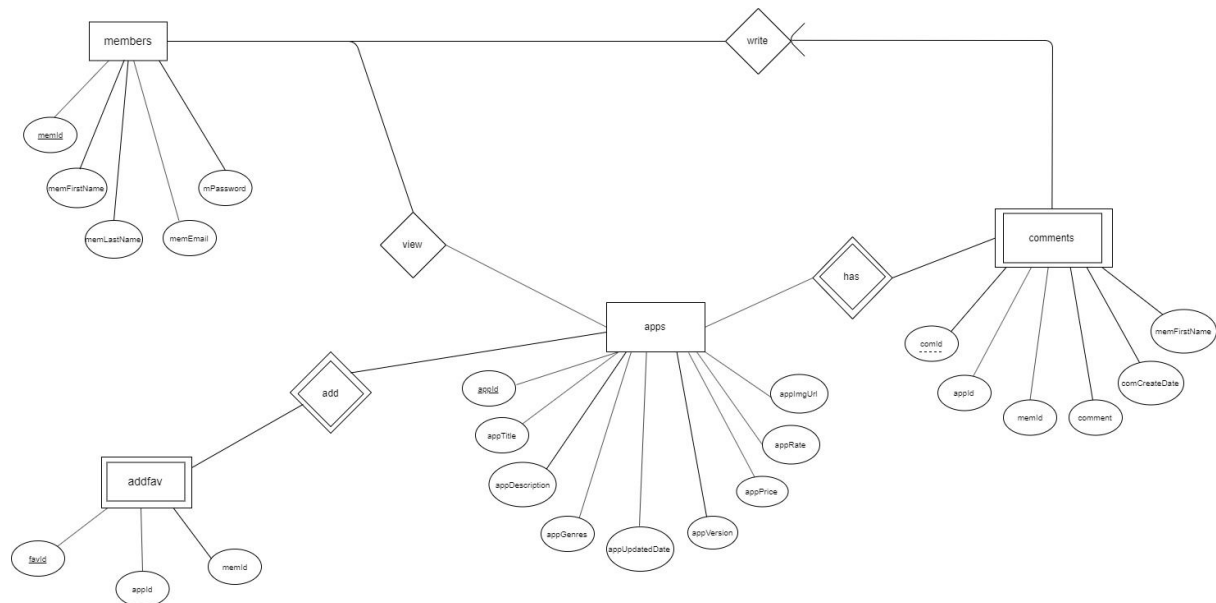


## IAT 352 Project Report

Part 1 - Database design - explain how you designed your database (what are the design decisions) + ER diagram:



Consider the function I need in this project, I created 4 tables that I need which include “members”, “apps”, “comments” & “addfav” .

