ALL TALK AND NO FRACTION

Structure and technologies

Project Overview

- All Talk and No FrACTION is an interactive web application which gamifies practicing the addition and subtraction of fractions.
- It engages the user by providing them a customizable avatar, in-game leveling and progression tracking as well as various sets of questions of varying difficulty level.

Application Components

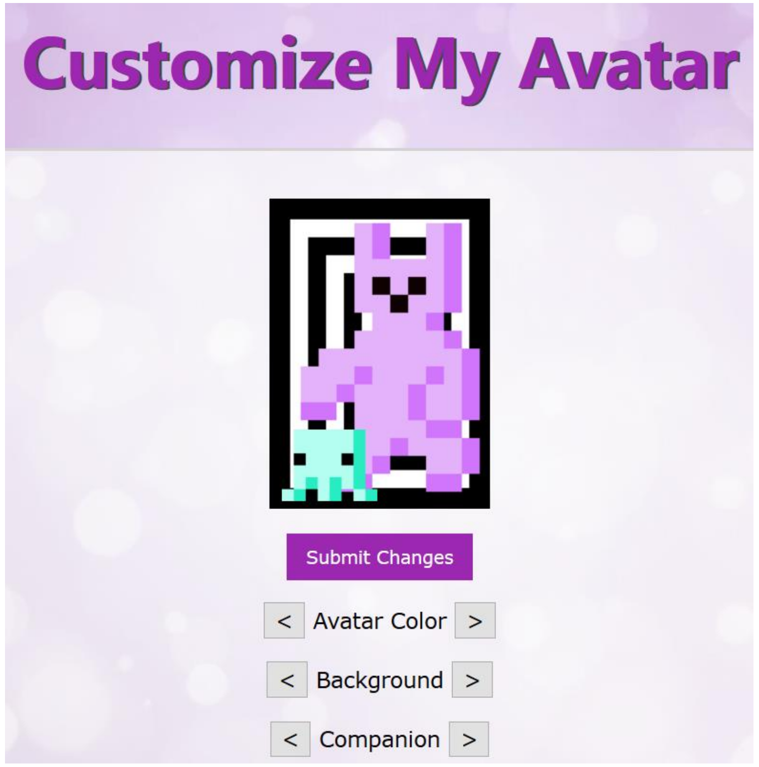
- Client side:
  - HTML
  - CSS
  - Javascript (no external libraries)
- Server side:
  - PHP
- Other technologies:
  - PostgreSQL database (Google Cloud SQL Instance)
  - Deployment to Google Cloud App Engine

Why so many components from scratch?

- HTML, CSS and Javascript were all new to me; though using some external libraries may have made my work easier, I believe it is important to learn to do something myself before I use someone else’s work to abstract it all.
- After taking the course CSI4139 last session, which assumed some understanding of authentication, I was interested in learning how to implement a basic user and session management system.

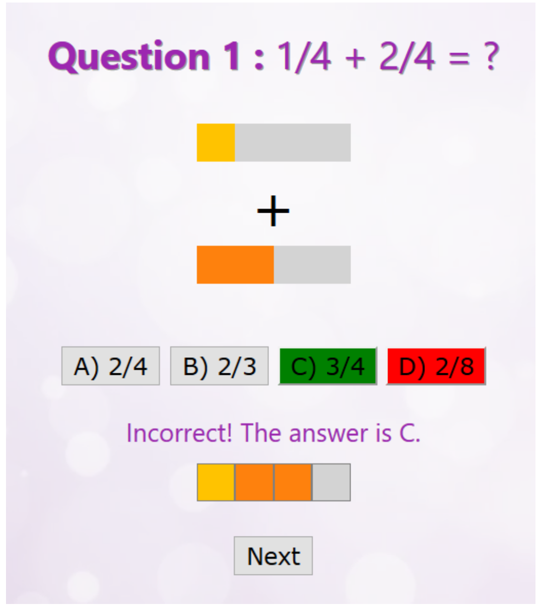
Client-Side Functionality

- Avatar Customization (Javascript)
  - Allows the user to cycle through different options for the avatar, pet and background by clicking on the arrow buttons



Client-Side Functionality

- Meter representation of fractions (Javascript)
  - For each question, equal-length meters are generator for each of the operands using rectangles of different colours, emphasizing the importance of finding a common denominator
- Show answer (Javascript)
  - Once an answer choice is selected, the correct answer is shown along with a meter representation of the fraction.


