Sample application: SimplePDBBrowser

**Overview of the application**

SimplePDBBrowser will be a relatively simple gateway to access Protein Data Bank data. It will allow to search by codes, and for textual data included in PDB headers like, description, techniques used, resolution, source or author. Results of the search will give basic information about the PDB entry and link to the appropriate web pages in the main PDB web site.

**Input data.** *Nature and origin (user input, replicated database, calculation results, etc.). Input validation if necessary. User control, etc.*

User input: User should fill a web form including either a PDB id or some information. Fields available would be:

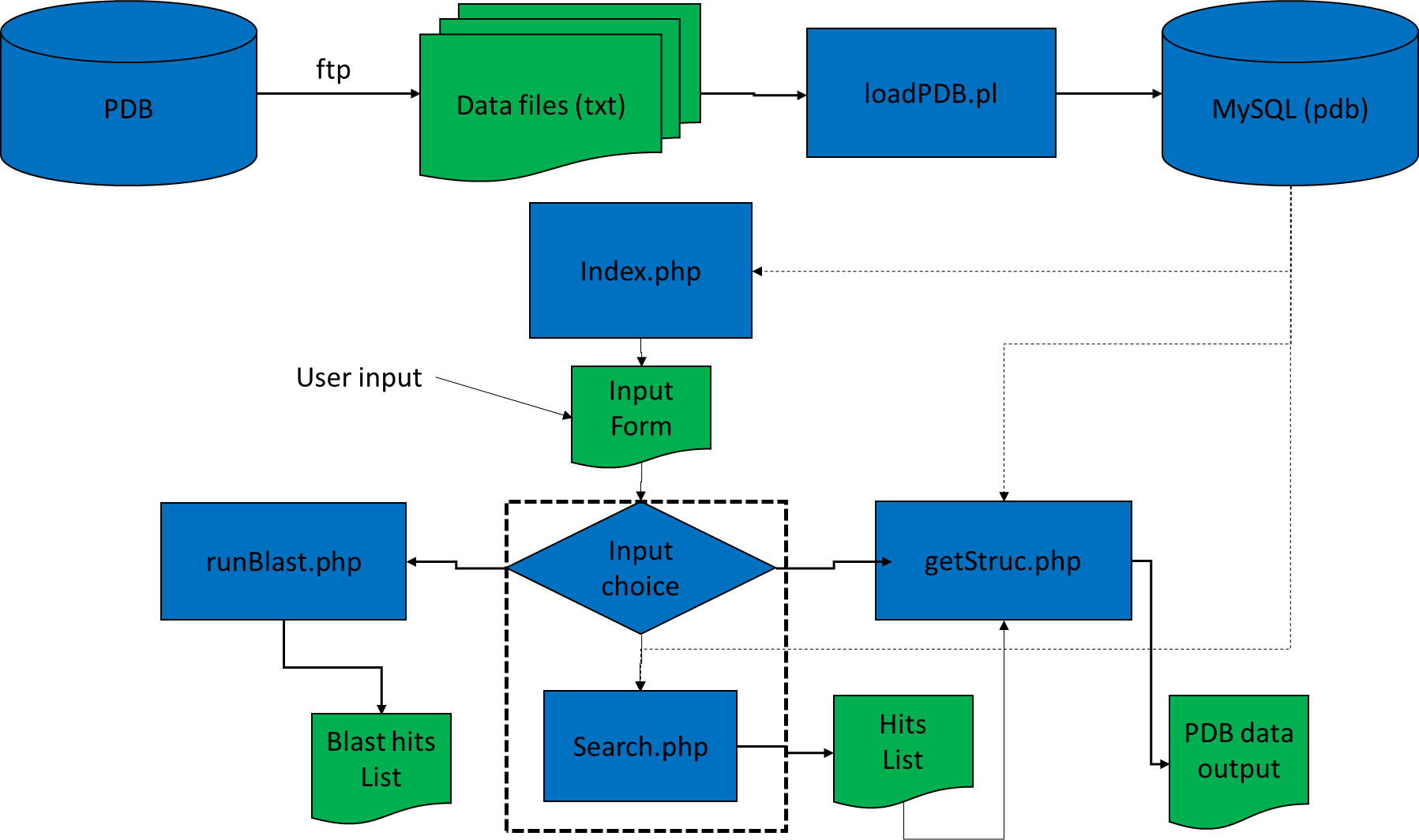
* PDB code (4 letter code). Will direct to a single entry if available
* Search Fields
  + Description, headers, authors, source (text search)
  + Resolution (numeric input)
  + Technique used (selection of possibles, X Ray, NMR, etc)
  + Type of molecule (selection of protein, nucleic acid, complex)
  + Year of submission (numeric input)
* Sequence search: PDB entries will be obtained through a Standard Blast search. Sequence can be provided as text or uploaded file

Data input: Data will be obtained off-line by parsing the index files provided at Protein Data Bank Ftp (/derived\_data), and stored in the a database with the appropriate structure.

**Output.** *Data, way of presentation (web/email).*

Search/blast search outputs: Output of a search would be a list of the found hits, showing a limited amount of information just to identify the hit: PDB Id, entry description, type of molecule, source. List items will have links to entries data.

Entry data: Complete information available about the entry, links to the PDB main site, and to other related sites with structural information.

****

**globals.inc.php**

(to be included by any .php script)

<?php

/\*

\* globals.inc.php

\* Global variables and settings

\*/

// Base directories

// Automatic, taken from CGI variables.