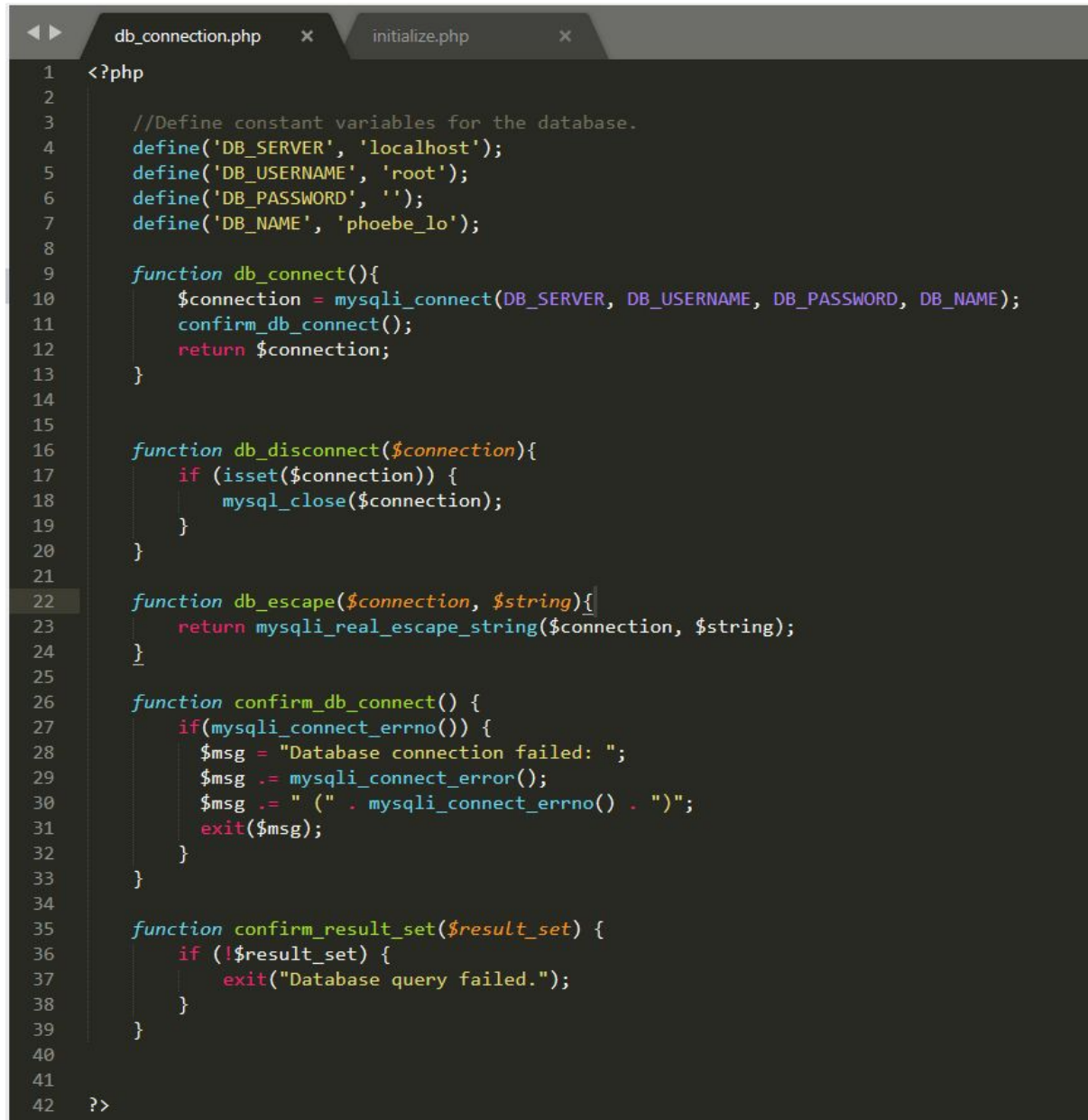
For “members” table, the attributes include “memId”, “memFirstName”, “memLastName”, “memEmail” and “mPassword”. “memId” is the primary key on this table. The reason I have those attributes because for member register they have to fill their information which include their first name, last name, email and password. However, there may be a chance that people have the same first and last name, so the email attribute is really important here because if there already have a same email registered, the email can’t be registered again. And they have to register by another email address.

For “apps” table, the attributes include “appId”, “appTitle”, “appDescription”, “appGenres”, “appUpdatedDate”, “appVersion”, “appRate”, “appPrice” and “appImgUrl”. “appId” is the primary key on this table. Consider the app details information that I want to show to the user, so I create those attributes to store the app details information. To display the information on the website, I used the “appId” to GET the data, because “appId” is a primary key which is unique and will not repeat.

For “comments” table, the attributes include “comId”, “appId”, “memId”, “comment”, “comCreateDate” and “memFirstName”. This table is weak entity, foreign key include “appId” and “memId”. The data of this table is related to the table of “members” and “apps”, if an “appId” or “memId” was deleted on the database the “comment” will be deleted too. So the comment only exists if the appId exists on the database. The reason I need those attributes because I need to know who(“memId”, “memFirstName” is used to display who create the comment on the website) create the comment(“comment”), which app(“appId”) they are commenting on and when(“comCreateDate”) they created the comment.

For “addfav” table, the attributes include “favId”, “appld” and “memId”. This table is weak entity, foreign key include “appld” and “memId”. The data of this table is related to the table of “members” and “apps”, if an “appld” or “memId” was deleted on the database the “favId” will be deleted too. This table is to store the app that have been bookmark by the member.

## Part 2 - Database connectivity code:



```
1 <?php
2
3 //Define constant variables for the database.
4 define('DB_SERVER', 'localhost');
5 define('DB_USERNAME', 'root');
6 define('DB_PASSWORD', '');
7 define('DB_NAME', 'phoebe_lo');
8
9 function db_connect(){
10     $connection = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD, DB_NAME);
11     confirm_db_connect();
12     return $connection;
13 }
14
15
16 function db_disconnect($connection){
17     if (isset($connection)) {
18         mysql_close($connection);
19     }
20 }
21
22 function db_escape($connection, $string){
23     return mysqli_real_escape_string($connection, $string);
24 }
25
26 function confirm_db_connect() {
27     if(mysqli_connect_errno()) {
28         $msg = "Database connection failed: ";
29         $msg .= mysqli_connect_error();
30         $msg .= " (" . mysqli_connect_errno() . ")";
31         exit($msg);
32     }
33 }
34
35 function confirm_result_set($result_set) {
36     if (!$result_set) {
37         exit("Database query failed.");
38     }
39 }
40
41
42 ?>
```