Question 1 : 1/4 + 2/4 = ?

+

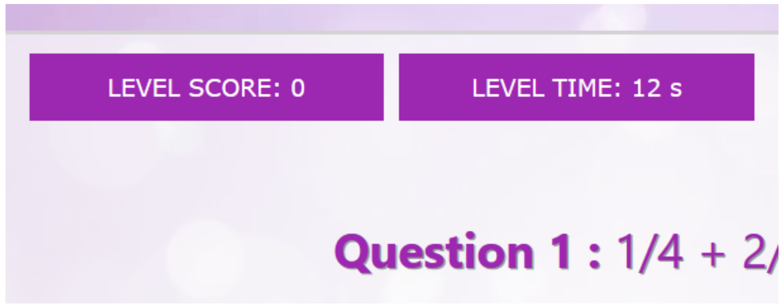
A) 2/4 B) 2/3 C) 3/4 D) 2/8

Incorrect! The answer is C.

Next

Client-Side Functionality

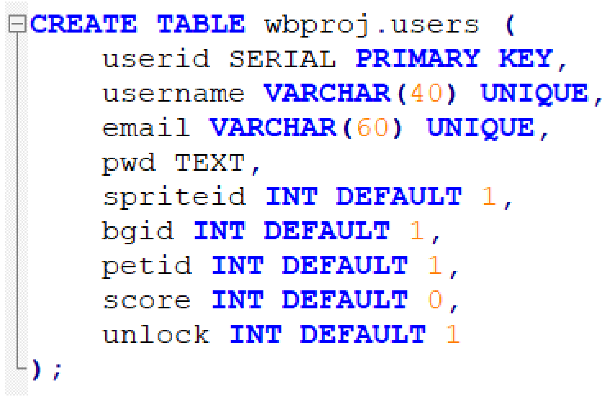
- Timer for Challenge Levels (JavaScript)
  - A timer counts down from a predetermine length of time from the beginning of a level to the end
  - The timer is paused when the correct answer is displayed, then resumed when the next question is loaded