$baseDir = dirname($\_SERVER['SCRIPT\_FILENAME']);

#$baseDir = '/home/dbw00/public\_html/PDBBrowser';

$baseURL = dirname($\_SERVER['SCRIPT\_NAME']);

// Temporal dir, create if not exists, however Web server

// may not have the appropriate permission to do so

$tmpDir = "$baseDir/tmp";

if (!file\_exists($tmpDir)) {

mkdir($tmpDir);

}

// Blast details, change to adapt to local settings

// Blast databases should be created using the appropriate programs.

$blastHome = "$baseDir/../../blast";

$blastDbsDir = "$blastHome/DBS";

$blastExe = "$blastHome/bin/blastp";

$blastDbs = array("SwissProt" => "sprot", "PDB" => "pdb");

$blastCmdLine = "$blastExe -db $blastDbsDir/" . $blastDbs['PDB'] . " -evalue 0.001 -max\_target\_seqs 100 -outfmt \"6 sseqid stitle evalue\"";

// Include directory

$incDir = "$baseDir/include";

// Load accessory routines

include\_once "$incDir/bdconn.inc.php";

include\_once "$incDir/libDBW.inc.php";

// Load predefined arrays

// Fulltext search fields

$textFields = Array('e.header', 'e.compound', 'a.author', 's.source', 'sq.header');

// Compounds

$rs = mysqli\_query($mysqli, "SELECT \* from comptype") or print mysql\_error();

while ($rsF = mysqli\_fetch\_array($rs)) {

$compTypeArray[$rsF['idCompType']] = $rsF['type'];

}

//expTypes

$rs = mysqli\_query($mysqli,"SELECT \* from expType") or print mysql\_error();

while ($rsF = mysqli\_fetch\_array($rs)) {

$expTypeArray[$rsF['idExpType']] = $rsF;

}

//expClasses

$rs = mysqli\_query($mysqli,"SELECT \* from expClasse") or print mysql\_error();

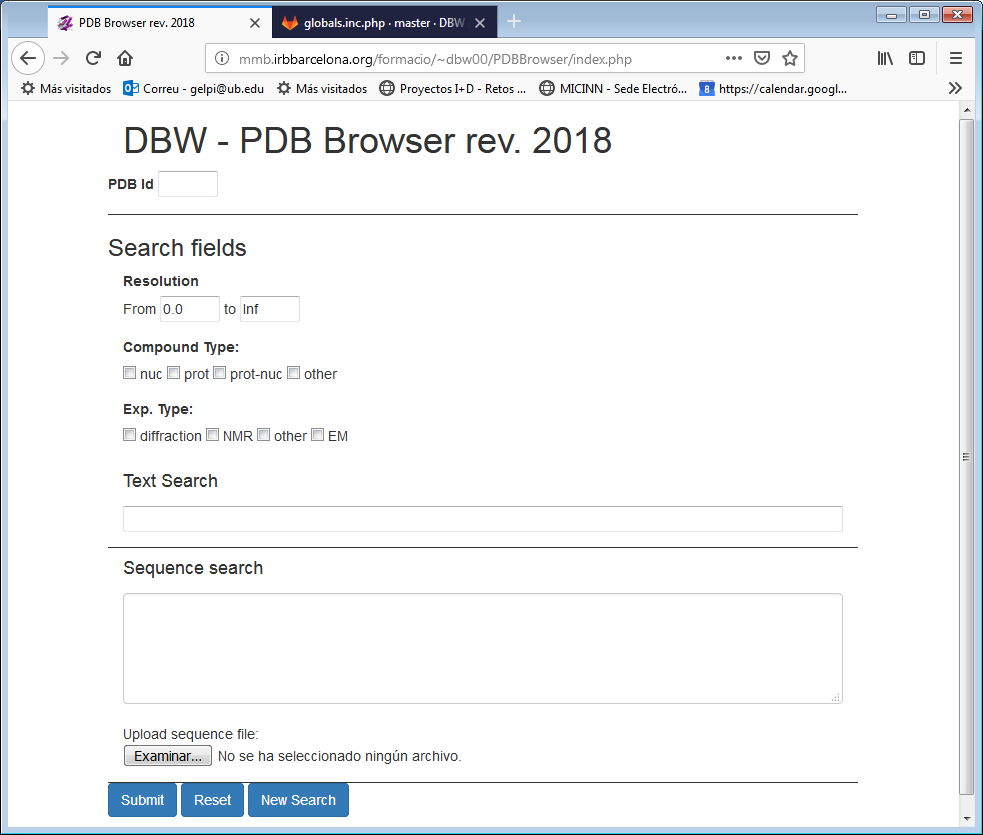
while ($rsF = mysqli\_fetch\_array($rs)) {

$expClasseArray[$rsF['idExpClasse']] = $rsF['expClasse'];

}

// Start session to store queries

session\_start();



**index.php**

Main entry for the application. Uses default values from $\_SESSION to reload pass choices

<?php

/\*

\* index.php, main form

\*/

# Loading global variables and DB connection

include "globals.inc.php";

//

// $\_SESSION['queryData'] array holds data from previous forms,

// if empty it should be initialized to avoid warnings, and set defaults

// also a ...?new=1 allows to clean it from the URL.