To connect the database I wrote the code inside db\_connection.php file. The reason I do that because in this project there have multiple files that I need to insert or select data from the database, this method I don't need to write the code for database connection every time.

```
//Define constant variables for the database.
define('DB_SERVER', 'localhost');
define('DB_USERNAME', 'root');
define('DB_PASSWORD', '');
define('DB_NAME', 'phoebe_lo');
```

I define constant variables for the database, which include DB\_SERVER, DB\_USERNAME, DB\_PASSWORD, DB\_NAME.

```
function db_connect(){
    $connection = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD, DB_NAME);
    confirm_db_connect();
    return $connection;
}
```

“\$connection” is the database connection resource variable. “mysqli\_connect(DB\_SERVER, DB\_USERNAME, DB\_PASSWORD, DB\_NAME)” is the function for php database connection.

confirm\_db\_connect();

Call a function, will discuss the details of this function in below.

```
function db_disconnect($connection){
    if (isset($connection)) {
        mysql_close($connection);
    }
}
```

Create a function of db\_disconnection to disconnect the connection. Pass \$connection in function as an argument, apply if statement in this, use isset function that is basically check that the value is set or not such as pass \$connect in them that means it check that the connection is set or not. mysqli\_close(\$connection) in this statement use mysqli\_close function it closes a previously.

```
function confirm_db_connect() {
    if(mysqli_connect_errno()) {
        $msg = "Database connection failed: ";
        $msg .= mysqli_connect_error();
        $msg .= " (" . mysqli_connect_errno() . ")";
        exit($msg);
    }
}
```

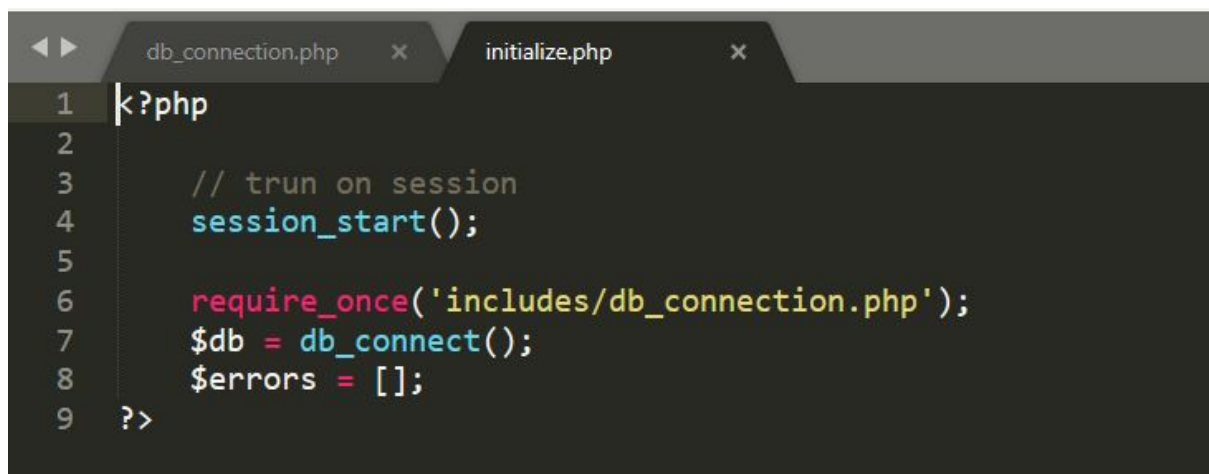
Create a function of confirm\_db\_connect() to confirm the database connection.

If(mysqli\_connect\_errno()) it returns the error code from the connection error, if found any.

\$msg = "Database Connection Failed: " use a variable named msg and pass a string in this variable is db connection failed. If found error then this msg is appear. \$msg .= mysqli\_connect\_error(); the function of error is call if there is any error found. \$msg .= "(" . mysqli\_connect\_errno() . ")"; a string that describes the error. Null if no error occur. exit(\$msg); this return the message and exit the script.

```
function confirm_result_set($result_set) {  
    if (!$result_set) {  
        exit("Database query failed.");  
    }  
}
```

Create a function of confirm\_result\_set and pass a variable of \$result\_set it is used to confirm Result set and pass a result set variable as a parameter. If(!\$result\_set) apply if statement and set a condition that if result is not set mean according to the requirement so it will exit statement Database query failed.