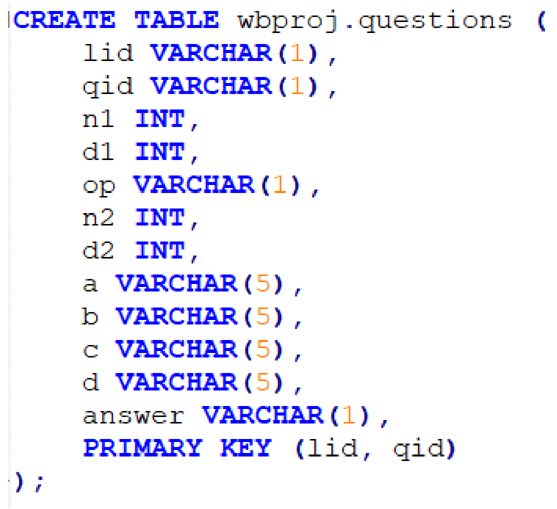
Question 1 : 1/4 + 2/

Database Schema

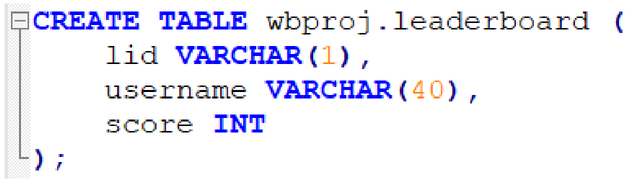
- Three tables: Users, Questions, Leaderboard



```
CREATE TABLE wbproj.users (
    userid SERIAL PRIMARY KEY,
    username VARCHAR(40) UNIQUE,
    email VARCHAR(60) UNIQUE,
    pwd TEXT,
    spriteid INT DEFAULT 1,
    bgid INT DEFAULT 1,
    petid INT DEFAULT 1,
    score INT DEFAULT 0,
    unlock INT DEFAULT 1
);
```



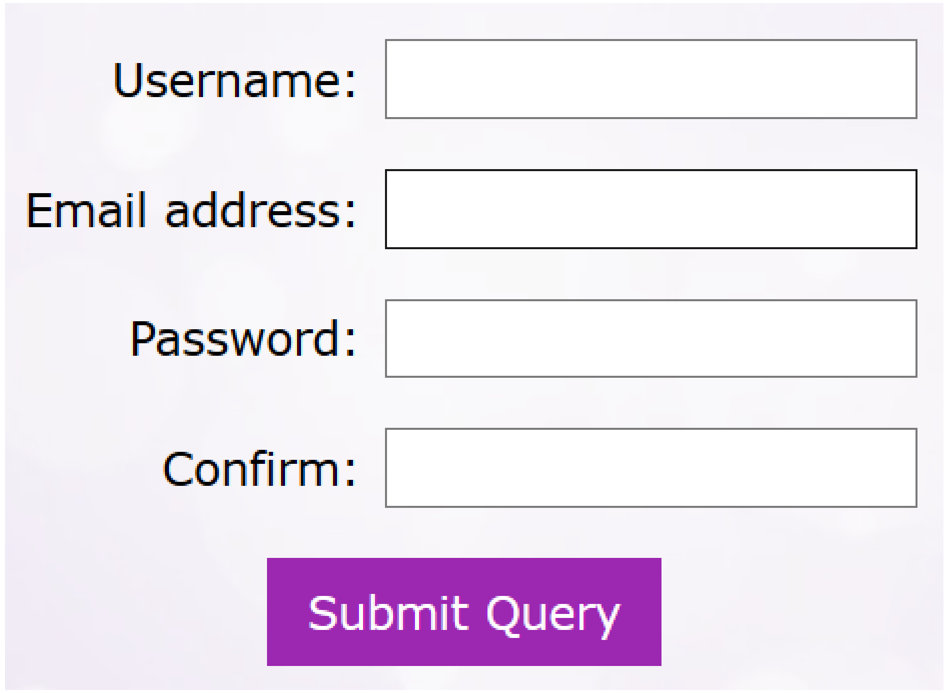
```
CREATE TABLE wbproj.questions (
    lid VARCHAR(1),
    qid VARCHAR(1),
    n1 INT,
    d1 INT,
    op VARCHAR(1),
    n2 INT,
    d2 INT,
    a VARCHAR(5),
    b VARCHAR(5),
    c VARCHAR(5),
    d VARCHAR(5),
    answer VARCHAR(1),
    PRIMARY KEY (lid, qid)
);
```



```
CREATE TABLE wbproj.leaderboard (
    lid VARCHAR(1),
    username VARCHAR(40),
    score INT
);
```

Server-Side Functionality

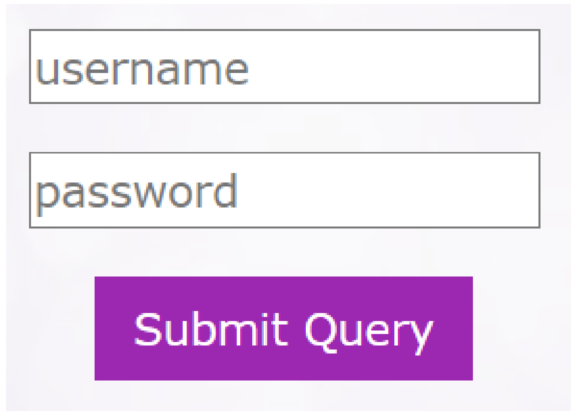
- User and Session Management
  - Users are created through the Register page; if the username has not been taken, then the user information is added to the Users table of the database
  - The user is then redirected to the login page (the user is not, at this point, logged in).

