|  |  |  |  |
| --- | --- | --- | --- |
| logo-CvSU-2  Republic of the Philippines  **CAVITE STATE UNIVERSITY**  **Bacoor City Campus**  SHIV, Molino VI, City of Bacoor  **DEPARTMENT OF COMPUTER STUDIES**  **ITEC 106: IT Elective 2 (Web Systems and Technologies 2)**  **Activity Sheet**  **2nd Semester, A.Y. 2021-2022** | | | |
| **MIDTERM LABORATORY EXAM** | | | |
|  | | | |
| DIRECTION  * Create a user registration system using PHP that saves and retrieves the data in the database. Your program should include the following:   Login form:   * Username and Password * Include error message “Incorrect Username or password”. * If fields are empty vice-versa, include “Enter your username” or “Enter your password”. * If login successful, include a message “Welcome” and prompt to another webpage with a message “Hello Sir Good Morning” and provide a change password and logout button under the message. * In the logout button, when clicked, include a message “Are you sure you want to log-out?” with options “Yes/No” if yes, it should go back to the login form.   Signup form:   * Include Name, User Name, Password, and Re-Password. * If fields are empty vice-versa, include error message “Password required”, “Username required” * If username is already taken, include error message: “The username is taken try another”. * If signup successful, include a message “Your account has been created successfully”   Change Password form:   * Include Old Password, New Password, Confirm New Password. * Include buttons “Confirm” and “Cancel” * Include error messages “Unmatched fields”, “Old Password Incorrect” * Include message “You have successfully changed your password” it should go back to the login form. * Provide a screenshot and source code of your work. * Present a video recording of your program to check its functionality. * Design the interface in HTML and CSS. | | | |
| **rubrics for grading:** | | |
|  | | |
| **Category** | **Description** | |
| **Program**  **Execution (25pts)** | Program executes correctly with no syntax or runtime errors. | |
| **Correctness**  **(25pts)** | Program produces correct answers or appropriate results for all  inputs tested. | |
| **Layout**  **(25pts)** | The page has an exceptionally attractive and usable layout. It is  easy to locate all important elements. White space, graphic  elements and/or alignment are used effectively to organize  material. | |
| **Logic**  **(25pts)** | Program logic is correct, with no known boundary errors, and  no redundant or contradictory conditions. | |

Republic of the Philippines

**CAVITE STATE UNIVERSITY**

**Bacoor City Campus**

SHIV, Molino VI, City of Bacoor

**DEPARTMENT OF COMPUTER STUDIES**

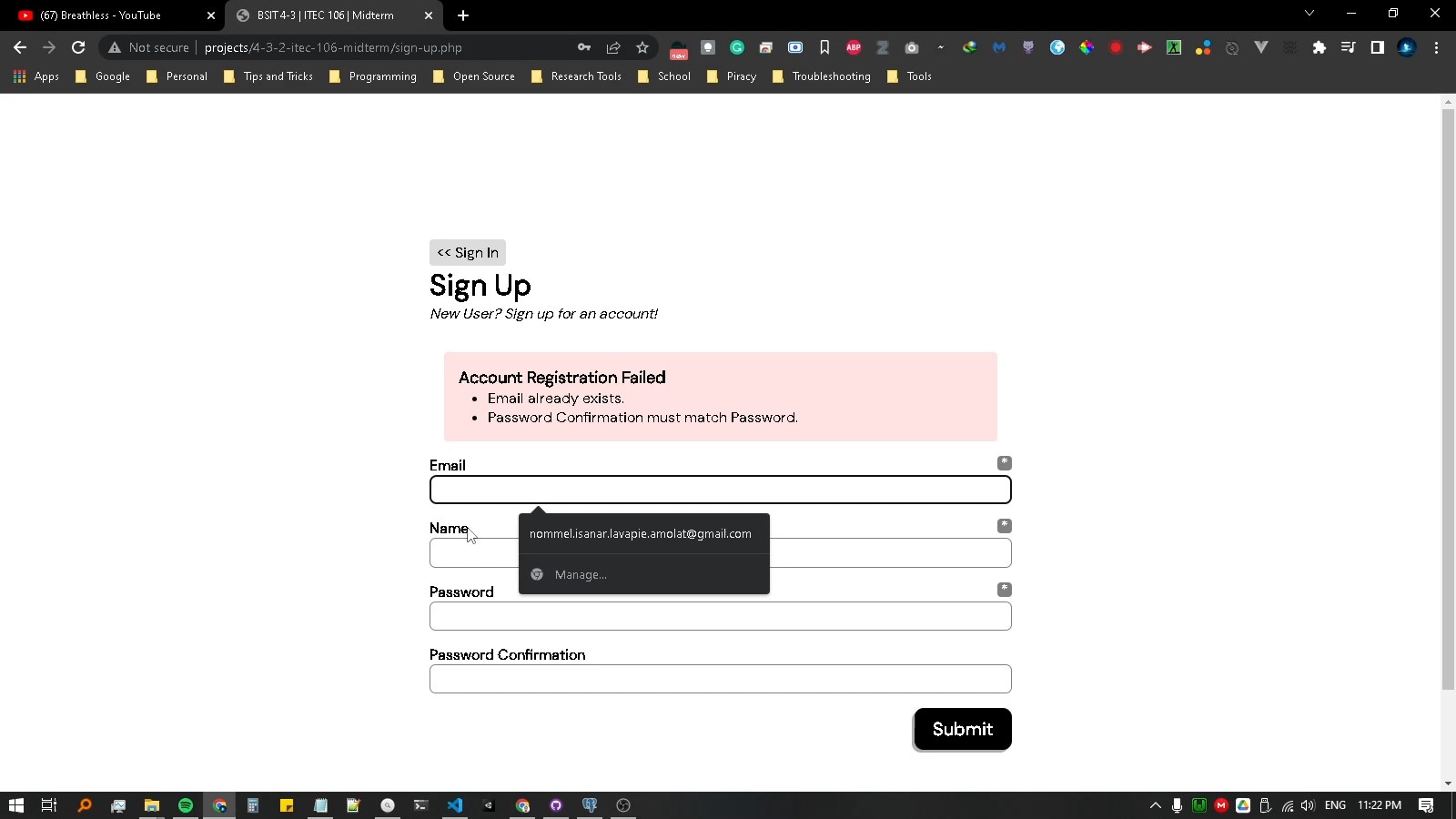
**ITEC 106: IT Elective 2 (Web Systems and Technologies 2)**