//

if (isset($\_REQUEST['new']) or !isset($\_SESSION['queryData'])) {

$\_SESSION['queryData'] = [

'minRes' => '0.0',

'maxRes' => 'Inf',

'query' => ''

];

}

print headerDBW("PDB Browser rev. 2018");

#Main Form follows

?>

<form name="MainForm" action="search.php" method="POST" enctype="multipart/form-data">

<div class="row" style="border-bottom: solid 1px">

<div class="form-group">

<label><b>PDB Id</b></label>

<input type="text" name="idCode" value="<?php print $\_SESSION['queryData']['idCode']?>" size="5" maxlength="4"/>

</div>

</div>

<div class="row">

<h3>Search fields</h3>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

<label>Resolution</label>

<p> From <input type="text" name="minRes" value="<?php print $\_SESSION['queryData']['minRes'] ?>" size="5">

to <input type="text" name="maxRes" value="<?php print $\_SESSION['queryData']['maxRes'] ?>" size="5" >

</p>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

<label>Compound Type:</label>

<div class="form-check">

<?php

/\*

\* input options from $compTypeArray[] array

\*/

foreach (array\_keys($compTypeArray) as $idCompType ) {?>

<input class="form-check-input" type="checkbox"

name="idCompType[<?php print $idCompType ?>]" /> <?php print $compTypeArray[$idCompType]."\n" ?>

<?php } ?>

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

<label>Exp. Type:</label>

<div class="form-check">

<?php

/\*

\* input options from $expClasseArray

\*/

foreach (array\_keys($expClasseArray) as $idExpClasse) { ?>

<input class="form-check-input" type="checkbox" name="idExpClasse[<?php print $rsF['idExpClasse'] ?>]" /> <?php print $expClasseArray[$idExpClasse] . "\n" ?>

<?php } ?>

</div>

</div>

</div>

</div>

<div class="row" style="border-bottom: 1px solid">

<div class="col-md-6">

<label><h4> Text Search</h4></label>

<div class="form-group">

<input type="text" name="query" value="<?php print $\_SESSION['queryData']['query'] ?>" style="width:100%" />

</div>

</div>

</div>

<div class="row" style="border-bottom: 1px solid">

<div class="col-md-6">

<label><h4>Sequence search</h4></label>

<div class="form-group">

<textarea class="form-control" name="seqQuery" rows="4" cols="60" style="width:100%"></textarea><br>

Upload sequence file: <input type="file" name="seqFile" value="" width="50" style="width:100%"/>

</div>

</div>

</div>

<div class="row">

<p>

<button type='submit' class="btn btn-primary">Submit</button>

<button type='reset' class="btn btn-primary">Reset</button>

<button class="btn btn-primary" onclick="window.location.href='index.php=?new=1">New Search</button>

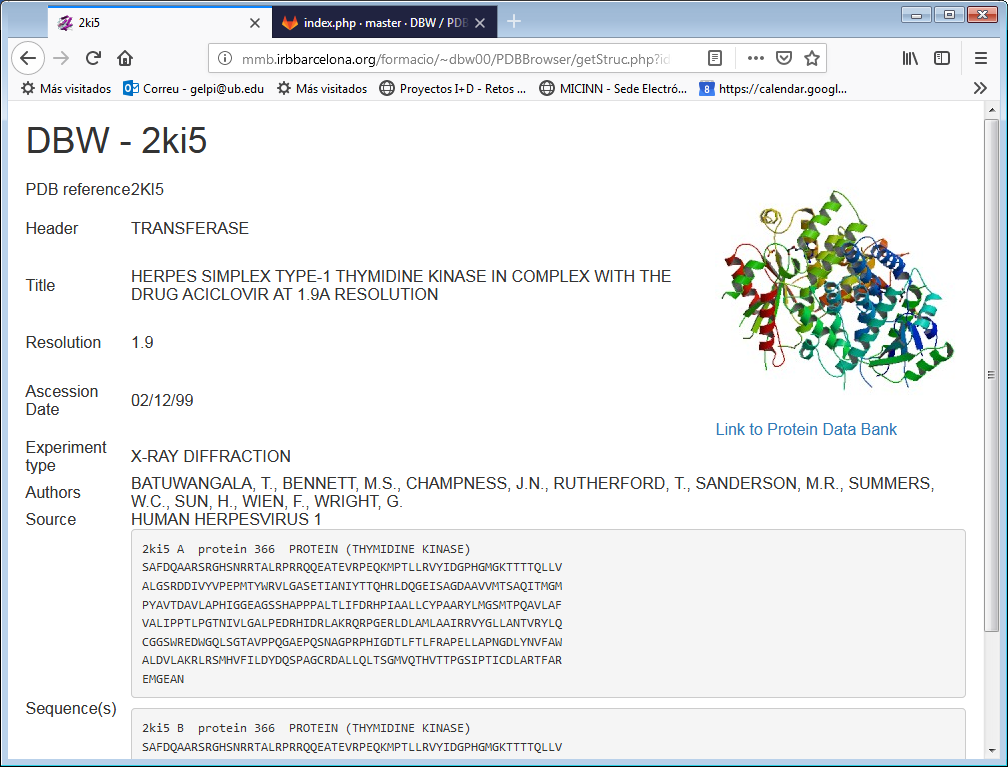
</p>

</div>

</form>

<?php

print footerDBW();



**getStruc.php**

Output view for PDB data (parameters passed using GET)

<?php

/\*

\* getStruc.php Shows data for a PDB entry

\*/

// load global vars and includes

include "globals.inc.php";

//get data for the structure requested

$sql = "SELECT e.\* from entry e where e.idCode='" . $\_REQUEST['idCode'] . "'";