The screenshot shows a code editor with two tabs: 'db\_connection.php' and 'initialize.php'. The 'db\_connection.php' tab is active, displaying the following PHP code:

```
1 <?php  
2  
3 // trun on session  
4 session_start();  
5  
6 require_once('includes/db_connection.php');  
7 $db = db_connect();  
8 $errors = [];  
9 ?>
```

```
<?php  
include('includes/initialize.php');  
?>
```

I just have to include the function on the top of my code and call its function and use it. I include the function on my header.php, so it will run in all my file if the function call.

### Part 3 - Secure authentication handling:

For secure authentication handling, I implemented in session and password\_hash. To do that, I first need to start the PHP session.



```
db_connection.php x initialize.php x
1 k?php
2
3 // trun on session
4 session_start();
5
6 require_once('includes/db_connection.php');
7 $db = db_connect();
8 $errors = [];
9 ?>
```

```
<?php
    include('includes/initialize.php');
?>
```

I add the `session_start()`; on the `initialize.php` which is the same file that connect the database. And then include this file into my `header.php`. The reason to have `session_start()` on `header.php` because it allow to share information across the different pages of a single site. This lets the server know that all requests originate from the same user, thus allowing the site to display user-specific information and preferences.

Session and password\_hash have been implemented in `signup.php` and `login.php`.

```
if($_SERVER['REQUEST_METHOD'] == 'POST'){
    //set session value
    $firstName = $_POST['firstname'] ?? '';
    $lastName = $_POST['lastname'] ?? '';
    $email = $_POST['email'] ?? '';
    $password = $_POST['password'] ?? '';
    $hashed_password = password_hash($_POST['password'], PASSWORD_DEFAULT);
```

I used the PHP `password_hash()` function to create password hash from the password string entered by the user (line 30 on `signup.php`). This function creates a password hash using a strong one-way hashing algorithm. It also generates and applies a random salt automatically when hashing the password. This means that even if two users have the same passwords, their password hashes will be different.

```
while ($row = mysqli_fetch_assoc($result)) {
    if(password_verify($password, $row['memPassword'])) {
        $_SESSION['addEmail'] = $email;
        $db->close();
        header("Location: index.php");
    } else{
        echo "<script type='text/javascript'>alert('Please try again!')</script>";
    }
}
```

At the time of login, I will verify the given password with the password hash stored in the database using the PHP `password_verify()` function (line 48 on `login.php`). If user login

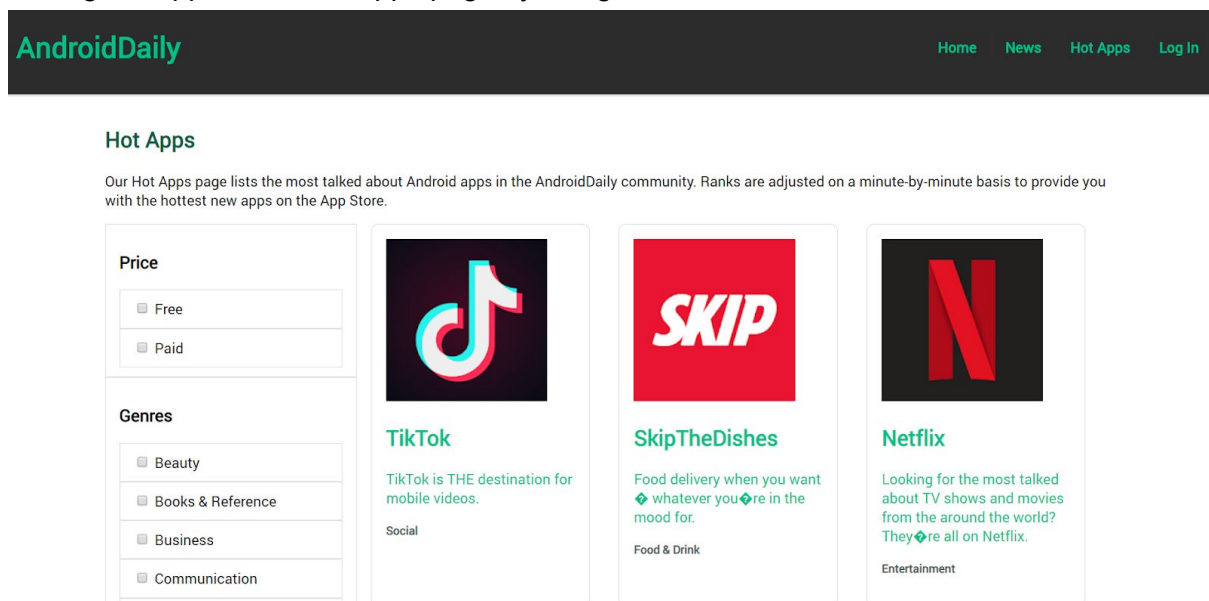
success, I used `$_SESSION` to store the email that entered by user. The reason I stored email in `$_SESSION` because email unique and there will not have the same email address exists in the database.

```
1  <?php
2
3      session_start();
4
5      //unset session value
6      unset($_SESSION['addEmail']);
7
8      header("Location: index.php");
9
10 ?>
```

When user logout, I used the `unset` function (line 6 on `logout.php`) to free all the session variable.