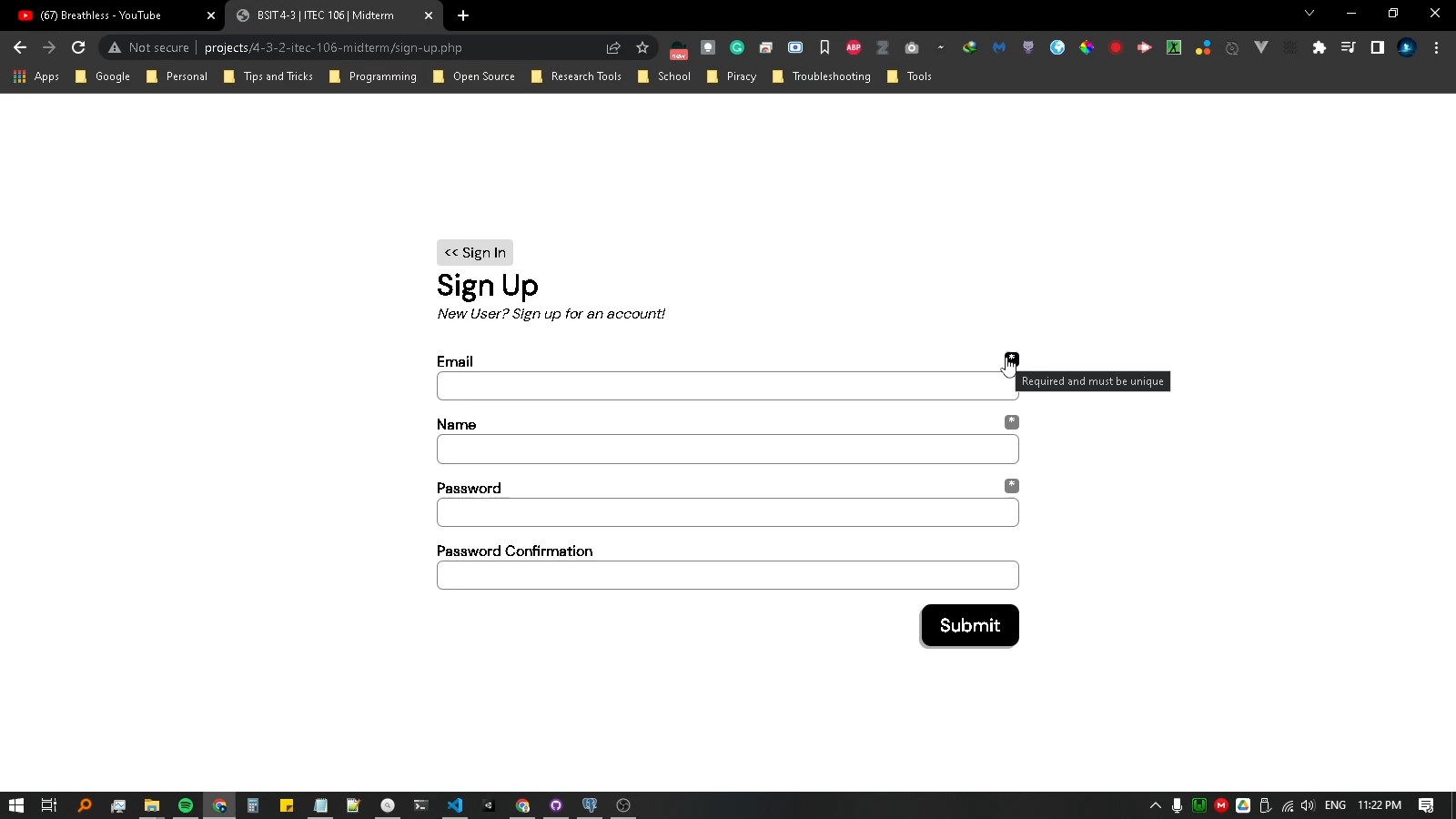
**Activity Sheet**

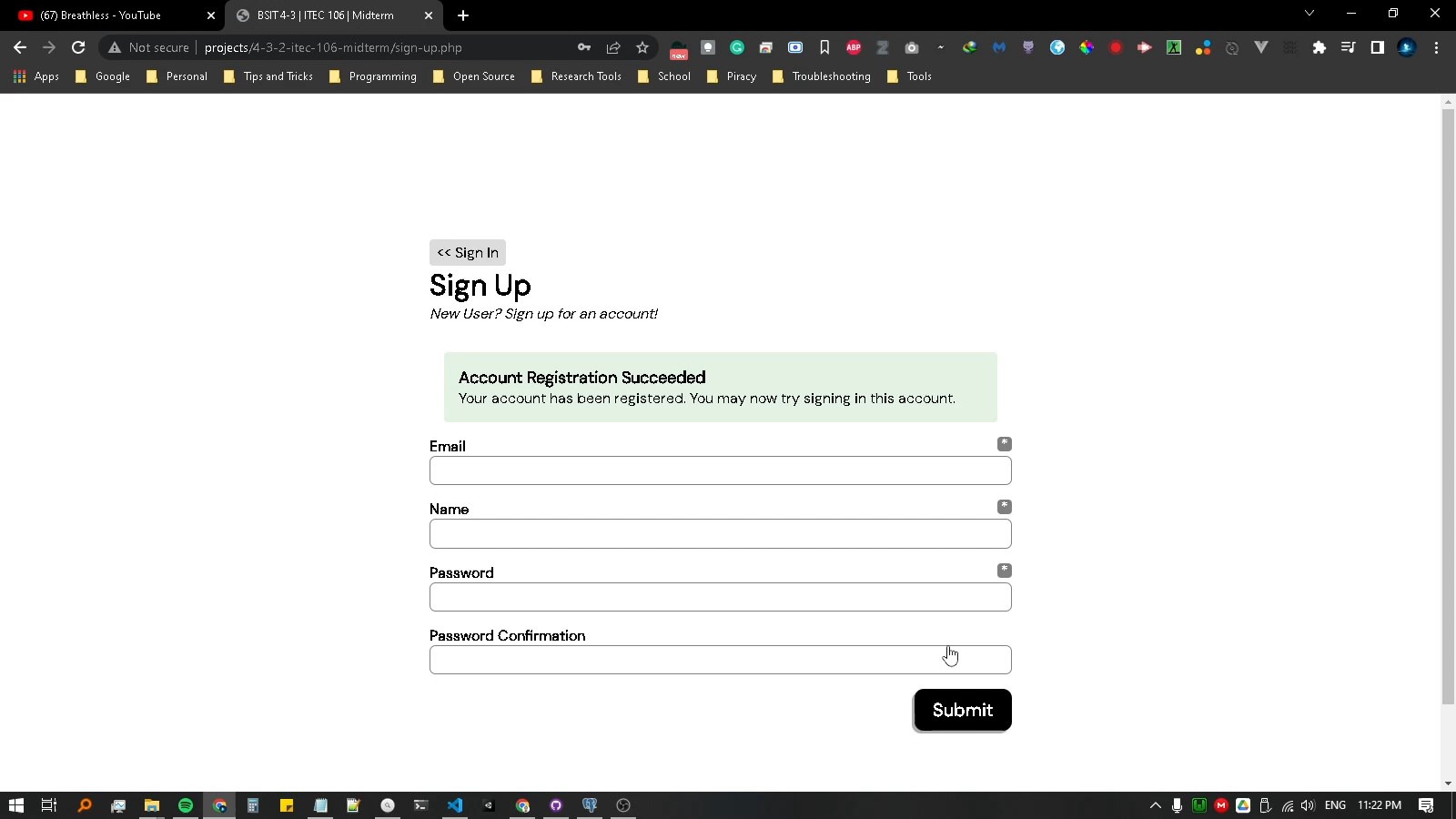
**2nd Semester, A.Y. 2021-2022**

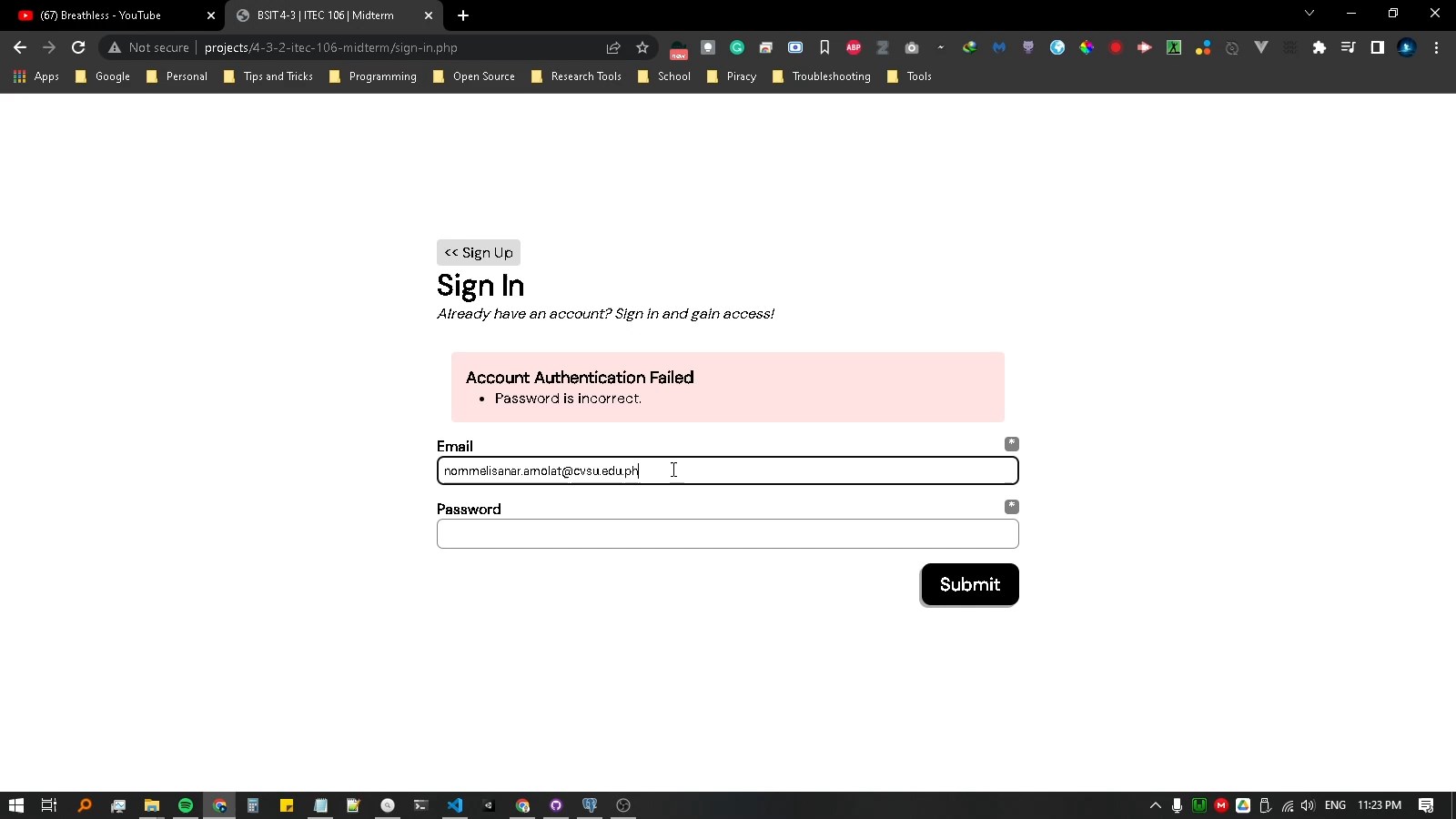
|  |  |
| --- | --- |
