CS 490 Project: Release Version

May 5th, 2020

Alexander Chavkin (alc26@njit.edu) Front End

Nicholas Patterson (np595@njit.edu) Middle

Yazmin Villalba (yav3@njit.edu) Back End

```
<?php
//                         Filename change to CS490RC.php for RC, the beta
version is there, I just made a copy for this
error_reporting(E_ERROR | E_WARNING | E_PARSE | E_NOTICE);
ini_set('display_errors', 1);

$backurl = 'https://web.njit.edu/~yav3/backEndCS490Betha.php';

$requestID = $_POST['RequestType'];
$data = $_POST['data'];

if ($requestID == 'login'){

        $post = http_build_query(array('RequestType' => $requestID, 'data' =>
        $data));

        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $post);

        $result = curl_exec($ch);
        echo $result; //Echos login return from back to front
        curl_close($ch);

}

elseif ($requestID == 'CreateQuestion'){
//Creates the question then sends data to back to store in database
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=>
        $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif ($requestID == 'GetQuestions'){//Send the request data forward for the
//back to retreive the question data from the database to then send to front
//Data will be holding the request type for back to determine which to send
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=> $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);
        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif ($requestID == 'createExam'){
//Data will be holding the exam created to save in database
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=>
        $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif ($requestID == 'listExams'){
//Data will be sending the list of exams created to the front
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=>
        $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif($requestID == 'showExam'){
//Data will be sending the exam chosen to the front to display
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=>
        $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);
```

```
}

elseif($requestID == 'submitExam'){ //Perform auto-grader here!

        $ARGS_START_DELIMITER = "(";
        $ARGS_END_DELIMITER = ")";
        $CASE_DELIMITER = "BORDERLINEN";
        $RETURN_DELIMITER = "DRAGONLORD";

        $ucid = $data['ucid'];
        $examName = $data['exaName'];
        $questionIDs = $data['questionsid'];
        $answers = $data['answers'];
        $maxScores = $data['points'];

        $tData = array('questionsid' => $questionIDs);
        $requesting = 'retrieve';

        $datas = http_build_query(array('RequestType' => $requesting, 'data'
=> $tData));
        $chr = curl_init();

        curl_setopt($chr, CURLOPT_URL, $backurl);
        curl_setopt($chr, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($chr, CURLOPT_POSTFIELDS, $datas);

        $resultEn = curl_exec($chr);
        curl_close($chr);

        $result = json_decode($resultEn, true);

        $scores = array();
        $comments = array();
        $expecteds = array();
        $resulting = array();

        $deductTest = array();
        $deductName = array();
        $deductDef = array();
        $deductColon = array();
        $deductCons = array();

        for($i = 0; $i < count($questionIDs); ++$i){

                //Deducted for both testcases
                $topic = $result[$i]['topic'];
                $question = $result[$i]['questText'];
                $testcasesS = $result[$i]['questTest'];
                $answer = stripslashes($answers[$i]);
                $constrain = $result[$i]['constrain'];
                //One max score for each question for total points compared
to
                //total missed
                $functionName = substr($testcasesS, 0, strpos($testcasesS,
                $ARGS_START_DELIMITER));
                $fname = substr($answer, 0, strpos($answer,
$ARGS_START_DELIMITER));
                $fname = preg_replace("/def /", "", $fname);
                $testcases = explode($CASE_DELIMITER, $testcasesS);
                $inputs = array();
                $expectedReturns = array();
                $S = $maxScores[$i];
                $testFile =

'/afs/cad.njit.edu/u/n/p/np595/public_html/CS490Work/test.py';

                $NAMED = 3;
                $DEFD = 3;
                $COLOND = 2;
                $CONSD = 5;
                $TESTD = (int)(($S - $NAMED - $COLOND -
$CONSD)/count($testcases));

                $totDed = array();
                $p = 0;
                foreach($testcases as $k){
                        $expectedReturns[$p] = substr($k, strpos($k,
                        $RETURN_DELIMITER) + 1);
                        $expectedReturns[$p] =

str_replace("LITERALPLUSCHARACTER","+",$expectedReturns[$p]);

                        $inputs[$p] = substr($k, strpos($k,
                        $ARGS_START_DELIMITER), strpos($k,
                        $ARGS_END_DELIMITER) - strpos($k,
                        $ARGS_START_DELIMITER) + 1);
                        $inputs[$p] = str_replace("LITERALPLUSCHARACTER","+",
                        $inputs[$p]);

                        $p = 1 + $p;
                }

                //This grabs the user made inputs to allow their program to
run

                $tempAnswer = $answer;

                $deductColon[$i] = 0;
                $hasColon = colon_check($answer);
                if(! $hasColon){
                        $deductColon[$i] = $COLOND;
                        $tempAnswer = add_colon($tempAnswer);
                }

                if(strpos($answer,"def $fname") === false){
                        $deductDef[$i] = $DEFD;
                        $tempAnswer = "def $tempAnswer";
                }
                else{
                        $deductDef[$i] = 0;
                }
```

```
file_put_contents($testFile, $tempAnswer);