Part 4 - Visitors - explain what functionality your web app has for visitors and how this has been implemented:

Right now the functionality I have on my web app has for visitors are allowing the visitors filtering the apps on the `hotapps` page by using the checkbox filter.



To display the content on this page, first I need to create a sql statement `$select = "SELECT * FROM apps";` (line 156 on `hotapps.php`) that used to select all the column in the "apps" table. Then I create a container to display each app that I have on my database. On this page the app container will display the app information include the app image (column "appImgUrl" on "apps" table), title (column "appTitle" on "apps" table), description (column "appDescription" on "apps" table) and genres (column "appGenres" on "apps" table).

```
// display all apps
$appQuery = $select . $where;
// echo $appQuery;
$appResult = $db->query($appQuery);
if (!$appResult) {
    die("query fail");
}
while($row = mysqli_fetch_assoc($appResult)){
    echo"
    <div class=\"app-container\">
        <div class=\"app-container-list-item\">
            <a href=\"appdetails.php?id=\".$row['appId'].\"\">
                <div class=\"app-image-container\">
                    <img src=\".$row['appImgUrl'].\" alt=\".$row['appTitle'].\" class=\"appImage\">
                </div>
                <div class=\"app-container-list-item\">
                    <h2 class=\"app-title\">.$row['appTitle'].</h2>
                </div>
                <div class=\"app-container-list-item\">
                    <p class=\"app-container-details\">.$row['appDescription'].</p>
                </div>
                <div class=\"app-container-list-item\">
                    <h4 class=\"app-container-details genres-title\">.$row['appGenres'].</h4>
                </div>
            </a>
        </div>
    </div>
    ";
}
```

If visitor used filter to search the app, I need to add WHERE statement to the sql query therefore I generate the where string (line 159 on hotapps.php). When I create the “apps” table, the “appPrice” and “appGenres” are two different columns, so there will be a little different of two query.

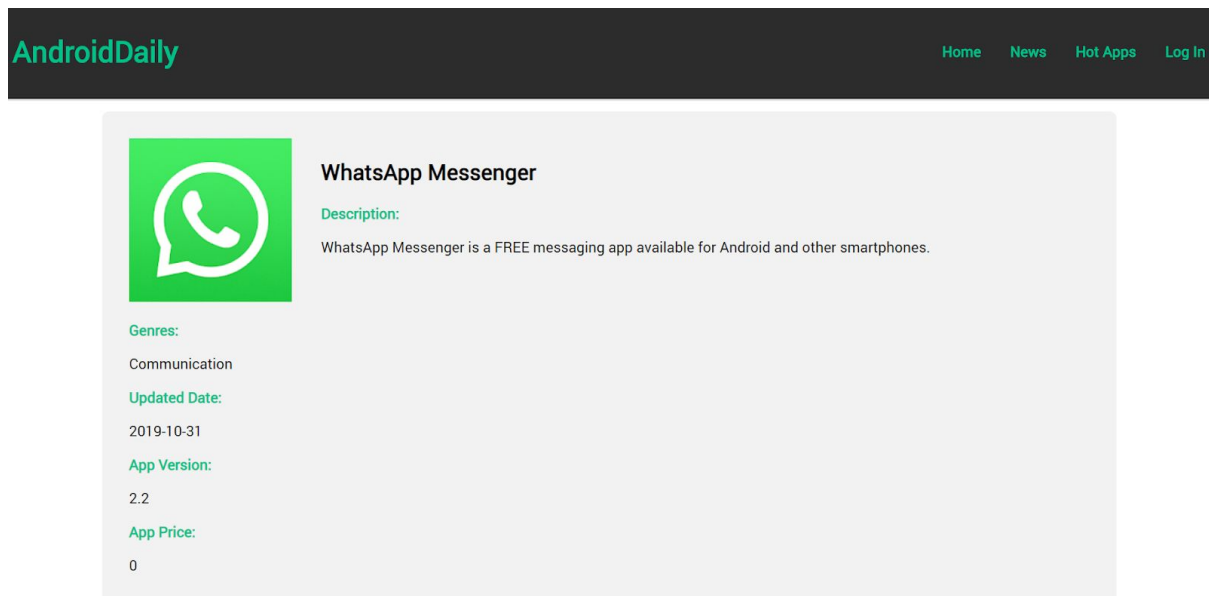
```
// check price - paid
if (isset($_POST['paid'])) {
    $where .= "appPrice > '0' AND ";
}

// check genres - beauty
if (isset($_POST['beauty'])) {
    $where .= "appGenres = 'Beauty' OR ";
}
```

```
$where = substr($where, 0, -4);
```