| **Name: Nommel Isanar L. Amolat** | **Date: May 1, 2022** |
| **Course/Yr&Section: BSIT 4-3** | **Score:** |

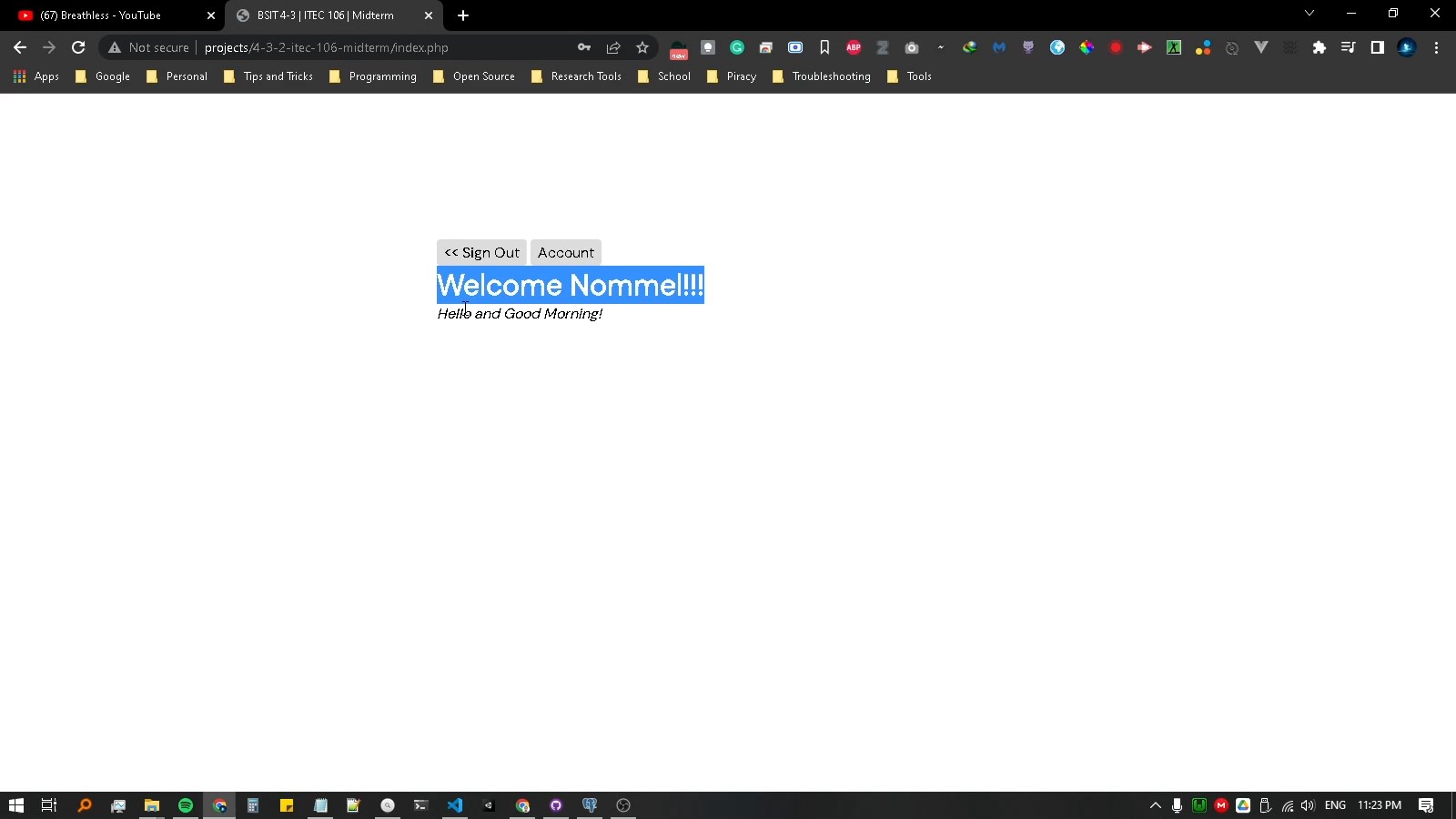
|  |
| --- |
| **screenshot** |
|  |
|  |
|  |