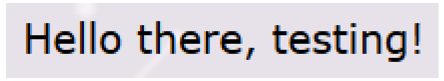


Username:	
Email address:	
Password:	
Confirm:	

Server-Side Functionality

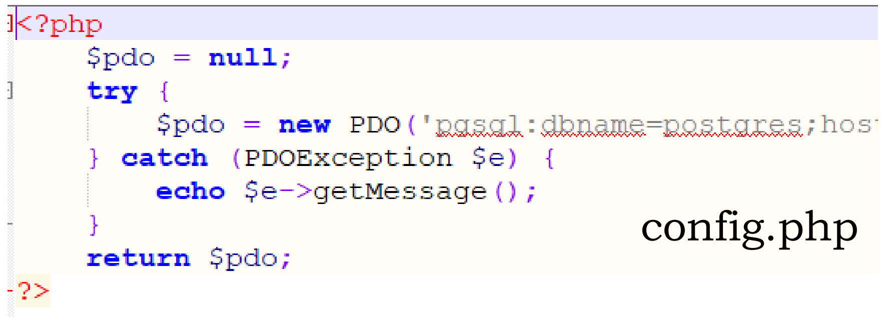
- User and Session Management (ctd)
  - When a user attempts to login, the username and hashed password inputted are compared to the entries in the User table of the database: if there is a match, the user is logged in.
  - Session variables track the user's userid and username as he or she navigates the application.




Hello there, testing!

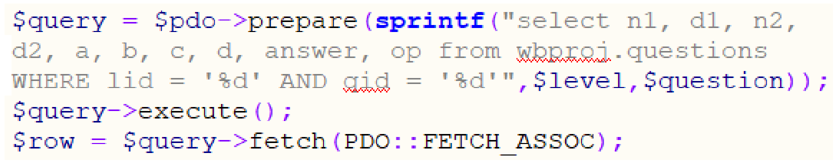
Server-Side Functionality

- Fetching fraction questions
  - For each question of each level, the two operands, four possible answers and the correct answer are fetched from the database.
- Avatar customization
  - Initial avatar configuration is fetched from the database
  - Any submitted changes are loaded into the database



```php
<?php
    $pdo = null;
    try {
        $pdo = new PDO('pgsql:dbname=postgres;hos
    } catch (PDOException $e) {
        echo $e->getMessage();
    }
    return $pdo;
?>
```

config.php

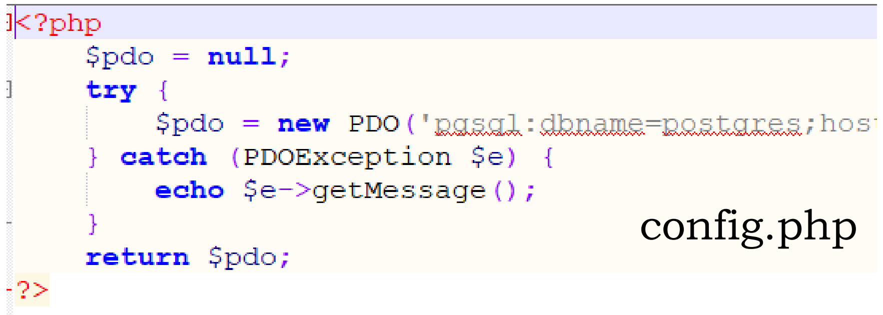


```php
$query = $pdo->prepare(sprintf("select n1, d1, n2,
d2, a, b, c, d, answer, op from wbproj.questions
WHERE lid = '%d' AND qid = '%d'",$level,$question));
$query->execute();
$row = $query->fetch(PDO::FETCH_ASSOC);
```