The difference between the two is the check price will have “AND” in the end, check genres will have “OR”, having the “AND” is make sure if the visitors want to search app by both price and genres, and the “OR” is allow visitors to search multiple genres. Because I add “AND” and “OR” into the \$where query, I need to use the php substr() function(line 265 on hotapps.php) to return part of the string. After the substr(), I add the \$select and \$where to the \$appQuery, and run the \$appResult, then the search result will run successful.

Visitors are allowed to click on any app on the hotapps page, after the click, it will navigate to appdetails page, this page will display the content that user click on the hotapps page.



```
<?php
$appInfoQuery = "SELECT * FROM apps WHERE appId=".$_GET['id'];
$appInfoResult = $db->query($appInfoQuery);

while($row = mysqli_fetch_assoc($appInfoResult)){
    // app information
    echo"
        <div class=\"apppost-container\">
            <div class=\"apppost-info\">
                <div class=\"app-post-list-item\">
                    <img src=\".$row['appImgUrl'].\" alt=\".$row['appTitle'].\" class=\"appImage\">
                </div>
                <div class=\"app-post-list-item\">
                    <p class=\"app-container-head\">Genres:</p>
                    <p class=\"app-container-details\">.$row['appGenres'].</p>
                </div>
                <div class=\"app-post-list-item\">
                    <p class=\"app-container-head\">Updated Date:</p>
                    <p class=\"app-container-details\">.$row['appUpdatedDate'].</p>
                </div>
                <div class=\"app-post-list-item\">
                    <p class=\"app-container-head\">App Version:</p>
                    <p class=\"app-container-details\">.$row['appVersion'].</p>
                </div>
                <div class=\"app-post-list-item\">
                    <p class=\"app-container-head\">App Price:</p>
                    <p class=\"app-container-details\">.$row['appPrice'].</p>
                </div>
            </div>
        </div>
    "
```

To know which app the user click on I use the `$appInfoQuery = "SELECT * FROM apps WHERE appId=".$_GET['id'];` (line 10 on appdetails.php), this query allow me to get the id and get the information (taken the information based on the appId on the database, display the information from the column on "appImgUrl", "appTitle", "appDescription", "appGenres", "appUpdatedDate", "appVersion" & "appPrice" on the "apps" table) by what the user click on. On this page visitors can view more detailed information about the app and allow them to view the comment that created by the members. If a non-register visitor try to create a comment after click on submit button, it will pop up a message to notice them to create an account to complete their comment.



**Reviews**

Tell others what you think about this app. Would you recommend it and why?

---

123      good

2019-11-12 08:49:02

localhost says

Please create an account to complete your comment!

```
// if no error
$userQuery = "SELECT * FROM members WHERE memEmail='".$$_SESSION['addEmail']."'";
// echo $userInfoQuery;
$userResult = mysqli_fetch_assoc($db->query($userQuery));
$userId = $userResult['memId'];
$username = $userResult['memFirstName'];
// $date = date('m/d/Y h:i:s a', time());
$date = date('Y-m-d H:i:s');
$insertCommentQuery = "INSERT INTO comments (appId, memId, comment, comCreateDate, memFirstName) VALUES (".
    $_GET['id'].", '$userId', '$comment', '$date', '$username')";
$insertCommentResult = $db->query($insertCommentQuery);
```

The comment function is available to the registered user. First the php code will check if the commenter is a member or not, if they are member `$userQuery = "SELECT * FROM members WHERE memEmail='".$$_SESSION['addEmail']."'";` will take the member information to `$userResult = mysqli_fetch_assoc($db->query($userQuery));` then set the `userId` and `userName` based on the result on the `userQuery`.