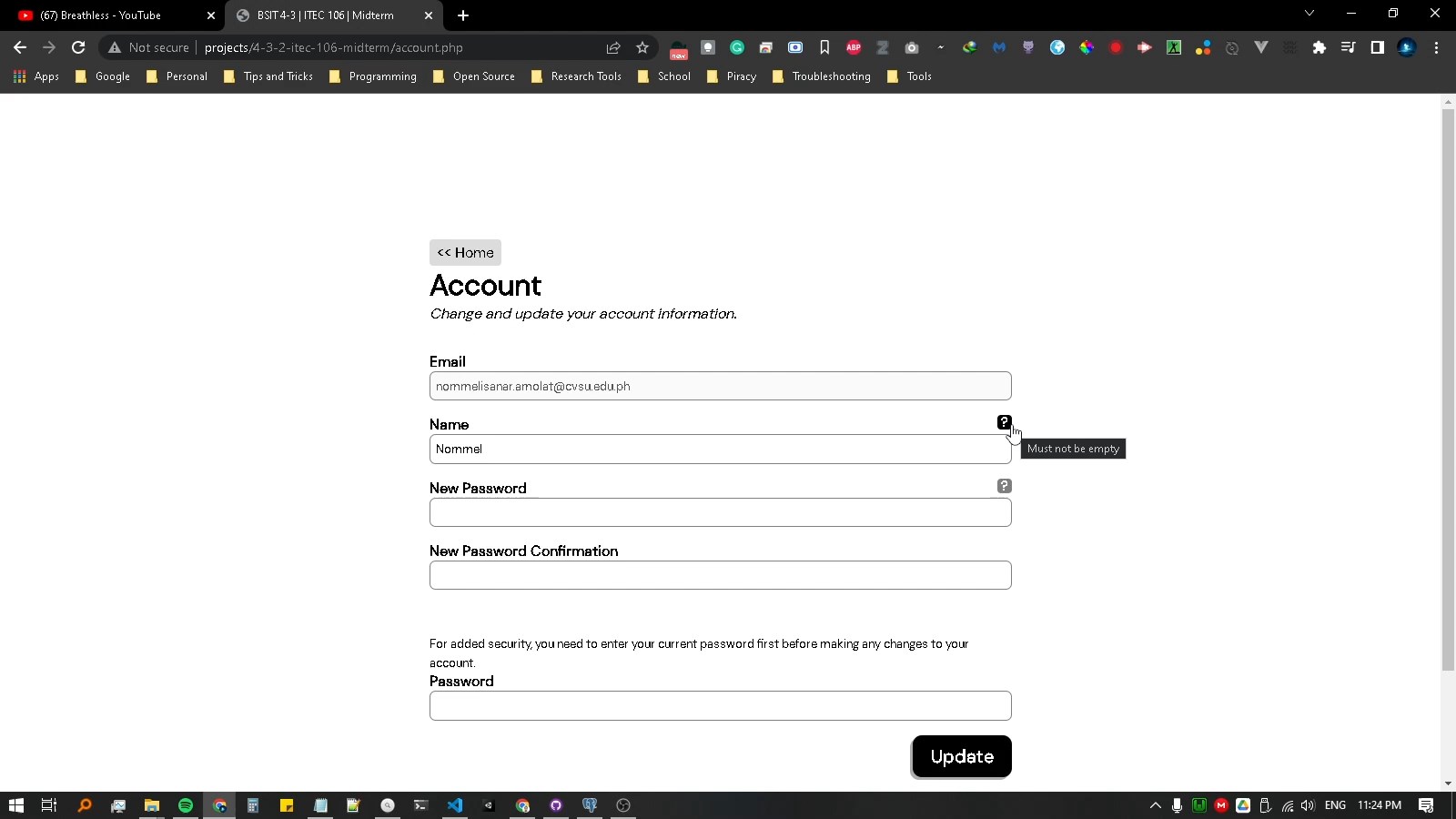


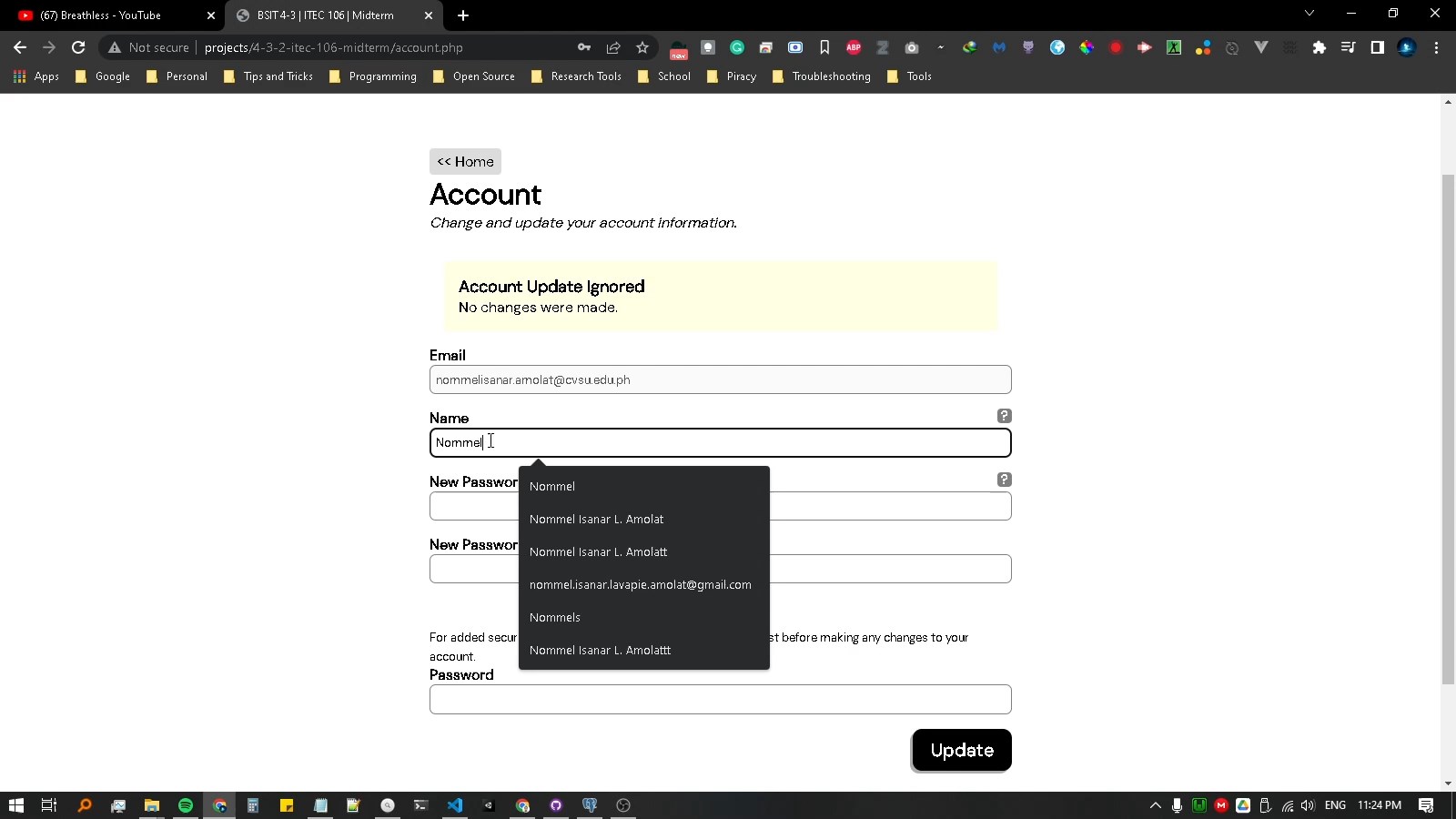


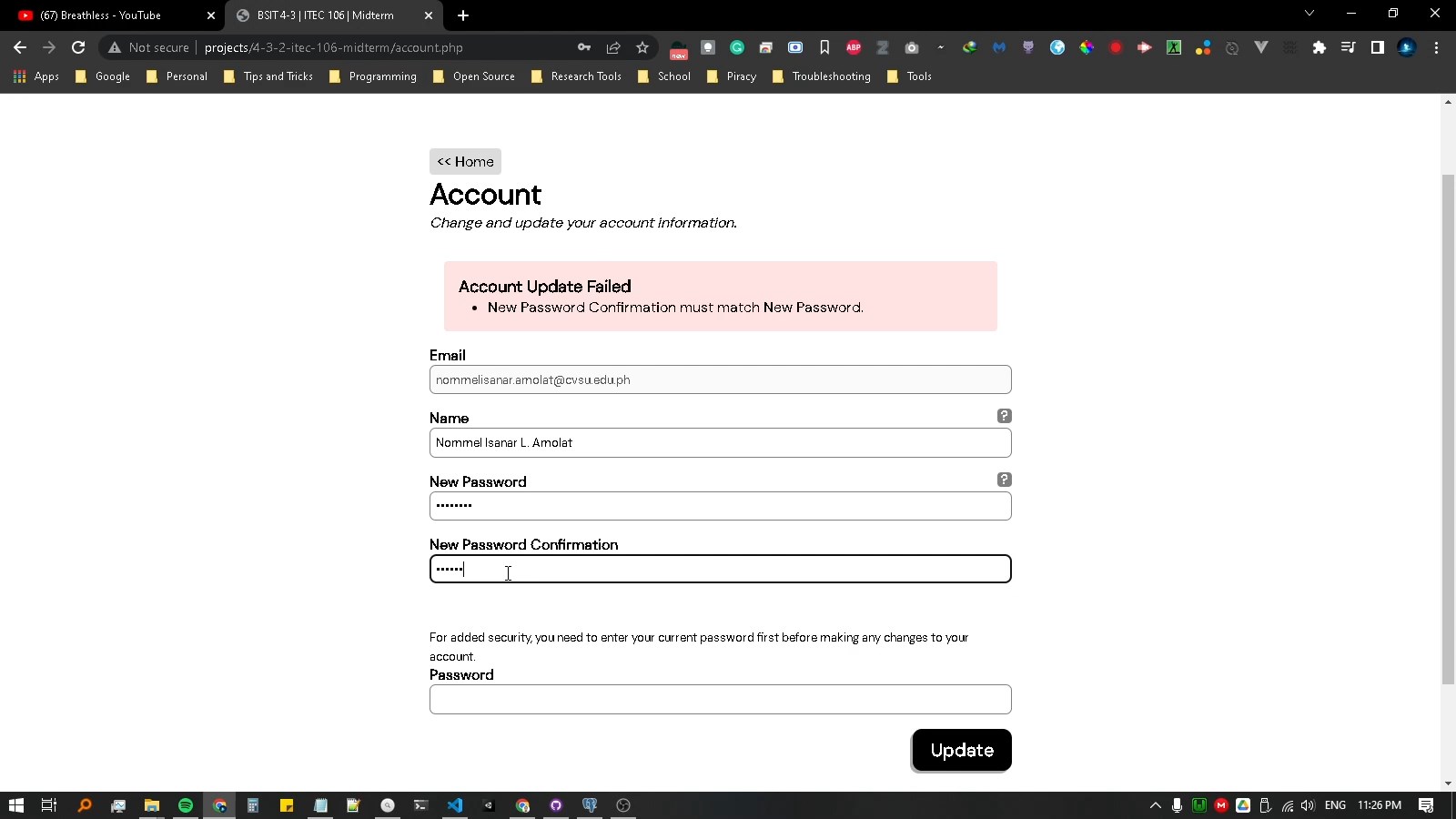


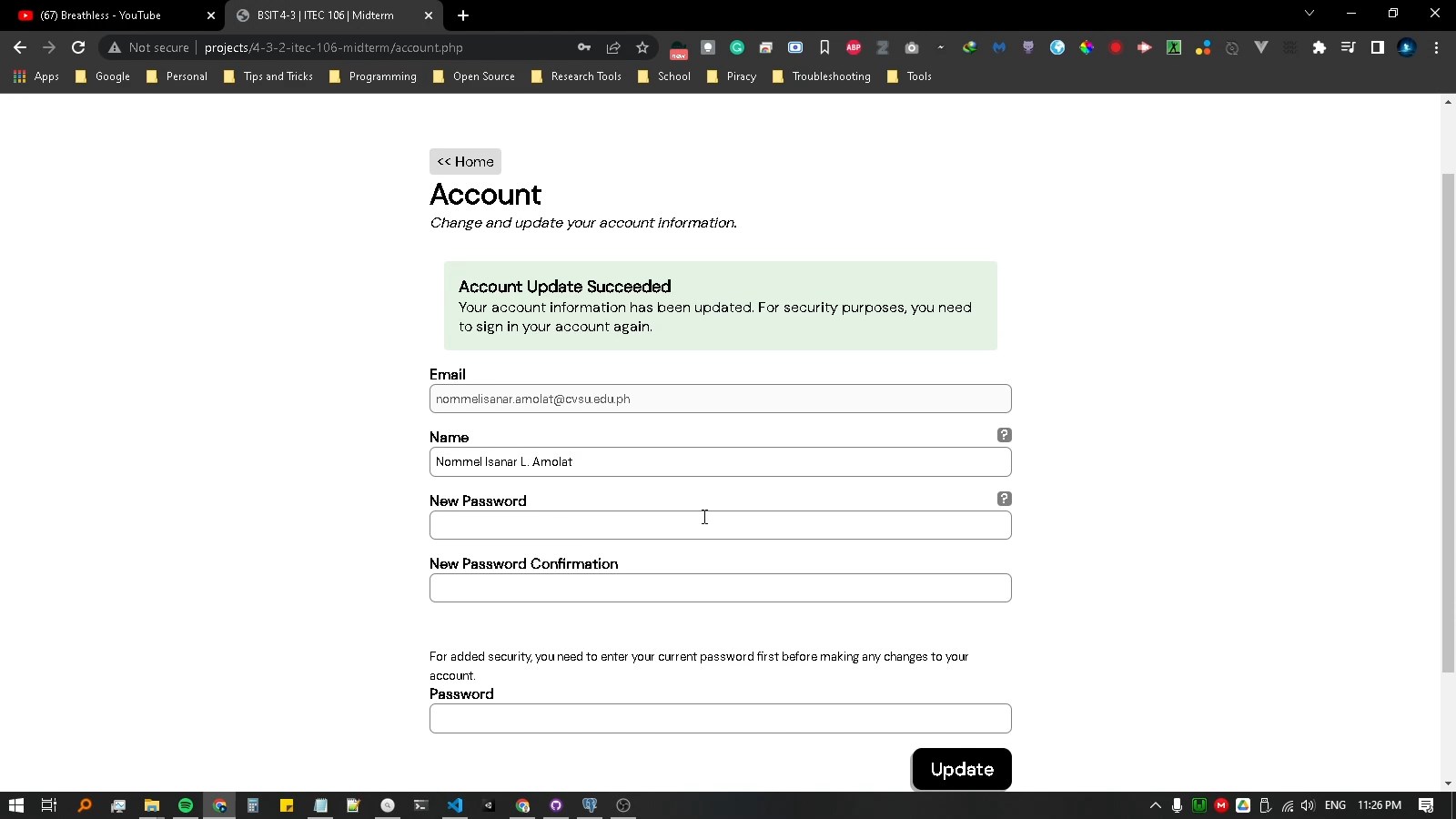












|  |
| --- |
| **source code** |
|  |

<!-- \_init.php -->

<?php

    require\_once("./vendor/autoload.php");

    require\_once("./utils.php");

    require\_once("./helpers.php");

    require\_once("./database.php");

    require\_once("./tables.php");

?>

<!-- \_start.php -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>BSIT 4-3 | ITEC 106 | Midterm</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <div id="main">

<!-- \_end.php -->

    </div>

</body>

</html>

<!-- uitls.php -->

<?php

    function render($file, $params=[]) {

        foreach($params as $key => $value) {

            global ${$key};

            ${$key} = $value;

            // $GLOBALS[$key] = $value;

        }

        include($file);

    }

    function stitch(...$args) : string {

        return join(

            " ",

            array\_filter($args,

                fn($e) => is\_string($e) && strlen(trim($e)) > 0));

    }

    function check($array, $key, $value) {

        if(!isset($array[$key])) return false;

        if(is\_callable($value)) return $value($array[$key]);

        return $array[$key] == $value;

    }

    function pop(&$array, ...$keys) {

        $vals = [];

        foreach($keys as $key) {

            if(isset($array[$key])) {

                array\_push($vals, $array[$key]);

                unset($array[$key]);

            }

        }

        $c = count($vals);

        return $c == 0 ? null : ($c == 1 ? $vals[0] : $vals);

    }

    function keep(&$array, ...$keys) {

        $rkeys = array\_filter($keys, fn($k) => !in\_array($k, $keys));

        return pop($array, ...$rkeys);

    }

    function apply(&$array, $callback) {

        if(!is\_callable($callback)) return;

        foreach($array as $key => $val) {

            $array[$key] = $callback($key, $val);

        }

    }

?>

<!-- helpers.php -->

<?php

    require\_once("./utils.php");

    function user() {

    }

    function signIn($user) {

        startSession();

        $\_SESSION["user\_id"] = $user["id"];

    }

