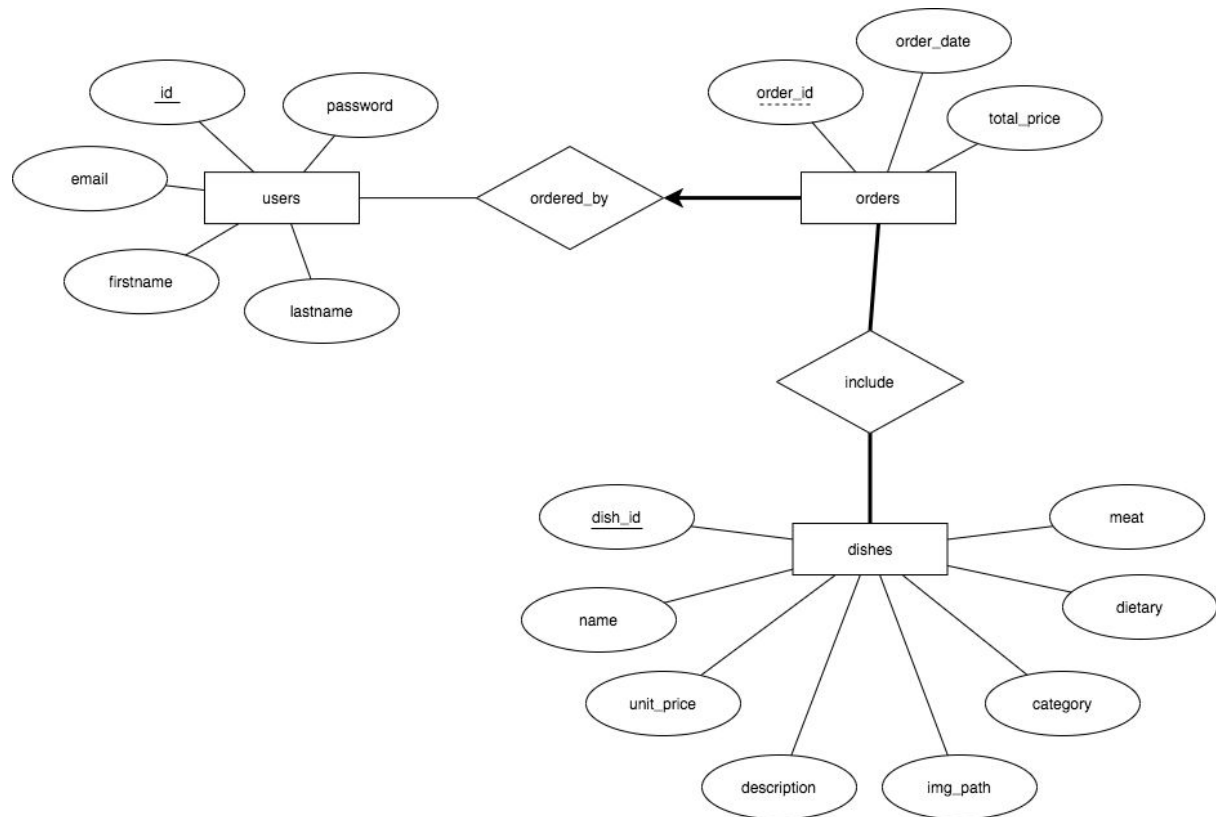


IAT 352 - PA2 Report

1. Database Design

ER Diagram:



Design Decisions:

Entities:

- Each user is identified by a unique ID (id), and also has an email, a password, first name and last name.
- Each order is identified by an order ID (order_id), and also has an order date, and total price.
- Each dish is identified by a unique ID (dish_id), and also has a name, a unit price, a brief description, a related image stored in the folder. Additionally, each dish belongs to a specific category, fits in certain dietary options, and includes one kind of meat.

Relations:

- One or more orders can be ordered by the same user, but different users won't order the same dishes at the same time.
- Each order must be ordered by at least one user.

- If a user account is deleted, all of its past orders will be removed as well.
- Each order must include at least one dish.
- One dish could be ordered multiple times in different orders.