`$userId = $userResult['memId'];`

`$username = $userResult['memFirstName'];`

After getting the member information,

`$insertCommentQuery = "INSERT INTO comments (appId, memId, comment, comCreateDate, memFirstName) VALUES (". $$_GET['id'].", '$userId', '$comment', '$date', '$username')";`

`$insertCommentResult = $db->query($insertCommentQuery);`

`insertCommentQuery` will insert information into the columns ("appId", "memId", "comment", "comCreateDate" & "memFirstName" on the "comments" table).

```
$commentQuery = "SELECT * FROM comments WHERE appId=".$_GET['id'];
// echo $commentQuery;
$commentResult = $db->query($commentQuery);

while($row = mysqli_fetch_assoc($commentResult)){
    echo "<div class=\"display-comments-container\">
    <div class=\"commenter-container\">
    <p class=\"commenter\">".$row['memFirstName']."</p>
    <p class=\"comment-date\"><br>".$row['comCreateDate']."</p>
    </div>
    <div class=\"display-comments\">
    <p>".$row['comment']."</p>
    </div>
    </div>
    ";
};
```

After the comment stored on the database ("comments" table), `$commentQuery = "SELECT * FROM comments WHERE appId=".$_GET['id'];` will take the comment out and display based on the `appId`.

## Part 5 - Member registration and login:

As I mention the session and password\_hash I used on signup.php and login.php in part 3 - secure authentication handling. So in this part, I will discuss how a member sign up and insert those signup information into the database.

First, users need to sign up their information on the signup.php in order to create an account.

```
<div class="body-container">
  <!-- sign up form -->

  <div class="login">
    <h2 class="form-header">Sign up an account</h2>
    <!-- <form action="signup.php" method="post"> -->
    <form action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']);?>" method="post">
      <input type="text" name="firstname" placeholder="First name" id="firstname">
      <input type="text" name="lastname" placeholder="Last name" id="lastname">
      <input type="text" name="email" placeholder="Email" id="email">
      <input type="password" name="password" placeholder="Password" id="password">
      <p>Already have an account? <a href="login.php">Log in here</a>.</p>
      <input type="submit" value="Sign up">
    </form>
  </div>
</div>
```

This form allows the users fill in their information, if the user click on the submit button it will run the php code.

```
//if no error
if (count($errors) == 0) {
  //check account already exist or not
  $checkQuery = "SELECT * FROM members WHERE memEmail = '{$email}'";
  $result = $db->query($checkQuery);

  //if account has same credentials already exist
  if ($result->num_rows > 0) {
    //if exist
    $errors['acc'] = "The account has already been registered, please try to login or sign up with a different account";
    echo "<script type='text/javascript'>alert('The account has already been registered, please try to login or sign up with a different account')</script>";
  } else {
    //import to database
    $query = "INSERT INTO members (memFirstName, memLastName, memEmail, memPassword) VALUES ('$firstName', '$lastName', '$email', '$hashed_password')";

    echo $query;
    $result2 = $db->query($query);
    if ($result2) {
      $_SESSION['addEmail'] = $email;
    }
    $db->close();
    header("Location: index.php");
    //exit();
  }
}
```

On the php code will first check if there is any error on the fill in information, if yes the sign up will not success. As I mention before, there can't have the same email address on database, so I use "\$checkQuery = "SELECT \* FROM members WHERE memEmail = '{\$email}'";" to check if the email is already exist in the database or not, if yes it will pop up a message to let the user know this email already registered, please sign up by different email. If no error, then "\$query = "INSERT INTO members (memFirstName, memLastName, memEmail, memPassword) VALUES ('\$firstName', '\$lastName', '\$email', '\$hashed\_password')";"

'\$hashed\_password');" this query will insert the information entered by the user into the database by the column of "memFirstName", "memLastName", "memEmail" & "memPassword".