    function signOut() {

        startSession();

        session\_unset();

        session\_destroy();

    }

    function signedIn($true=null, $false=null) {

        StartSession();

        if(isset($\_SESSION["user\_id"])) {

            try {

                $query = query("select \* from users where id = ?", [$\_SESSION["user\_id"]]);

                if($query->rowCount() > 0) {

                    if($true) header("Location: {$true}");

                    $user = $query->fetch();

                    return $user;

                }

            } catch(Throwable $th) {}

        }

        if($false) header("Location: {$false}");

        return null;

    }

    function startSession() {

        if(session\_status() === PHP\_SESSION\_NONE) session\_start();

    }

?>

<!-- database.php -->

<?php

    require\_once("./vendor/autoload.php");

    $dotenv = Dotenv\Dotenv::createImmutable(\_\_DIR\_\_);

    $dotenv->safeLoad();

    function getDSN($dbms, $host, $port, $dbname, $user, $password) {

        return "{$dbms}:host={$host};port={$port};dbname={$dbname};user={$user};password={$password}";

    }

    function query($string, $params=[]) {

        global $pdo;

        $query = $pdo->prepare($string);

        $query->execute($params);

        return $query;

    }

    try {

        $dsn = array(

            "dbms" => $\_ENV["DBMS"],

            "host" => $\_ENV["HOST"],

            "port" => $\_ENV["PORT"],

            "dbname" => $\_ENV["DBNAME"],

            "user" => $\_ENV["USER"],

            "password" => $\_ENV["PASSWORD"]

        );

        if(isset($\_ENV["DATABASE\_URL"])) {

            $db = parse\_url($\_ENV["DATABASE\_URL"]);

            $dsn["dbms"] = "pgsql";

            $dsn["host"] = $db["host"];

            $dsn["port"] = $db["port"];

            $dsn["dbname"] = ltrim($db["path"], "/");

            $dsn["user"] = $db["user"];

            $dsn["password"] = $db["pass"];

        }

        $pdo = new PDO(

            getDSN($dsn["dbms"], $dsn["host"], $dsn["port"],

                $dsn["dbname"], $dsn["user"], $dsn["password"])

        );

        $pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

    } catch (\Throwable $th) {