2. Database Connectivity Code

We write the database connectivity code in a separate PHP file called connect.php so that we are able to embed this modular into any required files.

```
connect.php
1  <?php
2
3  $dbhost = "localhost";
4  $dbuser = "root";
5  $dbpass = "";
6  $dbname = "dan_peng";
7
8  $db = mysqli_connect($dbhost, $dbuser, $dbpass, $dbname);
9
10 if(mysqli_connect_error()){
11     echo "Failed to connect to MySQL: " . mysqli_connect_error();
12 }
13
14 ?>
```

Then, we use *require_once* keyword to embed connect.php into changePassword.php and manageAccount.php since these two pages need to display or change some information stored in the database.

```
profile.php  x  manageAccount.php  •  changePassword.php  x

1  <?php
2      // include auth.php file on all secure pages
3      require_once("auth_sessionNotActiveCheck.php");
4
5      // connect to database
6      require_once('connect.php');
7      $update_errors = array();
8  ?>
9
10
11 <!DOCTYPE html>
12 <html>
13     <head>
```

```
profile.php  x  manageAccount.php  •  changePassword.php  x

1  <?php
2      // include auth.php file on all secure pages
3      require_once("auth_sessionNotActiveCheck.php");
4
5      // connect to database
6      require_once('connect.php');
7      $manage_errors = array();
8  ?>
9
10
11 <!DOCTYPE html>
12 <html>
13     <head>
```

3. Secure Authentication Handling

We write the secure authentication handling code in a separate PHP file called `auth_sessionNotActiveCheck.php` so that we can effectively reuse it in other files.

```
auth_sessionNotActiveCheck.php ×
1  <?php
2
3  // check if there is an active session
4  // if there is no active session --> user is not logged in
5  // redirect user to login page
6  session_start();
7  if(!isset($_SESSION["email"])) {
8      header("location: register.php");
9      exit();
10 }
11
12 ?>
13
```

Then, we use `require_once` keyword to embed `auth_sessionNotActiveCheck.php` into `profile.php`, `manageAccount.php`, and `changePassword.php` to make sure only members can access these pages. Otherwise, we will require them to log in or sign up first.

```
profile.php × manageAccount.php ● changePassword.php ×
1  <?php
2  // include auth.php file on all secure pages
3  require_once("auth_sessionNotActiveCheck.php");
4  ?>
5
6
7  <!DOCTYPE html>
8  <html>
9  <head>
```

```
profile.php × manageAccount.php ● changePassword.php ×
1  <?php
2  // include auth.php file on all secure pages
3  require_once("auth_sessionNotActiveCheck.php");
4
5  // connect to database
6  require_once('connect.php');
7  $update_errors = array();
8  ?>
9
10
11 <!DOCTYPE html>
12 <html>
13 <head>
```

```
profile.php × manageAccount.php ● changePassword.php ×
1  <?php
2  // include auth.php file on all secure pages
3  require_once("auth_sessionNotActiveCheck.php");
4
5  // connect to database
6  require_once('connect.php');
7  $manage_errors = array();
8  ?>
9
10 <!DOCTYPE html>
11 <html>
12 <head>
```

4. Visitors

Browse Page - This is one of the core functionality of this website which is to let visitors browse the menu and select the item they want to order. The menu items are saved in the database and they are all assigned with a category. When retrieving each menu category, it is displayed using an array in order to show all items belong to that category. For example, the function `get_starters()` is created to get all starters items from the “dishes” table and displaying the html tags along with the content such as image path, item name, description and price. Then the function `get_starters` is called to execute the code. The rest of the menu categories are done in the same method.

Sort by

Price Range

☒ All

☐ \$0-\$5

☐ \$5-\$10

☐ \$10-\$15

☐ \$15-\$20

☐ \$20+

Ratings

☒ Any

☐ ★★☆☆☆

☐ ★★★☆☆

☐ ★★★★☆

☐ ★★★★★

Dietary

☒ All


☐ Gluten-free

☐ Vegetarian

☐ No soy

☐ No peanut


Starters



Tomato Kimchi

Sweet and spicy tomato salad


4.99



Spicy Gyoza

Deep fried gyoza mixed with greens feat Korean style spicy sauce.