$rs = mysqli\_query($mysqli, $sql) or print mysqli\_error($mysqli);

if (!mysqli\_num\_rows($rs)) { //search is empty

print errorPage('Not Found', 'The requested structure is not available');

} else {

$data = mysqli\_fetch\_assoc($rs);

print headerDBW($\_REQUEST['idCode']);

?>

<table border="0" cellspacing="2" cellpadding="4">

<tbody>

<tr>

<td>PDB reference</td>

<td><?php print $data['idCode'] ?></td>

<td rowspan="5">

<a href="http://www.pdb.org/pdb/explore.do?structureId=<?php print $data['idCode'] ?>">

<img src="http://www.pdb.org/pdb/images/<?php print strtolower($data['idCode']) ?>\_bio\_r\_250.jpg" border="0" width="250" height="250"><br>

Link to Protein Data Bank</a>

</td>

</tr>

<tr>

<td>Header</td>

<td><?php print $data['header'] ?></td>

</tr>

<tr>

<td>Title</td>

<td><?php print $data['compound'] ?></td>

</tr>

<tr>

<td>Resolution</td>

<td>

<?php

if ($data['resolution']) {

print $data['resolution'];

} else {

print "N.D.";

}

?>

</td>

</tr>

<tr>

<td>Ascession Date</td>

<td><?php print $data['ascessionDate'] ?></td>

</tr>

<tr>

<?php // $expTypeArray is generated in globals.inc.php?>

<td>Experiment type</td>

<td colspan="2"><?php print $expTypeArray[$data['idExpType']]['ExpType'] ?></td>

</tr>

<tr>

<td>Authors</td>

<td colspan="2">

<?php

// new DB query to get authors

$rsA = mysqli\_query($mysqli, "SELECT \* from author a, author\_has\_entry ae where ae.idCode='" . $data['idCode'] . "' and a.idAuthor = ae.idAuthor order by a.author") or print mysqli\_error($mysqli);

if (mysqli\_num\_rows($rsA)) {

$auts = array();

while ($rsAF = mysqli\_fetch\_assoc($rsA))

$auts[] = $rsAF['author'];

print join(", ", $auts);

}

?>

</td>

</tr>

<tr>

<td>Source</td>

<td colspan="2">

<?php

// new DB query to get sources

$rsA = mysqli\_query($mysqli, "SELECT \* from source s, entry\_has\_source es where es.idCode='" . $data['idCode'] . "' and s.idSource = es.idSource order by s.source") or print mysql\_error($mysqli);

if (mysqli\_num\_rows($rsA)) {

$sources = array();

while ($rsAF = mysqli\_fetch\_assoc($rsA))

$sources[] = $rsAF['source'];

print join(", ", $sources);

}

?>

</td>

</tr>

<tr>

<td>Sequence(s)</td>

<td colspan="2">

<?php

// new DB query to get sequences, output in FASTA format

$rsA = mysqli\_query($mysqli, "SELECT \* from sequence s where s.idCode='" . $data['idCode'] . "' order by s.chain")

or print mysqli\_error($mysqli);

if (mysqli\_num\_rows($rsA)) {

while ($sq = mysqli\_fetch\_assoc($rsA)) {

print "<pre>" . $sq['header'] . "\n" . preg\_replace("/(.{60})/", "$1\n", $sq['sequence']) . "</pre>";

}

}

?>

</td>

</tr>

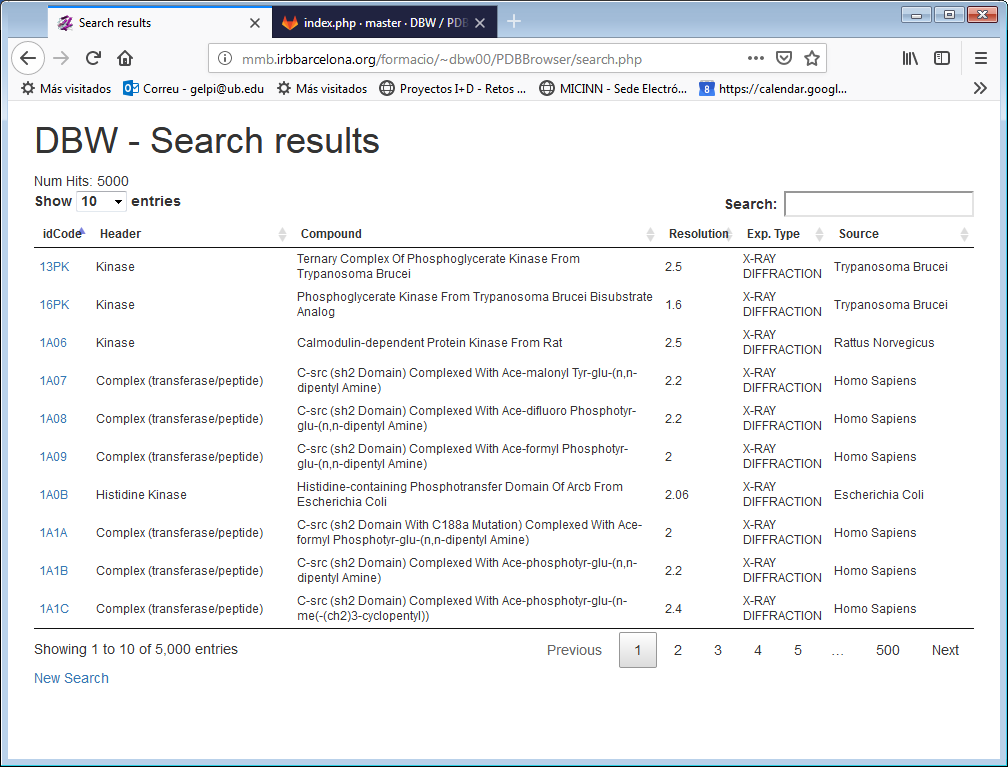
</tbody>

</table>

<?php

print footerDBW();