        die("Cannot connect to the Database.");

    }

?>

<!-- tables.php -->

<?php

    require\_once("./database.php");

    $queries = [

        // users

        "create table if not exists users(

                id serial,

                email character varying not null,

                name character varying,

                password character varying,

                PRIMARY KEY (id)

        );"

    ];

    foreach($queries as $query) {

        try {

            query($query);

        } catch(Throwable $th) {

            die("Something went wrong while setting up the Database");

        }

    }

?>

<!-- sign-in.php -->

<?php require\_once("./\_init.php"); ?>

<?php

    $user = signedIn("./index.php");

    $messages = [];

    if(check($\_POST, "submit", "sign-in")) {

        $session = $\_POST["session"];

        keep($session, "email", "password");

        apply($session, fn($k, $v) => trim($v));

        $subtitles = [];

        $user = null;

        if(empty($session["email"])) array\_push($subtitles, "Email must not be empty.");

        else {

            $query = query("select id, password from users where email = ?", [$session["email"]]);

            if($query->rowCount() == 0)

                array\_push($subtitles, "Email does not exist.");

            else $user = $query->fetch();

        }

        if(empty($session["password"])) array\_push($subtitles, "Password must not be empty.");

        elseif($user) {

            if(!password\_verify($session["password"], $user["password"])) {

                array\_push($subtitles, "Password is incorrect.");

            }

        }

        if(!empty($subtitles)) array\_push($messages, [

            "type" => "negative",

            "title" => "Account Authentication Failed",

            "subtitles" => $subtitles

        ]);

        else {

            try {

                signIn($user);

                header("Location: ./index.php");

                // array\_push($messages, [

                //     "type" => "positive",

                //     "title" => "Account Authentication Succeeded",

                //     "subtitle" => "Your account has been registered. You may now try signing in this account."

