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:

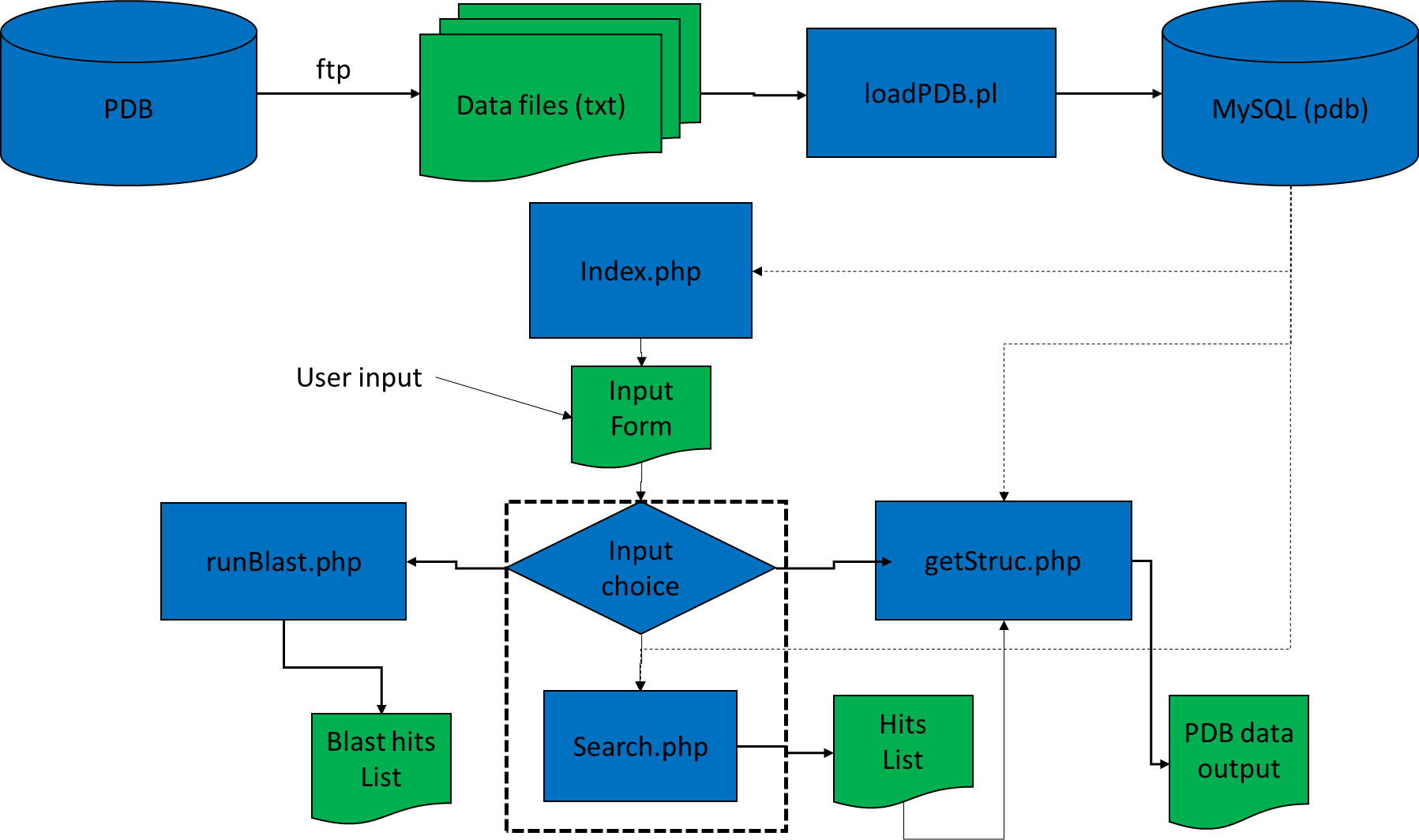
* PDB code (4 letter code). Will direct to a single entry if available
* Search Fields
  + Description, headers, authors, source (text search)
  + Resolution (numeric input)
  + Technique used (selection of possibles, X Ray, NMR, etc)
  + Type of molecule (selection of protein, nucleic acid, complex)
  + 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

#$baseDir = $\_SERVER['DOCUMENT\_ROOT'].'/PDBBrowser';

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

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

// Temporal dir, create if not exists

$tmpDir = "$baseDir/tmp";

if (!file\_exists($tmpDir))

mkdir($tmpDir);

// Blast details, change to adapt to local settings

$blastHome = "/home/dbw00/blast";

$blastDbsDir = "$blastHome/dbs";

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

$blastDbs = array("SwissProt" => "sprot", "PDB" => "pdb\_seqres.txt");

$blastCmdLine = "$blastExe -d $blastDbsDir/" . $blastDbs['PDB'] . " -p blastp -e 0.001 -v 100 -b 0 ";