6.99



Edamame

Steamed and seasoned with salt

2.99




Takoyaki

Filled with diced octopus, tempura scraps, pickled ginger, and green onion.

7.99


Donburi



Chashu Don

Japanese style braised pork rice feat. hacho miso sauce and mustard mayo


14.99



Kimchi Don


Korean style pan fried kimchi and bacon.

13.99



Mushroom Don

Mushroom "Shiitake and



Salmon Don

Wild Salmon sashimi on top

3

```

<?php

function get_starters(){
    global $db;
    $ret = array();
    $sql = "SELECT * FROM dishes WHERE category='Starters'";
    $result = mysqli_query($db, $sql);

    while($ar = mysqli_fetch_assoc($result)){
        $ret[] = $ar;

        echo '
        <div id="menuItem" class="menu_item">

        <div class="item_image">
            
        </div>

        <div class="item_text">
            <div class="item_name"> '.$ar['name'].' </div>
            <div class="item_description text-body-sml"> '.$ar['description'].' </div>
            <div class="item_price"> '.$ar['unit_price'].' </div>
        </div>

        </div>';
    }
    return $ret;
}
$dishes = get_starters();

?>

```

5. Member Registration and Login

Visitors can register as a member by clicking the sign up button on the top right of the home page. Visitors will be guided to the registration page. When signing up, the information will be collected and stored in the database. First of all, the user's input variables are created and referenced to each of the columns in the user's table. The second part is to check email's format, password confirmation and whether the input fields are empty. The last part is to store the user's input into the database if there is no error found. After the account is created successfully, the user will be directed to the homepage.

Receive all inputs values from the form.

```

$firstname = mysqli_real_escape_string($db, $_POST['first_name']);
$lastname = mysqli_real_escape_string($db, $_POST['last_name']);
$email = mysqli_real_escape_string($db, $_POST['email']);
$password_1 = mysqli_real_escape_string($db, $_POST['password_1']);
$password_2 = mysqli_real_escape_string($db, $_POST['password_2']);

```

Store receives data to the database and assigns each value to global session variables.

```
if (count($errors) == 0) {
    $password = md5($password_1); //encrypt the password before saving in the database
    $query = "INSERT INTO users (firstname, lastname, email, password)
    VALUES('$firstname', '$lastname', '$email', '$password')";
    mysqli_query($db, $query);

    $_SESSION['firstname'] = $firstname;
    $_SESSION['lastname'] = $lastname;
    $_SESSION['email'] = $email;
    $_SESSION['password'] = $password;
    $_SESSION['id'] = mysqli_insert_id($db);
    // echo $_SESSION['id'];

    header('location: index.php');
}
```

Visitors can then login as a member. Here is the code to verify if the visitor is a member.

```
// LOGIN USER
if (isset($_POST['login_user'])) {
    $email = mysqli_real_escape_string($db, $_POST['email']);
    $password = mysqli_real_escape_string($db, $_POST['password']);

    // check if all inputs are not empty
    if (empty($email)) array_push($errors, "Email is required.");
    if (empty($password)) array_push($errors, "Password is required.");

    // check if the variable $email is a valid email address
    $check_email = filter_var($email, FILTER_VALIDATE_EMAIL);
    if ($check_email == FALSE) array_push($errors, "Please type in valid email address.");

    if (count($errors) == 0) {
        $password = md5($password);
        $query = "SELECT * FROM users WHERE email='$email' AND password='$password'";
        $results = mysqli_query($db, $query);

        if (mysqli_num_rows($results) == 1) {
            $_SESSION['email'] = $email;
            $_SESSION['password'] = $password;

            $results_array = mysqli_fetch_assoc($results);
            $_SESSION['firstname'] = $results_array['firstname'];
            $_SESSION['lastname'] = $results_array['lastname'];
            $_SESSION['id'] = $results_array['id'];

            header('location: index.php');
        } else {
            array_push($errors, "Sorry, your email or password is wrong.");
        }
    }
}
```