columns are NOT NULL, this they both must the filled in order to create a new user.
  + The links table is composed of 5 diferent columns of which only two will be of the user’s concern, the other three are for backend processing.
  + *link\_id* is a numeric INT value, automatically incremented and set by the database. This number is primary key and allow us to identify every individual link, since diferent users could store the same URL and even under the same name.
  + *link\_adr* is the url address of the website to bookmark.
  + *link\_name* refers to the name the link is given by the user, this for easier browsing on the user’s side.
  + *link\_click* helps count how many times a link was clicked, latter on ussed to explain popularity
  + *user\_name* is a foreign key to the user table, ensuring every links instance is assigned to an user.
* SingIn & LogIn/Out:
  + For the login, I ussed as mix of Ajax, so that the user could send in new links and navigate without reloading, and php, to fetch the credentials.
  + The user credentials have been hashed with a simple hash, as to protect the user’s identity ([PHP: password\_hash - Manual](https://www.php.net/manual/en/function.password-hash.php)).
  + The user credentials serve to open a session, that will remain open as long as the user is on the website and will be closed once they leave, as a safety measure.
  + When the user Signs in, their password is automatically hashed (password\_hash($plaintextPassword, PASSWORD\_BCRYPT); ) and can the only be deciphered via password\_verify($password, $row['user\_pswrd']) during the log in.
* Adding a bookmark:
  + Simple SQL query passed through php allows to create a new bookmark
  + The bookmark entity is composed of a numeric primary key for storage purposes, a name, a username (foreign key to the user that created the link) and a click counter.
  + Both the link counter and the id are automatically generated, the username is fetched from the session in course.
  + The click counter, link\_click, serves to account the number of times a link was clicked by the user.
  + It is via the link counter that we use a SQL query to search the links with the most gross visits across all users.
* Browsing bookmarks:
  + To browse the bookmarks the user is expected to lookup the name they gave to the bookmark.
  + This, because it is more likely for the user to remember the name, they use than the link path.
  + The SQL query only looks through the links belonging to that specific user.
  + If no search is being executed, then all links are displayed. This, thanks to AJAX processing, to avoid constant re-loading. ([PHP - AJAX and PHP](https://www.w3schools.com/php/php_ajax_php.asp))
* Editing/Deleting:
  + As requested, the user can edit or delete a bookmark of their choice.
  + Both buttons related to these processes appear besides each bookmark, linking the action to the data to be altered.
  + The delete button simply deletes the link entity from the database, thus making it disappear form the users list.
  + A window prompt does appear before deletion, asking the user if they are certain of their decision. Upon approval, the bookmark is deleted.
  + For editing, we call upon a modal, allowing the user to edit either the link or the name assigned to the bookmark.
  + At any point the user may click on the x to close the window, and no changes will be made.
  + If the user wants to precede with the changes, they may press the save changes button, which will then call the php script (SQL query) to update the link instance in the link table of the database.
  + Let it be noted that all this process uses the link\_id as guiding thread for all the SQL queries, as they are unique to each link\_id and automatically generated by the system. ([MySQL PRIMARY KEY Constraint](https://www.w3schools.com/mysql/mysql_primarykey.asp))
* The page is divided in two main sections, the form to stored bookmarks and the browser.
* An additional home page (hero section) is added to greet the user and display the top 10 links that, as previously stated, are calculated based on total click across all the entities containing the same link.
* A navigation bar helps jump from one section to another and additionally allows to log out.
* The website cannot be accessed without first signing in.
* Please ensure proper database acces in in connection.php
* Database:
  + *Lessons* is the database for the homework’s second part. It is composed of three tables, serving to authenticate, store content and keep track of made progress.
  + The user table is identical to that of part 1, so you may refer to the database section of the documentation, under users.
  + The *xml\_storage* table, is composed of 3 columns and serves to store the quiz and lesson EMLs.
  + *id*  column assigns every xml file an autogenerate unique id that serves as primary key and will be used all through part 2 to identify the lesson and quiz EMLs.
  + *filename* is the name of the file uploaded, there is not much need for precision but in real world circumstances it is important to properly identify documents with significative names.
  + *xml\_content* contains the EMLs themselves, stored as LONGTEXT in the database to be called later on by the application.
  + The progress\_record table is made up of 4 diferent columns, are used for backend logic support.
  + *record\_id* table
* Authentication:
  + This part employs the same authentication system as part 1. The user table in the database is private to the lessons database, but the logic is the same.
  + Please refer to the SingIn & LogIn/Out section of part 1.
* Additional notes:
  + Same hashing rules from part 1 apply.
  + The EMLs can be uploaded to the database via xmlUpload.php, one at a time, specifying proper file URL and name.
  + Please ensure the EML’s ids are numbered 1-6, from 1st to 3rd as units and 4th to 6th as quizzes.
  + Please ensure proper database acces in in connect.php