}



**search.php**

Search script, includes initial selection of operations

<?php

// load global vars and includes

include "globals.inc.php";

// Store input data in $\_SESSION to reload initial form if necessary

$\_SESSION['queryData'] = $\_REQUEST;

// Selection of action to do

// 1. IdCode -> Results page

// 2. Sequence input -> runBlast

// 3. Other -> search on DB

//

// 1. Redirection to the requested entry if code selected

if ($\_REQUEST['idCode']) {

header('Location: getStruc.php?idCode=' . $\_REQUEST['idCode']);

// 2. Sequence input. If uploaded file, this takes preference

} elseif ($\_FILES['seqFile']['name'] or $\_REQUEST['seqQuery']) {

if (($\_FILES['seqFile']['tmp\_name'])) {

$\_SESSION['queryData']['seqQuery'] = file\_get\_contents($\_FILES['seqFile']['tmp\_name']);

}

// Redirect to Blast if sequence, data is already stored in $\_SESSION

header('Location: runBlast.php');

} else {

// 3. normal search, Bluiding SQL SELECT from the input form

// $ANDConds will contain all SQL conditions found in the form

$ANDconds = ["True"]; // required to fulfill SQL syntax if form is empty

// Resolution, we consider only cases where user has input something

if (($\_REQUEST['minRes'] != '0.0') or ( $\_REQUEST['maxRes'] != 'Inf')) {

if ($\_REQUEST['minRes'] != '0.0') {

$ANDconds[] = "e.resolution >= " . $\_REQUEST['minRes'];

}

if ($\_REQUEST['maxRes'] != 'Inf') {

$ANDconds[] = "e.resolution <= " . $\_REQUEST['maxRes'];

}

}

// Compound type $ORconds holds options selected

if (isset($\_REQUEST['idCompType'])) { //should be isset as idCompType come from checkboxes

$ORconds = [];

foreach (array\_keys($\_REQUEST['idCompType']) as $k) {

$ORconds[] = " e.idCompType = " . $k;

}

$ANDconds[] = "(" . join(" OR ", $ORconds) . ")";

}

// Classe of experiment

if (isset($\_REQUEST['idExpClasse'])) {//should be isset as idExpClasse come from checkboxes

$ORconds = [];

foreach (array\_keys($\_REQUEST['idExpClasse']) as $k) {

$ORconds[] = " et.idExpClasse = " . $k;

}

$ANDconds[] = "(" . join(" OR ", $ORconds) . ")";

}

// text query, adapted to use fulltext indexes, $textFields is defined in globals.inc.php and

// lists all text fields to be searched in.

if ($\_REQUEST['query']) {

$ORconds = [];

foreach (array\_values($textFields) as $field) {

$ORconds[] = "MATCH (" . $field . ") AGAINST ('" . $\_REQUEST['query'] . "' "

. "IN BOOLEAN MODE)";

}

$ANDconds[] = "(" . join(" OR ", $ORconds) . ")";

}

// text query without fulltext indexes

// if ($\_REQUEST['query']){

// foreach (split (' ',$\_REQUEST['query']) as $wd) {

// $ANDconds=array();

// foreach (array\_values($textFields) as $field) {

// $ORconds[] = $field." like '%".$wd."%'";

// }

// $ANDconds[] = "(".join (" OR ", $ORconds).")";

// }

// }

// main SQL string, make sure that all tables are joint, and relationships included

// SELECT columns FROM tables WHERE Conditions\_from\_relationships AND Conditions\_from\_query\_Form

$sql = "SELECT distinct e.idCode,e.header,e.compound,e.resolution,s.source,et.expType FROM

expType et, author\_has\_entry ae, author a, source s, entry\_has\_source es, entry e, sequence sq WHERE

e.idExpType=et.idExpType AND

ae.idCode=e.idCode and ae.idAuthor=a.idAuthor AND

es.idCode=e.idCode and es.idsource=s.idSource AND

e.idCode = sq.idCode AND

" . join(" AND ", $ANDconds);

// Ordering will be done by the DataTable element using JQuery, if not available can also be done from the SQL

// switch ($order) {

// case 'idCode':

// case 'header':

// case 'compound':

// case 'resolution':

// $sql .= " ORDER BY e." . $order;

// break;

// case 'source':

// $sql .= " ORDER BY s.source";

// break;

// case 'expType':

// $sql .= " ORDER BY et.expType";

// break;

// }

if (!isset($\_REQUEST['nolimit'])) {

$sql .= " LIMIT 5000"; // Just to avoid too long listings when testing

}

#DEBUG

print "<p>$sql</p>";

// DB query

$rs = mysqli\_query($mysqli,$sql) or print mysqli\_error($mysqli);

// We check whether there are results to show

if (!mysqli\_num\_rows($rs)) {

print errorPage("Not Found", "No results found.");

} else {

// Results table formated with DataTable

print headerDBW("Search results");

print "Num Hits: " . mysqli\_num\_rows($rs);

