```php
<?php

    /**
     * functions.php
     *
     * Computer Science 50
     * Problem Set 7
     *
     * Helper functions.
     */

    require_once("constants.php");

    /**
     * Apologizes to user with message.
     */
    function apologize($message)
    {
        render("apology.php", ["message" => $message]);
        exit;
    }

    /**
     * Facilitates debugging by dumping contents of variable
     * to browser.
     */
    function dump($variable)
    {
        require("../templates/dump.php");
        exit;
    }

    /**
     * Logs out current user, if any.  Based on Example #1 at
     * http://us.php.net/manual/en/function.session-destroy.php.
     */
    function logout()
    {
        // unset any session variables
        $_SESSION = [];

        // expire cookie
        if (!empty($_COOKIE[session_name()]))
        {
            setcookie(session_name(), "", time() - 42000);
        }

        // destroy session
        session_destroy();
    }

    /**
     * Returns a stock by symbol (case-insensitively) else false if not found.
     */
    function lookup($symbol)
    {
        // reject symbols that start with ^
        if (preg_match("/^\^/", $symbol))
        {
            return false;
        }

        // reject symbols that contain commas
        if (preg_match("/,/", $symbol))
        {
            return false;
        }

        // headers for proxy servers
        $headers = [
            "Accept" => "*/*",
            "Connection" => "Keep-Alive",
            "User-Agent" => sprintf("curl/%s", curl_version()["version"])
        ];

        // open connection to Yahoo
        $context = stream_context_create([
            "http" => [
                "header" => implode(array_map(function($value, $key) { return sprintf("%s: %s\r\n",
$key, $value); }, $headers, array_keys($headers))),
                "method" => "GET"
            ]
        ]);
        $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?f=snl1&s={$symbol}", "r",
false, $context);
        if ($handle === false)
        {
            // trigger (big, orange) error
            trigger_error("Could not connect to Yahoo!", E_USER_ERROR);
            exit;
        }

        // download first line of CSV file
        $data = fgetcsv($handle);
        if ($data === false || count($data) == 1)
        {
            return false;
        }

        // close connection to Yahoo
        fclose($handle);

        // ensure symbol was found
        if ($data[2] === "0.00")
        {
            return false;
        }

        // return stock as an associative array
        return [
            "symbol" => $data[0],
            "name" => $data[1],
            "price" => floatval($data[2])
        ];
    }

    /**
     * Executes SQL statement, possibly with parameters, returning
     * an array of all rows in result set or false on (non-fatal) error.
     */
    function query(/* $sql [, ... ] */)
    {
        // SQL statement
        $sql = func_get_arg(0);

        // parameters, if any
        $parameters = array_slice(func_get_args(), 1);

        // try to connect to database
        static $handle;
        if (!isset($handle))
        {
            try
            {
                // connect to database
                $handle = new PDO("mysql:dbname=" . DATABASE . ";host=" . SERVER, USERNAME, PASSWORD);

                // ensure that PDO::prepare returns false when passed invalid SQL
                $handle->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
            }
            catch (Exception $e)
            {
                // trigger (big, orange) error
                trigger_error($e->getMessage(), E_USER_ERROR);
                exit;
            }
        }

        // prepare SQL statement
        $statement = $handle->prepare($sql);
        if ($statement === false)
        {
            // trigger (big, orange) error
            trigger_error($handle->errorInfo()[2], E_USER_ERROR);
            exit;
        }

        // execute SQL statement
        $results = $statement->execute($parameters);

        // return result set's rows, if any
        if ($results !== false)
        {
            return $statement->fetchAll(PDO::FETCH_ASSOC);
        }
        else
        {
            return false;
        }
    }

    /**
     * Redirects user to destination, which can be
     * a URL or a relative path on the local host.
     *
     * Because this function outputs an HTTP header, it
     * must be called before caller outputs any HTML.
     */
    function redirect($destination)
    {
        // handle URL
        if (preg_match("/^https?:\/\//", $destination))
        {
            header("Location: " . $destination);
        }

        // handle absolute path
        else if (preg_match("/^\//", $destination))
        {
            $protocol = (isset($_SERVER["HTTPS"])) ? "https" : "http";
            $host = $_SERVER["HTTP_HOST"];
            header("Location: $protocol://$host$destination");
        }

        // handle relative path
        else
        {
            // adapted from http://www.php.net/header
            $protocol = (isset($_SERVER["HTTPS"])) ? "https" : "http";
            $host = $_SERVER["HTTP_HOST"];
            $path = rtrim(dirname($_SERVER["PHP_SELF"]), "/\\");
            header("Location: $protocol://$host$path/$destination");
        }

        // exit immediately since we're redirecting anyway
        exit;
    }

    /**
     * Renders template, passing in values.
     */
    function render($template, $values = [])
    {
        // if template exists, render it
        if (file_exists("../templates/$template"))
        {
```

```php
            // extract variables into local scope
            extract($values);

            // render header
            require("../templates/header.php");

            // render template
            require("../templates/$template");

            // render footer
            require("../templates/footer.php");
        }

        // else err
        else
        {
            trigger_error("Invalid template: $template", E_USER_ERROR);
        }
    }

?>
```