// 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 = mysql\_query("SELECT \* from comptype") or print mysql\_error();

while ($rsF = mysql\_fetch\_array($rs))

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

//expTypes

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

while ($rsF = mysql\_fetch\_array($rs))

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

//expClasses

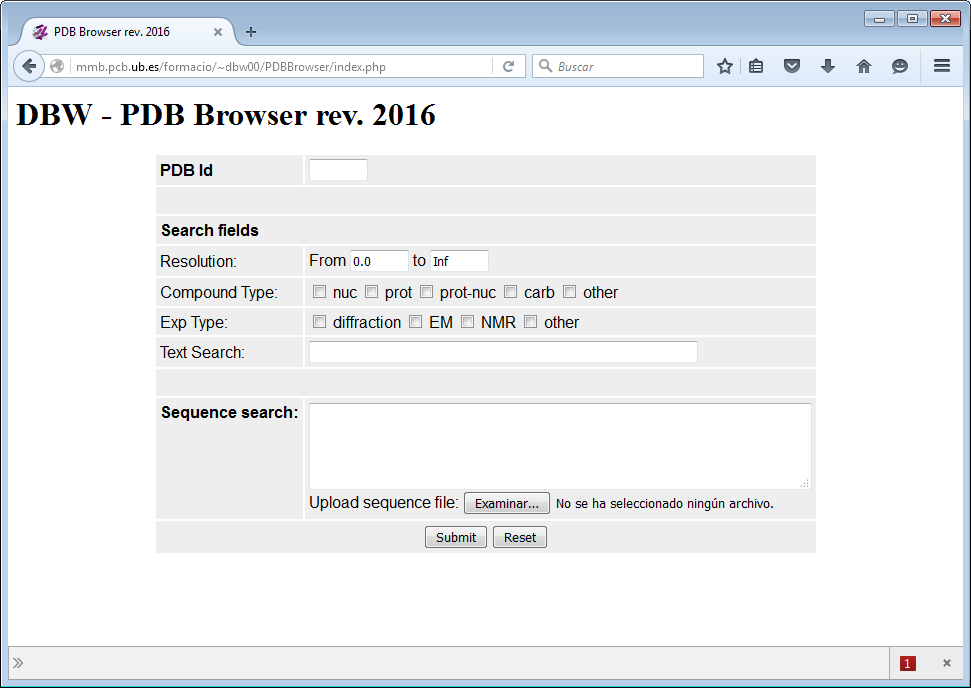
$rs = mysql\_query("SELECT \* from expClasse") or print mysql\_error();

while ($rsF = mysql\_fetch\_array($rs))

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

// 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 allow to clean it from the URL.

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

$\_SESSION['queryData'] = array(

'minRes' => '0.0',

'maxRes' => 'Inf',

'query' => ''

);

}

# All HTML settings and CSS is loaded from headerDBW

print headerDBW("PDB Browser rev. 2016");

#Main Form follows

?>

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

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

<tbody>

<tr>

<td><b>PDB Id</b></td>

<td><input type="text" name="idCode" value="" size="5" maxlength="4"/></td>

</tr>

<tr>

<td colspan="2">&nbsp;</td>

</tr>

<tr>

<td colspan="2"><b>Search fields</b></td>

</tr>

<tr>

<td>Resolution:</td>

<td>

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" />

</td>

</tr>

<tr>

<td>Compound Type:</td>

<td>

<?php

/\* We obtain the possible fields from the comptype table,

\* alternatively this could be done in globals.inc.php and stored as $compTypeArray

\* Then here we will use $compTypeArray instead

\*

\* foreach (array\_values($compTypeArray) as $c) {

\* <input type="checkbox" name="idCompType[<?php print $c ?>]" /> <?php print

\* $c['type'] ?>-->

\* }

\*

\* Note that input names build idCompType[] array

\*/

$rs = mysql\_query("SELECT \* from comptype");

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

?>

<input type="checkbox" name="idCompType[<?php echo $rsF['idCompType'] ?>]" /> <?php echo $rsF['type'] . "\n" ?>

<?php } ?>

</td>

</tr>

<tr>

<td>Exp Type:</td>

<td>

<?php

/\* We obtain the possible fields from the expClasse table,

\* that is a condensed version of expType, alternatively this could be done in

\* globals.inc.php

\* as compType

\* input names build idExpClasse[] array

\*/

$rs = mysql\_query("SELECT \* from expClasse order by ExpClasse");

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

?>

<input type="checkbox" name="idExpClasse[<?php echo $rsF['idExpClasse'] ?>]" /> <?php echo $rsF['expClasse'] . "\n" ?>

<?php } ?>

</td>

</tr>

<tr>

<td>Text Search:</td>

<td><input type="text" name="query1" value="<?php print $\_SESSION['queryData']['query1'] ?>" size="60" /></td>