?>

<table border="0" cellspacing="2" cellpadding="4" id="dataTable">

<thead>

<tr>

<th>idCode</th>

<th>Header</th>

<th>Compound</th>

<th>Resolution</th>

<th>Exp. Type</th>

<th>Source</th>

</tr>

</thead>

<tbody>

<?php while ($rsF = mysqli\_fetch\_assoc($rs)) { ?>

<tr>

<td><a href="getStruc.php?idCode=<?php print $rsF['idCode'] ?>">

<?php print $rsF['idCode'] ?></a></td>

<td><?php print ucwords(strtolower($rsF['header'])) ?></td>

<td><?php print ucwords(strtolower($rsF['compound'])) ?></td>

<td><?php print $rsF['resolution'] ?></td>

<td><?php print $rsF['expType'] ?></td>

<td><?php print ucwords(strtolower($rsF['source'])) ?></td>

</tr>

<?php } ?>

</tbody>

</table>

<p class="button"><a href="index.php?new=1">New Search</a></p>

<script type="text/javascript">

<!-- this activates the DataTable element when page is loaded-->

$(document).ready(function () {

$('#dataTable').DataTable();

});

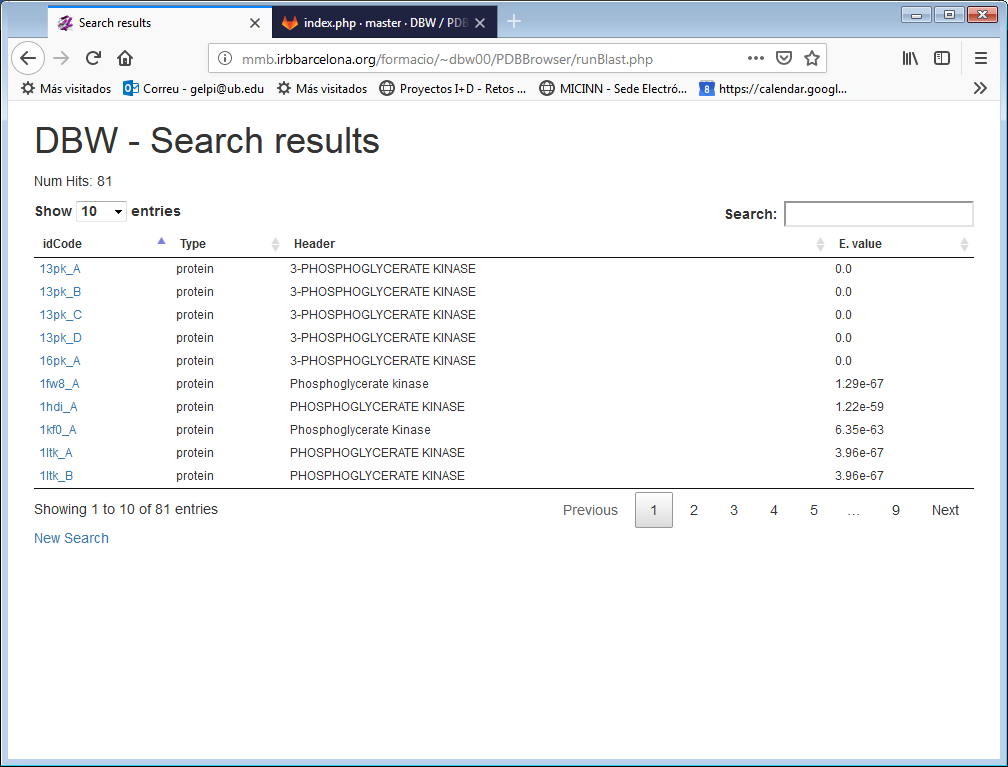
</script>

<?php

print footerDBW();

}

}



**runBlast.php**

Blast search

<?php

require "globals.inc.php";

// Take data from $\_SESSION, loaded in search.php, if empty back to front page

if (!isset($\_SESSION['queryData']) or ! $\_SESSION['queryData']['seqQuery'])

header('Location: index.php');

// prepare FASTA file

// Identify file format, ">" as first char indicates FASTA

$p = strpos($\_SESSION['queryData']['seqQuery'], '>');

if (!$p and ( $p !== 0)) { // strpos returns False if not found and "0" if first character in the string

// When is not already FASTA, add header + new line

$\_SESSION['queryData']['seqQuery'] = ">User provided sequence\n" . $\_SESSION['queryData']['seqQuery'];

}

// Set temporary file name to a unique value to protect from concurrent runs

$tempFile = $tmpDir . "/" . uniqId('pdb');

// Open temporary file and store query FASTA

$ff = fopen($tempFile . ".query.fasta", 'wt');

fwrite($ff, $\_SESSION['queryData']['seqQuery']);

fclose($ff);

// execute Blast, Command line set in globals.inc.php

print $blastCmdLine . " -query " . $tempFile . ".query.fasta -out " . $tempFile . ".blast.out\n";

exec($blastCmdLine . " -query " . $tempFile . ".query.fasta -out " . $tempFile . ".blast.out");

// Read results file and parse hits onto $result[]

$result = file($tempFile . ".blast.out");

if (!count($result)) {

print errorPage("Not Found", 'No results found. <p class="button" ><a href="index.php?new=1">New Search</a></p>');

} else {

// Results table

print headerDBW("Search results");