clearstatcache();
if($constrain == 'For'){
        $fitsConstraint = for_check($answer);
}
elseif($constrain == 'While'){
        $fitsConstraint = while_check($answer);
}
elseif($constrain == 'Print'){
        $fitsConstraint = print_check($answer);
}
else{
        $fitsConstraint = true;
}

if(! $fitsConstraint){
        $deductCons[$i] = $CONSD;
}
else{
        $deductCons[$i] = 0;
}
foreach($inputs as $l){
        if($constrain != 'Print' || ! $fitsConstraint){
                file_put_contents($testFile,
                "\nprint($fname$l)",FILE_APPEND);
        }
        else{
                file_put_contents($testFile,
                "\n$fname$l", FILE_APPEND);
        }
}

$returnSet = array();

exec("python test.py", $returnSet, $exec_return_code);

//If answers != testcase, no points, if second testcase, then
//points per testcase by total of testcases
if(count($returnSet) == count($expectedReturns)){
        for($j = 0; $j < count($expectedReturns); ++$j){
                $returnSet[$j] != $expectedReturns[$j] ?
                $totDed[$j] = $TESTD : $totDed[$j] = 0;
        }
}

else if($exec_return_code){
        for($j = 0; $j < count($expectedReturns); ++$j){
                if(!isset($returnSet[$j]))
                $returnSet[$j] = "(Python crashed!)";

                $returnSet[$j] != $expectedReturns[$j] ?
                $totDed[$j] = $TESTD : $totDed[$j] = 0;
        }
}

$deductTest[$i] = $totDed;
```

```
                $fitsConstraint ? $deductCons[$i] = 0 :
                $deductCons[$i] = $CONSD;

                $a = strtok($answer, "\n");
                while(ctype_space($a))
                        $a = strtok("\n");
                $r = preg_match('/def[ \t]+' . $functionName . '[ \t]*\(.+/',
$a);

                $r ? $deductName[$i] = 0 : $deductName[$i] = $NAMED;

                $ALLD = ($TESTD*count($testcaseS)) + $COLOND + $NAMED +
$CONSD;

                $TOTALD = $deductName[$i] + $deductColon[$i] +
$deductCons[$i];
                foreach($totDed as $t)
                        $TOTALD += $t;

                if(($maxScores[$i] - $ALLD)&&($TOTALD == $ALLD))
                        $totDed[count($testcases)-1] += $maxScores[$i] -
$ALLD;

                $deductDef[$i] = 0;

                $scores[$i] = $maxScores[$i] - $deductName[$i] -
$deductColon[$i] - $deductCons[$i];

                foreach($totDed as $test)
                        $scores[$i] -= $test;
                $comments[$i] = "";
                $expecteds[$i] = $expectedReturns;
                $resulting[$i] = $returnSet;
        }

        str_flatten("HACKMAGICK", $expecteds);
        str_flatten("HACKMAGICK", $resulting);
        str_flatten(", ", $deductTest);

//Comments are nothing since the autograder doesn't input comments nor gets
//when student completes exam, so they are empty

        $tData = array('comments' => $comments, 'ucid' => $ucid, 'exaName' =>
        $examName, 'questionsid' => $questionIDs, 'answers' => $answers,
        'scores' => $scores, 'maxScores' => $maxScores, 'expectedAnswers' =>
        $expecteds, 'resultingAnswers' => $resulting,
        'deductedPointscorrectName' => $deductName,
'deductedPointsPerEachTest'
        => $deductTest, 'deductedPointsHasDef' => $deductDef,
        'deductedPointsMissingColon' => $deductColon,
'deductedPointsConstrain'
        => $deductCons);

        $datas = http_build_query(array('RequestType' => 'gradingExam',
'data' => $tData));
```

```
        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $resulting = curl_exec($ch);
        curl_close($ch);
        echo $resulting;

}

elseif($requestID == 'showGradedExam'){
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=> $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif($requestID == 'modifyGradedExam'){
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=> $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif($requestID == 'listGradedExams'){

        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=>
        $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

elseif($requestID == 'listGradedExamsStudent'){
        $datas = http_build_query(array('RequestType' => $requestID, 'data'
=>
        $data));

        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $backurl);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $datas);

        $result = curl_exec($ch);
        echo $result;
        curl_close($ch);

}

function str_flatten($delim, &$arr){
        foreach($arr as &$a)
                $a = implode($delim, $a);
}

function colon_check($answer){
        $a = strtok($answer, "\n");
        while(ctype_space($a))
                $a = strtok("\n");
        $r = preg_match('/def[ \t]+[A-Za-z0-9_]+[ \t]*\(.*\)[ \t]*:/', $a);
        return $r;
}

function for_check($answer){
//Checks for loop with the key to search with and the three occurrences of
for
//loops. The variable that's looping with, the range of a number, and a
string.
//The \t checks for any potential spaces that could occur within range or in
//the string so it will continue to verify them anyway
        $r = preg_match('/for([ \t]+|[ \t]*\([ \t]*)[A-Za-z_](([ \t]*,)?[
        \t]*[A-Za-z0-9_][ \t]*)*\)?[ \t]+in/', $answer);
        return $r;
}

function while_check($answer){
        $r = preg_match('/while([ \t]+|\()*[^\t ]+.*:[ \t]*/', $answer);
        return $r;
}

function print_check($answer){
        $r = preg_match('/print([ \t]+|\().+/', $answer);
        $s = preg_match('/return([ \t]+|\().+/', $answer);
        return $r && ! $s;
}
```

```
function add_colon(&$answer){
        $s = array();
        $a = strtok($answer, "\n");
        while(ctype_space($a)){
                $s[] = a;
                $a = strtok("\n");
        }
        $a .= ":";
        $s[] = $a;
        while($a = strtok("\n"))
                $s[] = $a;
        return implode("\n", $s);
}

?>
```