# Homework 5 Server-side programming

## CSC 435 Web Programming

Contents: HTML, JS, Fetch/Ajax and PHP.

The codes needs to be executed on a local server.

You might find some basic knowledge on Fetch() API useful:

<https://developers.google.com/web/ilt/pwa/working-with-the-fetch-api>

Number of a team allowed: max 2 people.

Please write the names of your teams both on the folder name as well as comments in the code.

Each team just needs to submit one homework.

Due date: end of the day, April 26th, 2020.

Total points: 50 (pts) + Bonus 10(pts)

1. (20pts) Recipe generator

In this exercise, you will use folder and file processing to create a web service (e.g. like the number API we showed in class) that takes a single parameter “name” and outputs a randomly generated recipe idea based on the letters in the name.

The first letter of the name will correspond to the first letter of the recipe name, and the rest of the letters of the name will each correspond to a randomly generated ingredient for that recipe.

Download and unzip the provided recipe-generator.zip. In this folder, there are two sub directories, foods and ingredients. Each contains exactly 26 txt files each corresponding to a lower-cased letter of the English Alphabet. foods includes txt files listing recipe names, and ingredients includes txt files include ingredients.

There is also a starting recipe-generator.php provided with comments for you to fill in as well as two helper functions.

The following is an example output for a request for recipe-generator.php?name=xiao

xiao's Xiaolongbao

Directions:

In a bowl, mix:

1 cup Ice water

1 Almond

1 box of Oreos

Cook for 4 minutes and serve!

Make sure to write your recipegenerator.php in the same directory as the other to sub-folders. Follow the steps in the starter code to finish the web service. Feel free to edit the ingredients and recipe so they look less unrealistic.

1. (30pts) Trivia.

The goal of the assignment is make a trivia game with JavaScript, JSON, Ajax and PHP.

The page looks like the following (we will provide you with HTML and CSS, you will complete the .php and .js):

The user click “list categories” and in the categories div the categories are listed. They click “Next Question”, the next question showed up. If they then click “Show Answer”, the answer will be revealed.

A screenshot of a cell phone

Description automatically generated

Step 1: Download the triva.html, trivia.css and triva.zip from blackboard. Unzip the triva.zip. This folder contains several folders of trivia questions as in .txt files. Make sure your main code is outside these folders.

Step 2: Write a trivia.php code to read the .txt files (from particular category specified by the fetch()) and output as JSON. You can use $\_GET[“mode”], scandir and json\_encode().

What you need to do is given the category query parameter from the trivia.js, create a **random question** (e.g. using array\_rand()) send it back to the browser.

There are many ways to open the .txt files, for example:

$triviafiles = "../trivia/";

$categoryName = strtolower($\_GET["name"]);

$trivia = glob($triviafiles . $categoryName . "/\*.txt");

To print the question and answer, you can:

print(json\_encode(array("question" => $question, "answer" => $answer)));

Step 3: Write a trivia.js code to call the PHP code and render the content back to the browser.

The first step is do display the categories. You might find the following code snippets useful.

function fetchCategories() {

let hxr = new XMLHttpRequest()

hrx.onload = displayCategories;

hrx.open("GET", "trivia.php?mode=categories");

hrx.send();

}

You can achieve the above function also with a fetch API. To do so, you need to get the list of categories with a GET method and display them with displayCategories() function.

To fetch a question from a particular category.

function showTrivia() {

let url = "trivia.php?mode=category";

if (currentCategory) {

url += "&name=" + currentCategory;

}

fetch(url)

.then(checkStatus)

.then(JSON.parse)

.then(displayQuestion);

}

Bonus (10pts) :

1. add new questions into proper categories of your own choices by writing a writequestion. php
2. Ask the user to create a question and answer and write it to the directory.
3. Create new categories of questions and add question and answer.
4. Explore Trivia API (<https://opentdb.com/api_config.php>) to use the database for questions.

Grading:

To receive a passing grade (60%) for the question, you can skip the whole “category” option. The user can just click “next question” and one of the questions will show up. To achieve this, you can just read from one of the .txt file or you can even create your own question (just hardcode a JSON) and request that JSON back to browser. However, if you want to get an impressive grade, you should actually follow the steps above to make the game work to its full functions.