                // ]);

            } catch (\Throwable $th) {

                echo var\_dump($th);

                array\_push($messages, [

                    "type" => "negative",

                    "title" => "Account Authentication Failed",

                    "subtitle" => "Something went wrong while authenticating your account."

                ]);

            }

        }

    }

?>

<?php require\_once("./\_start.php"); ?>

<a class="link-button" href="./sign-up.php"><< Sign Up</a>

<h1>Sign In</h1>

<em>Already have an account? Sign in and gain access!</em>

<br>

<?php render("./messages.php", ["messages" => $messages]) ?>

<form action="./sign-in.php" method="post">

    <div class="field">

        <label for="session\_email">

            Email <?php render("./hint.php", ["char" => "\*",

                "title" => "Required"]); ?>

        </label>

        <input type="email" name="session[email]" id="session\_email" required>

    </div>

    <div class="field">

        <label for="session\_password">

            Password <?php render("./hint.php", ["char" => "\*",

                "title" => "Required"]); ?>

        </label>

        <input type="password" name="session[password]" id="session\_password" required>

    </div>

    <div class="field">

        <button type="submit" name="submit" value="sign-in">Submit</button>

    </div>

</form>

<?php require\_once("./\_end.php"); ?>

<!-- sign-up.php -->

<?php require\_once("./\_init.php"); ?>

<?php

    $user = signedIn("./index.php");

    $messages = [];

    if(check($\_POST, "submit", "sign-up")) {

        $user = $\_POST["user"];

        keep($user, "email", "name", "password", "password\_confirmation");

        apply($user, fn($k, $v) => trim($v));

        $subtitles = [];

        if(empty($user["email"])) array\_push($subtitles, "Email must not be empty.");

        else {

            $query = query("select id from users where email = ?", [$user["email"]]);

            if($query->rowCount() > 0)

                array\_push($subtitles, "Email already exists.");

        }

        if(empty($user["name"])) array\_push($subtitles, "Name must not be empty.");

        if(empty($user["password"])) array\_push($subtitles, "Password must not be empty.");

        elseif(strlen($user["password"]) < 7) array\_push($subtitles, "Password must be 7 characters or more.");

        if(!empty($user["password\_confirmation"]) && !empty($user["password"])

            && strlen($user["password"]) >= 7

            && $user["password"] != $user["password\_confirmation"])

            array\_push($subtitles, "Password Confirmation must match Password.");

        if(!empty($subtitles)) array\_push($messages, [

            "type" => "negative",

            "title" => "Account Registration Failed",

            "subtitles" => $subtitles

        ]);

        else {

            try {

                pop($user, "password\_confirmation");

                $user["password"] = password\_hash($user["password"], null);

                $keys = array\_keys($user);

                $columns = join(", ", $keys);

                $values = join(", ", array\_map(fn($e) => ":{$e}", $keys));

                query("insert into users ({$columns}) values ({$values})", $user);

                array\_push($messages, [

                    "type" => "positive",

                    "title" => "Account Registration Succeeded",

                    "subtitle" => "Your account has been registered. You may now try signing in this account."

                ]);

            } catch (\Throwable $th) {

                echo var\_dump($th);

                array\_push($messages, [

                    "type" => "negative",

                    "title" => "Account Registration Failed",

                    "subtitle" => "Something went wrong while registering your account."

                ]);

            }

        }

    }

?>

<?php require\_once("./\_start.php"); ?>

<a class="link-button" href="./sign-in.php"><< Sign In</a>

<h1>Sign Up</h1>

<em>New User? Sign up for an account!</em>

<br>

<?php render("./messages.php", ["messages" => $messages]) ?>

<form action="./sign-up.php" method="post">

    <div class="field">

        <label for="user\_email">

            Email <?php render("./hint.php", ["char" => "\*",

                "title" => "Required and must be unique"]); ?>

        </label>

        <input type="email" name="user[email]" id="user\_email" required>

    </div>

    <div class="field">

        <label for="user\_name">

            Name <?php render("./hint.php", ["char" => "\*",

                "title" => "Required"]); ?>

        </label>

        <input type="text" name="user[name]" id="user\_name" required>

    </div>

    <div class="field">

        <label for="user\_password">

            Password <?php render("./hint.php", ["char" => "\*",

                "title" => "Required and must be 7 or more characters long"]); ?>

        </label>

        <input type="password" name="user[password]" id="user\_password" minlength="7" required>

    </div>

    <div class="field">

        <label for="user\_password\_confirmation">

            Password Confirmation

        </label>

        <input type="password" name="user[password\_confirmation]" id="user\_password\_confirmation" minlength="7">

    </div>

    <div class="field">

        <button type="submit" name="submit" value="sign-up">Submit</button>

    </div>

</form>

<?php require\_once("./\_end.php"); ?>

<!-- sign-out.php -->

<?php require\_once("./\_init.php"); ?>

<?php

    signOut();

    header("Location: ./sign-in.php");

?>

<!-- index.php -->

<?php require\_once("./\_init.php") ?>

<?php

    $user = signedIn(null, "./sign-in.php");

?>

<?php require\_once("./\_start.php"); ?>

<a class="link-button" href="#sign-out" onclick="confirm('Are you sure you want to sign out?') ? window.location='./sign-out.php' : null"><< Sign Out</a>

<a class="link-button" href="./account.php">Account</a>

<h1>Welcome <?= $user["name"] ?>!!!</h1>

<em>Hello and Good Morning!</em>

<br>

<?php require\_once("./\_end.php"); ?>

<!-- account.php -->

<?php require\_once("./\_init.php") ?>

<?php

    $user = signedIn(null, "./sign-in.php");

    $messages = [];

    $redirect = null;

    if(check($\_POST, "submit", "account")) {

        $user\_ = $\_POST["user"];

        keep($user\_, "name", "new\_password", "new\_password\_confirmation", "password");

        apply($user\_, fn($k, $v) => trim($v));

        $subtitles = [];

        if(empty($user\_["password"])) array\_push($subtitles, "Password must not be empty.");

        elseif($user) {

            if(!password\_verify($user\_["password"], $user["password"])) {

                array\_push($subtitles, "Password is incorrect.");

            }

        }

        if(empty($subtitles)) {

            if(!empty($user\_["new\_password"]) && strlen($user\_["new\_password"]) < 7)

                array\_push($subtitles, "Password must be 7 characters or more.");

            if(!empty($user\_["new\_password\_confirmation"])

                && !empty($user\_["new\_password"])

                && strlen($user\_["new\_password"]) >= 7

                && $user\_["new\_password"] != $user\_["new\_password\_confirmation"])

                array\_push($subtitles, "New Password Confirmation must match New Password.");

        }

        if(!empty($subtitles)) array\_push($messages, [

            "type" => "negative",

            "title" => "Account Update Failed",

            "subtitles" => $subtitles

        ]);

        else {

            try {

                $name = pop($user\_, "name");

                $password = pop($user\_, "new\_password");

                pop($user\_, "new\_password\_confirmation");

                pop($user\_, "password");

                if(!empty($name) && $name != $user["name"])

                    $user\_["name"] = $name;

                if(!empty($password) && !password\_verify($password, $user["password"]))

                    $user\_["password"] = password\_hash($password, null);

                if(!empty($user\_)) {

                    $columns = join(", ", array\_map(fn($e) => "{$e} = :{$e}", array\_keys($user\_)));

                    $user\_["email"] = $user["email"];

                    query("update users set {$columns} where email = :email", $user\_);

                    array\_push($messages, [

                        "type" => "positive",

                        "title" => "Account Update Succeeded",

                        "subtitle" => "Your account information has been updated."