play.php

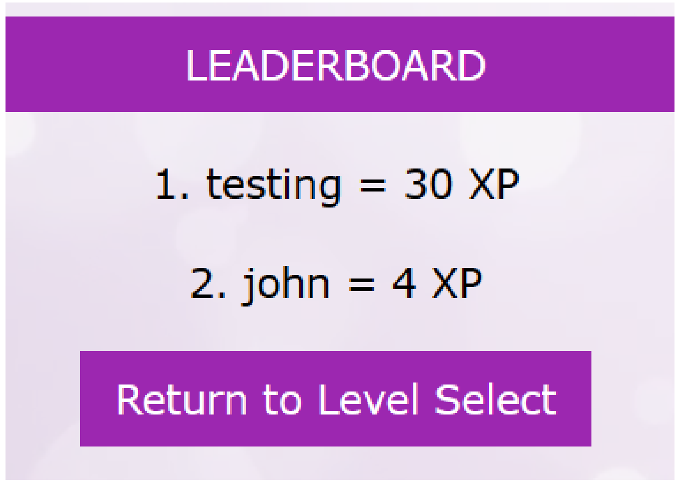
Server-Side Functionality

- Updating player score
  - At the end of each level, the XP scored are added to the total player score in the database.
- Updating leaderboard
  - At the end of each level, the username and score for the level are added as an entry in the Leaderboard table
  - The top three scores (and corresponding usernames) are fetched and displayed in the page leaderboard



```php
<?php
    $pdo = null;
    try {
        $pdo = new PDO('pgsql:dbname=postgres;hos
    } catch (PDOException $e) {
        echo $e->getMessage();
    }
    return $pdo;
?>
```

config.php



LEADERBOARD

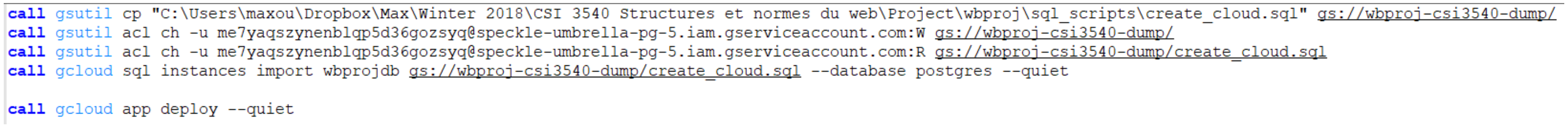
1. testing = 30 XP

2. john = 4 XP

Return to Level Select

Deployment

- The application is deployed from a local DEV environment to an instance of the Google Cloud App Engine, the PROD environment, using the deploy.bat script with gcloud and gsutils commands.



```
call gsutil cp "C:\Users\maxou\Dropbox\Max\Winter 2018\CSI 3540 Structures et normes du web\Project\wbproj\sql_scripts\create_cloud.sql" gs://wbproj-csi3540-dump/
call gsutil acl ch -u me7yaqszynenblqp5d36gozsyq@speckle-umbrella-pg-5.iam.gserviceaccount.com:W gs://wbproj-csi3540-dump/
call gsutil acl ch -u me7yaqszynenblqp5d36gozsyq@speckle-umbrella-pg-5.iam.gserviceaccount.com:R gs://wbproj-csi3540-dump/create_cloud.sql
call gcloud sql instances import wbprojdb gs://wbproj-csi3540-dump/create_cloud.sql --database postgres --quiet

call gcloud app deploy --quiet
```

- The database consists of an instance of Google SQL Cloud tied to the same administrative account as the web application. Fake credentials for this account can be found in the “fake_settings.yaml” file for evaluation purposes.