</tr>

<tr>

<td colspan="2">&nbsp;</td>

</tr>

<tr>

<td valign="top"><b>Sequence search:</b></td>

<td>

<textarea name="seqQuery" rows="4" cols="60"></textarea><br>

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

</td>

</tr>

<tr>

<td colspan="2" align="center">

<input type="submit" value="Submit" />

<input type="reset" value="Reset" />

</td>

</tr>

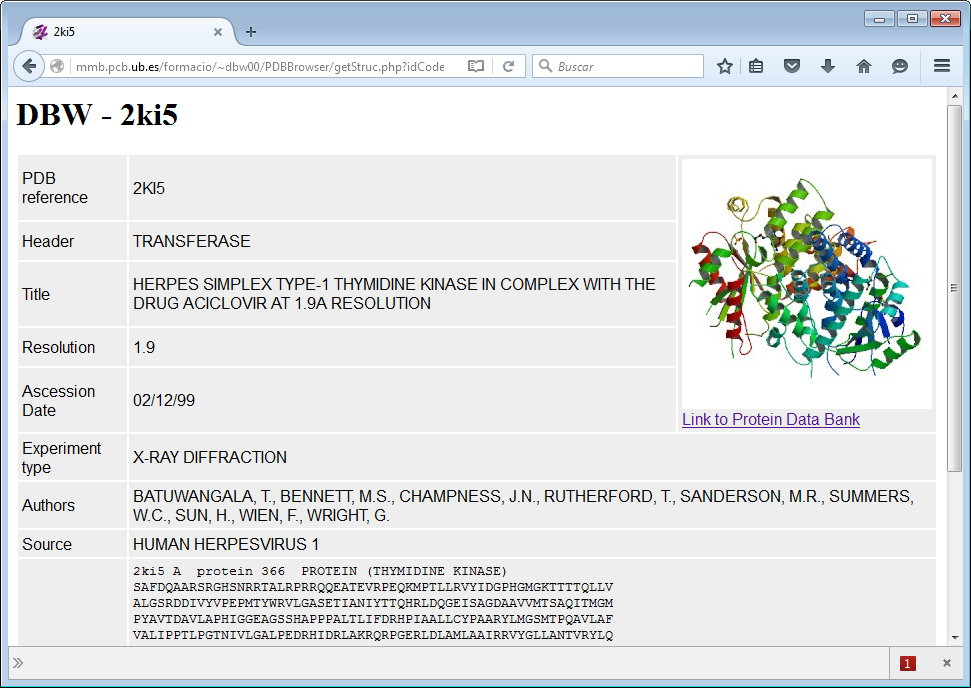
</tbody>

</table>

</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'] . "'";

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

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

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

} else {

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

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

$auts = array();

while ($rsAF = mysql\_fetch\_array($rsA))

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

print join(", ", $auts);

}

?>

</td>

</tr>

<tr>

<td>Source</td>

<td colspan="2">

<?php

// new DB query to get sources

$rsA = mysql\_query("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();

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

$sources = array();

while ($rsAF = mysql\_fetch\_array($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 = mysql\_query("SELECT \* from sequence s where s.idCode='" . $data['idCode'] . "' order by s.chain") or print mysql\_error();