?>

<p>Num Hits: <?php print count($result) ?> </p>

<table border="0" cellspacing="2" cellpadding="4" id="blastTable">

<thead

<tr>

<th>idCode</th>

<th>Type</th>

<th>Header</th>

<th>E. value</th>

</tr>

</thead>

<tbody>

<?php

foreach (array\_values($result) as $rr) {

if (strlen($rr) > 1) {

$data = explode ("\t",$rr);

preg\_match('/(....)\_(.) mol:([^ ]\*) length:([0-9]\*) \*(.\*)/', $data[1], $hits);

list ($r, $idCode, $sub, $tip, $l, $desc)= $hits;

$ev = $data[2];

?>

<tr>

<td>

<a href="getStruc.php?idCode=<?php print $idCode ?>"><?php print $idCode . "\_$sub" ?></a>

</td>

<td><?php print $tip ?></td>

<td><?php print $desc ?></td>

<td><?php print $ev ?></td>

</tr>

<?php

}

}

?>

</tbody>

</table>

<p class="button"><a href="index.php?new=1">New Search</a></p>

<?php

// Cleaning temporary files

if (file\_exists($tempFile . ".query.fasta"))

unlink($tempFile . ".query.fasta");

// if (file\_exists($tempFile . ".blast.out"))

// unlink($tempFile . ".blast.out");

print footerDBW();

}

// DataTable activation

?>

<script type="text/javascript">

$(document).ready(function () {

$('#blastTable').DataTable();

});

</script>

**Include/bdconn.inc.php**

<?php

/\*

\* bdconn.inc.php

\* DB Connection

\*/

$host = "localhost";

$dbname = "pdb";

$user = "dbw00";

$password = "dbw2018";

($mysqli = mysqli\_connect($host, $user, $password)) or die(mysqli\_error());

mysqli\_select\_db($mysqli, $dbname) or die(mysql\_error($mysqli));

**Include/libDBW.inc.php**

<?php

function headerDBW($title) {

return "<html lang=\"en\">

<head>

<meta charset=\"utf-8\">

<meta http-equiv=\"X-UA-Compatible\" content=\"IE=edge\">

<meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">

<title>$title</title>

<!-- Bootstrap styles -->

<link rel=\"stylesheet\" href=\"https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css\" integrity=\"sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u\" crossorigin=\"anonymous\">

<!-- IE 8 Support-->

<!--[if lt IE 9]>

<script src=\"https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js\"></script>

<script src=\"https://oss.maxcdn.com/respond/1.4.2/respond.min.js\"></script>

<![endif]-->

<link rel=\"stylesheet\" href=\"DataTable/jquery.dataTables.min.css\"/>

<script type=\"text/javascript\" src=\"DataTable/jquery-2.2.0.min.js\"></script>

<script type=\"text/javascript\" src=\"DataTable/jquery.dataTables.min.js\"></script>

</head>

<body bgcolor=\"#ffffff\">

<div class= \"container\">

<h1>DBW - $title</h1>

";

}

function footerDBW() {

return '

</div>

</body>

</html>';

}

function errorPage($title, $text) {

return headerDBW($title) . $text . footerDBW();

}

**loadPDB.pl**

ETL to load database data from PDB text files

#!/usr/bin/perl

use DBI;

use strict;

#

require "bdconn.pl";

#

my $dbh=connectDB();

# Clean Tables !!

foreach my $tab ('entry', 'source', 'sequence', 'author', 'author\_has\_entry', 'expType', 'comptype', 'entry\_has\_source', 'expClasse') {

print "Cleaning $tab\n";

$dbh->do ("DELETE FROM $tab");

}

foreach my $tab ('source','author','expType','comptype','expClasse') {

$dbh->do ("ALTER TABLE $tab AUTO\_INCREMENT=1");

}

#

$dbh->do("SET FOREIGN\_KEY\_CHECKS=0");

#

print "Authors...";

my %AUTHORS;

my %author\_has\_entry;

my $sthAuthor=$dbh->prepare ("INSERT INTO author (author) VALUES (?)");

my $sthEntryAuthor=$dbh->prepare("INSERT INTO author\_has\_entry VALUES (?,?)");

open AUTS, "author.idx";

#author.idx

#IDCODE; AUTHOR

#2NPB ; AACHMANN, F.L.

#2KV1 ; AACHMANN, F.L.

while (<AUTS>) {

next if !/;/;

chomp;

my ($idCode, $author) = split / \*; \*/;

next if (!$author);

if ($author && !$AUTHORS{$author}) {

$sthAuthor->execute ($author);

$AUTHORS{$author}=$dbh->last\_insert\_id('','','Author','idAuthor');

}

if (!$author\_has\_entry{"$AUTHORS{$author}-$idCode"}) {

$sthEntryAuthor->execute ($AUTHORS{$author},$idCode);

$author\_has\_entry{"$AUTHORS{$author}-$idCode"}=1;

}

print ".";

}

close AUTS;

print "ok\n";

#

#

print "Sources...";

my %SOURCES;

my %Entry\_has\_source;

my $sthSource =$dbh->prepare("INSERT INTO source (source) VALUES (?)");

my $sthEntrySource = $dbh->prepare("INSERT INTO entry\_has\_source (idCode,idSource) VALUES (?,?)");

open SOUR, "source.idx";

#source.idx

#IDCODE SOURCE

#102L ENTEROBACTERIA PHAGE T4

#102M PHYSETER CATODON

while (<SOUR>) {

chomp;

my ($idCode,$source) = split ' ', $\_, 2;

next if (!$source) || (length($idCode) != 4);

foreach my $s (split /; \*/, $source) {

if (!$SOURCES{$s}) {

$sthSource->execute ($s);