                    ]);

                    $user = signedIn();

                    if(isset($user\_["password"])) {

                        $messages[count($messages) - 1]["subtitle"] .= " For security purposes, you need to sign in your account again.";

                        $redirect = "./sign-out.php";

                        signOut();

                    }

                } else {

                    array\_push($messages, [

                        "type" => "neutral",

                        "title" => "Account Update Ignored",

                        "subtitle" => "No changes were made."

                    ]);

                }

            } catch (\Throwable $th) {

                echo var\_dump($th);

                array\_push($messages, [

                    "type" => "negative",

                    "title" => "Account Update Failed",

                    "subtitle" => "Something went wrong while updating your account."

                ]);

            }

        }

    }

?>

<?php require\_once("./\_start.php"); ?>

<a class="link-button" href="./index.php"><< Home</a>

<h1>Account</h1>

<em>Change and update your account information.</em>

<br>

<?php render("./messages.php", ["messages" => $messages]) ?>

<form action="./account.php" method="post">

    <div class="field">

        <label for="user\_email">

            Email

        </label>

        <input type="email" id="user\_email" value="<?= $user["email"] ?>" readonly disabled>

    </div>

    <div class="field">

        <label for="user\_name">

            Name <?php render("./hint.php", ["char" => "?",

                "title" => "Must not be empty"]); ?>

        </label>

        <input type="text" name="user[name]" value="<?= $user["name"] ?>" id="user\_name">

    </div>

    <div class="field">

        <label for="user\_new\_password">

            New Password <?php render("./hint.php", ["char" => "?",

                "title" => "Must be 7 or more characters long"]); ?>

        </label>

        <input type="password" name="user[new\_password]" id="user\_new\_password" minlength="7">

    </div>

    <div class="field">

        <label for="user\_new\_password\_confirmation">

            New Password Confirmation

        </label>

        <input type="password" name="user[new\_password\_confirmation]" id="user\_new\_password\_confirmation">

    </div>

    <br>

    <small>For added security, you need to enter your current password first before making any changes to your account.</small>

    <div class="field">

        <label for="user\_password">

            Password

        </label>

        <input type="password" name="user[password]" id="user\_password" required>

    </div>

    <div class="field">

        <button type="submit" name="submit" value="account">Update</button>

    </div>

</form>

<?php require\_once("./\_end.php"); ?>

<?php if($redirect != null) { ?>

    <script>

        setTimeout(function() {

            window.location="<?= $redirect ?>";

        }, 7000);

    </script>

<?php } ?>

<!-- hint.php -->

<?php

    $char ??= "?";

    $title ??= "...";

?>

<b class="hint" title="<?= $title ?>"><span><?= $char[0] ?></span></b>

<!-- messages.php -->

<?php

    require\_once("./utils.php");

    $messages ??= [];

?>

<?php

    foreach($messages as $message) {

        $type = $message["type"] ?? "positive";

        $title = $message["title"] ?? "";

        $subtitle = $message["subtitle"] ?? "";

        $subtitles = $message["subtitles"] ?? [];

        $subtitles = !is\_array($subtitles) ? [$subtitles] : $subtitles;

        if(empty($title) && empty($subtitles)) continue;

?>

    <div class="<?= stitch('message', $type) ?>">

        <?php if(!empty($title)) { ?>

            <strong><?= $title ?></strong>

        <?php } ?>

        <?php if(!empty($subtitle)) { ?>

            <span><?= $subtitle ?></span>

        <?php } ?>

        <?php if(!empty($subtitles)) { ?>

            <ul>

                <?php foreach($subtitles as $subtitle) { ?>

                    <li><?= $subtitle ?></li>

                <?php } ?>

            </ul>

        <?php } ?>

    </div>

<?php } ?>

/\* style.css \*/

@import "https://fonts.googleapis.com/css2?family=DM+Mono&family=DM+Sans&display=swap";

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

    /\* font-size: 11pt; \*/

    font-family: 'DM Sans', sans-serif;

}

#main {

    width: 75%;

    max-width: 640px;

    margin-top: 10em;

    margin-bottom: 10em;

    /\* border: 1px solid red; \*/

    margin-left: auto;

    margin-right: auto;

}

.hint {

    position: relative;

    display: inline-block;

    background: rgba(0, 0, 0, 0.5);

    /\* padding: 0.10em; \*/

    color: white;

    width: 1em;

    height: 1em;

    border-radius: 4px;

    font-weight: bold;

    cursor: pointer;

    text-align: center;

}

.hint:hover {

    background: black;

}

.hint > span {

    position: absolute;

    display: inline-block;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%) scale(95%);

}

.field {

    margin-bottom: 1em;

}

.field > label {

    position: relative;

    display: block;

    font-weight: bold;

}

.field > label > .hint {

    position: absolute;

    right: 0;

}

.field > input {

    display: block;

    width: 100%;

    padding: 0.50em;

    border: 1px solid gray;

    border-radius: 0.5em;

}

.field > button[type=submit] {

    background: black;

    display: block;

    color: white;

    border: none;

    /\* text-align: right; \*/

    margin-left: auto;

    padding: 0.5em 1em;

    cursor: pointer;

    font-size: 1.25em;

    border-radius: 0.5em;

    font-weight: bold;

    box-shadow: -0.15em 0.15em 0 rgba(0, 0, 0, 0.322);

}

.field > button[type=submit]:hover {

    background: rgba(0, 0, 0, 0.521);

}

br {

    height: 1em;

    display: block;

    content: "";

    margin-bottom: 1em;

}

.link-button {

    display: inline-block;

    padding: 0.25em 0.5em;

    background: rgba(0, 0, 0, 0.137);

    color: black !important;

    text-decoration: none !important;

    border-radius: 0.25em;

}

.link-button:hover {

    text-decoration: underline !important;

}

.message.positive {

    background: rgba(0, 128, 0, 0.11);

}

.message.negative {

    background: rgba(255, 0, 0, 0.11);

}

.message.neutral {

    background: rgba(255, 255, 0, 0.11);

}

.message {

    margin: 0 1em 1em 1em;

    background: rgba(128, 128, 128, 0.192);

    padding: 1em;

    border-radius: 0.25em;

}

.message > strong {

    font-size: 1.15em;

    display: block;

}

.message li {

    margin-left: 2em;

}