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

while ($sq = mysql\_fetch\_array($rsA)) {

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

}

}

?>

</td>

</tr>

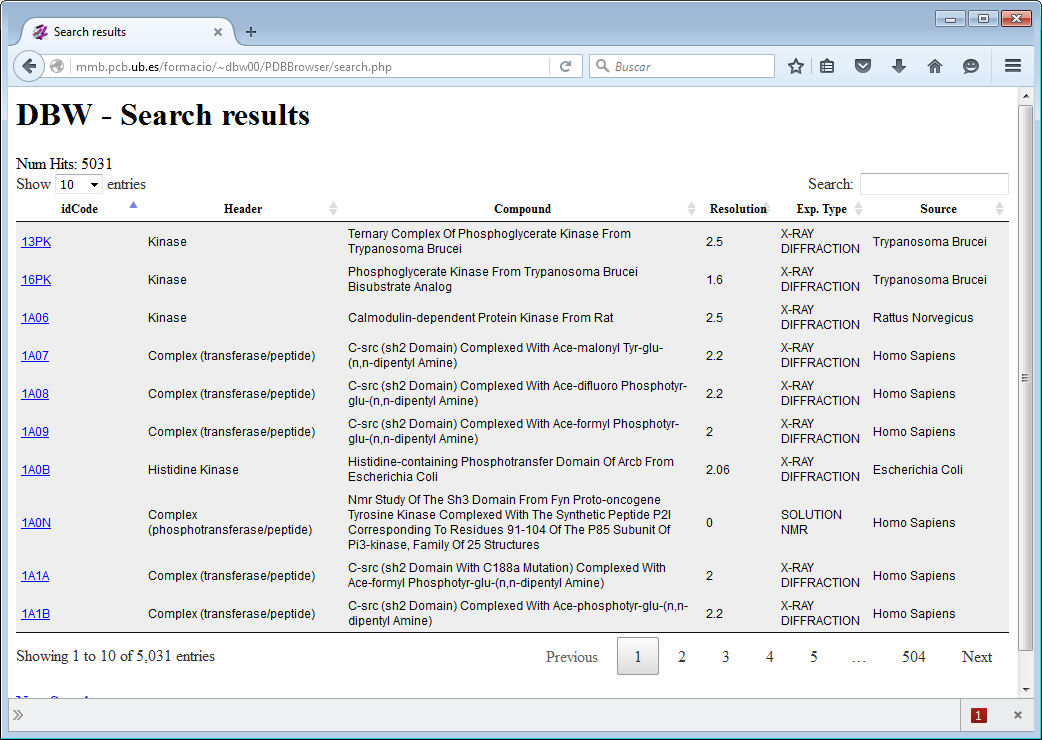
</tbody>

</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 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 = Array("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 = Array();

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 = Array();

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['query1']) {

$ORconds = array();

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

$ORconds[] = "MATCH (" . $field . ") AGAINST ('" . $\_REQUEST['query1'] . "' "

. "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 joined, 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, sequence sq,entry e 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;

// }

// $sql .= " LIMIT 50"; // Just to avoid long listings when testing

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

// DB query

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

// We check whether there are results to show

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

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

} else {

// Results table formated with DataTable (jQuery)

print headerDBW("Search results");

print "Num Hits: " . mysql\_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 ($rsF = mysql\_fetch\_array($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();

});

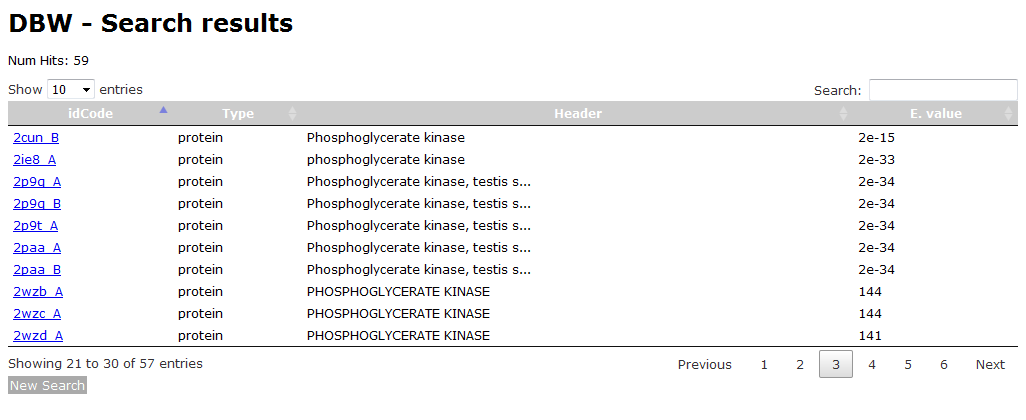
</script>

<?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

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

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

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

$i = 0;

while (!preg\_match('/Sequences producing/', $blast[$i]) and ( $i < count($blast)))

$i++;

$i++;

while (!preg\_match('/Database:/', $blast[$i]) and ( $i < count($blast))) {

if ($blast[$i])

$result[] = $blast[$i];

$i++;

}

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

// parsing hit following specific format, note that this format is not standard. It comes from the

// headers used to generate BLAST databases, this is from PDB

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

if (strlen($rr) > 1) {

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

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

?>

<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();

}

?>

<script type="text/javascript">

$(document).ready(function () {

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

});

</script>

**estil.css**

body {font-family: Verdana, Arial, sans-serif; font-size: 12px}

th {background-color: #cccccc; color:#ffffff}

.button {padding:2px;display: inline; color:#ffffff; background-color: #aaaaaa}

.button a {color:#ffffff; text-decoration:none}

.button a:hover {font-weight:bold}

**Include/bdconn.inc.php**

<?php

/\*

\* bdconn.inc.php

\* DB Connection

\*/

$host = "localhost";

$dbname = "pdb";

$user = "dbw00";

$password = "master00";

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

($id = mysql\_select\_db($dbname)) or die(mysql\_error());

**Include/libDBW.inc.php**

<?php

function headerDBW($title) {

return "<html>

<head>

<title>$title</title>

<link rel=\"stylesheet\" type=\"text/css\" href=\"estil.css\">

<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\">

<h1>DBW - $title</h1>

";

}

function footerDBW() {

return '

</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);