$SOURCES{$s}=$dbh->last\_insert\_id('','','source','idSource');

}

$sthEntrySource->execute($idCode, $SOURCES{$s});

}

print ".";

}

close SOUR;

print "ok\n";

#

#

print "Entries...";

open ENTR, "entries.idx";

#entries.idx

# IDCODE, HEADER, ACCESSION DATE, COMPOUND, SOURCE, AUTHOR LIST, RESOLUTION, EXPERIMENT TYPE (IF NOT X-RAY)

#100D DNA/RNA 12/05/94 CRYSTAL STRUCTURE OF THE HIGHLY DISTORTED CHIMERIC DECAMER R(C)D(CGGCGCCG)R(G)-SPERMINE COMPLEX-SPERMINE BINDING TO PHOSPHATE ONLY AND MINOR GROOVE TERTIARY BASE-PAIRING Ban, C., Ramakrishnan, B., Sundaralingam, M. 1.9 X-RAY DIFFRACTION

#101D DNA 12/14/94 REFINEMENT OF NETROPSIN BOUND TO DNA: BIAS AND FEEDBACK IN ELECTRON DENSITY MAP INTERPRETATION Goodsell, D.S., Kopka, M.L., Dickerson, R.E. 2.25 X-RAY DIFFRACTION

my $sthEntry = $dbh->prepare ("INSERT INTO entry (idCode, header, ascessionDate, compound, resolution) VALUES (?,?,?,?,?)");

my %ExpTypes;

my $sthExpType = $dbh->prepare ("INSERT INTO expType (ExpType) VALUES (?)");

my $sthEntryExpType = $dbh->prepare ("UPDATE entry SET idExpType=? WHERE idCode=?");

my %expTypesbyCode;

while (<ENTR>) {

chomp;

my ($idCode, $header, $ascDate, $compound, $source, $authorList, $resol, $expType) = split /\t/;

next if (length($idCode) != 4);

$sthEntry-> execute ($idCode,$header, $ascDate, $compound, $resol);

if (!$ExpTypes{$expType }) {

$sthExpType->execute($expType);

$ExpTypes{$expType}=$dbh->last\_insert\_id('','','ExpType','idExpType');

}

$sthEntryExpType->execute($ExpTypes{$expType},$idCode);

$expTypesbyCode{$idCode}=$expType;

print ".";

}

close ENTR;

print "ok\n";

#

#

open EXPCL , 'pdb\_entry\_type.txt';

#pdb\_entry\_type.txt

#100d nuc diffraction

#101d nuc diffraction

my %expClasses;

my %compTypes;

my $sthExpClasse=$dbh->prepare('INSERT INTO expClasse (expClasse) VALUES (?)');

my $sthcompType=$dbh->prepare('INSERT INTO comptype (type) VALUES (?)');

my $entryUpdate= $dbh->prepare('UPDATE entry SET idCompType=? WHERE idCode=?');

my $expTypeUpdate=$dbh->prepare('UPDATE expType SET idExpClasse=? where ExpType=?');

while (<EXPCL>) {

chomp;

my ($idCode, $compType, $expClass) = split ' ';

$idCode =~ tr/a-z/A-Z/;

if (!$expClasses{$expClass}) {

$sthExpClasse->execute($expClass);

$expClasses{$expClass}=$dbh->last\_insert\_id('','','expClasses','idExpClass');

}

if (!$compTypes{$compType}) {

$sthcompType->execute($compType);

$compTypes{$compType}=$dbh->last\_insert\_id('','','compType','idCompType');

}

$expTypeUpdate->execute($expClasses{$expClass}, $expTypesbyCode{$idCode});

$entryUpdate->execute($compTypes{$compType},$idCode);

}

#Sequence

print "Sequences...";

my $codes=[];

my %CODES;

my $sthInsert= $dbh->prepare ("INSERT INTO sequence (idCode,chain,sequence,header) VALUES (?,?,?,?)");

open "SEQS", "pdb\_seqres.txt";

#pdb\_seqres.txt (FASTA)

#>101m\_A mol:protein length:154 MYOGLOBIN

#MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHP…

#>…

my $seq;

my $idPdb;

my $chain;

my $header;

while (<SEQS>) {

chomp;

if (/^>/) {

if ($seq) {

$seq =~ s/\n//g;

$idPdb =~ tr/a-z/A-Z/;

$chain =~ s/ //;

$sthInsert->execute ($idPdb,$chain,$seq, $header) || die $DBI::errstr;

$seq="";

}

/^>([^\_]\*)\_(.\*)mol:(\S\*) length:(\S\*) (.\*)/;

($idPdb,$chain,$header)=($1,$2,"$1 $2 $3 $4 $5");

print "$header\n";

}

else {$seq .= $\_};

};

print "ok\n";

disconnectDB($dbh);

**bdconn.pl**

Script to connect to mysql

#!/usr/bin/perl

# Connect script for DBI-Mysql

#

use DBI;

1;

sub connectDB {

my $driver = "mysql";

my $database = "pdb";

my $host = "localhost";

my $user="gelpi";

my $passwd = "jl12gb";

my $autoCommit = 1;

my $dsn = "DBI:$driver:database=$database;host=$host;user=$user;password=$passwd";

return DBI->connect ($dsn, undef, undef,

{RaiseError => 1, AutoCommit => $autoCommit});

}

sub disconnectDB {

my $dbh = shift;

return $dbh->disconnect;

}