On the login page user need to enter the same email and password when the sign up.

```
$email = '';
$password = '';

if($_SERVER['REQUEST_METHOD'] == 'POST'){
    $email = $_POST['email'] ?? '';
    $password = $_POST['password'] ?? '';
    // $hashed_password = password_hash($_POST['password'], PASSWORD_DEFAULT);

    if (!empty($email) && has_string($email, '@')) {
        // $_SESSION['email'] = $email;
    } else {
        $errors['email'] = "Please enter a format email with abc@gmail.com";
    }

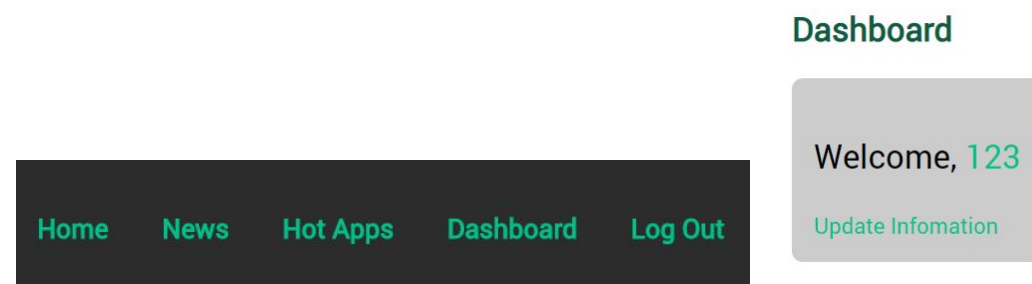
    if (empty($password)) {
        // $_SESSION['password'] = $password;
    } else {
        $errors['password'] = "Please fill in a password";
    }

    //if no error
    if (count($errors) == 0) {
        $query = "SELECT * FROM members WHERE memEmail = '{$email}'";
        $result = $db->query($query);

        if(!$result) {
            echo "query failed";
        }

        while ($row = mysqli_fetch_assoc($result)) {
            if(password_verify($password, $row['memPassword'])) {
                $_SESSION['addEmail'] = $email;
                $db->close();
                header("Location: index.php");
            } else{
                echo "<script type='text/javascript'>alert('Please try again!')</script>";
            }
        }
    }
}
```

After the user click on submit, the php code "\$query = "SELECT \* FROM members WHERE memEmail = '{\$email}'";" will select the email based on what user enters and the password\_verify is to check the enter password is it match to the password on the database, if the email and password was match with wil exists on the database, the login will be successful and the \$\_SESSION variable will set the user email. If there is any error, it will pop up a message "Please try again!" to the user.



If the login success, the navigation bar will have “Dashboard” and “Log Out”. On the dashboard, users can click on the “update information” it can allow the user to change their current password.

```
<?php
$password = '';

if($_SERVER['REQUEST_METHOD'] == 'POST'){
    $password = $_POST['password'] ?? '';
    $password = password_hash($password, PASSWORD_DEFAULT);

    if (empty($password)) {
        // $_SESSION['password'] = $password;
    } else {
        $errors['password'] = "Please fill in a password";
    }
}

//if no error
if (count($errors) == 0) {
    // $userInfoQuery = "SELECT * FROM members WHERE memEmail = '{$_SESSION['addEmail']}'";
    // $userInfoResult = $db->query($userInfoQuery);

    $updatePwQuery = "UPDATE members SET memPassword = '{$_password}' WHERE memEmail = '{$_SESSION['addEmail']}'";
    $userPwResult = $db->query($updatePwQuery);

    if(!$userPwResult) {
        echo "query failed";
    }

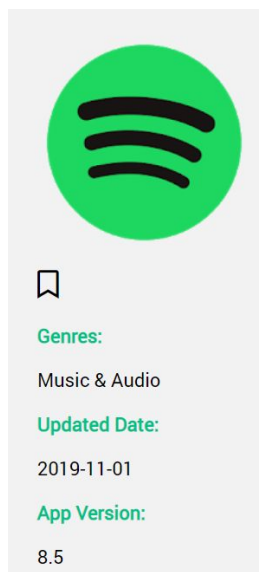
    // print_r($_SESSION);
    $db->close();
    // header("Location: index.php");
    echo "<script type='text/javascript'>alert('submitted successfully!')</script>";
}
```

\$updatePwQuery = "UPDATE members SET memPassword = '{\$\_password}' WHERE memEmail = '{\$\_SESSION['addEmail']}'"; This query will update the member password based on the variable on \$\_SESSION ['addEmail']. If the password update successfully, it will pop up a success message to alert the users.

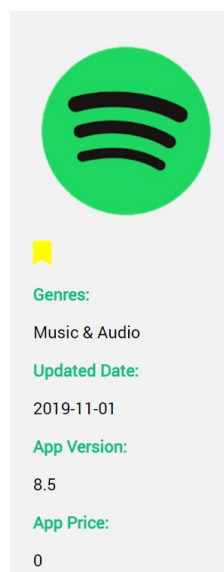
## Part 6 - Personalization

The personalization part on this project is a bookmark, the bookmark allow member bookmark an app, a bookmark item will be displayed on member dashboard, they are allowed to remove the bookmark item by clicking the remove icon.

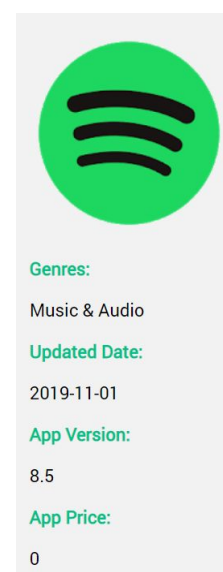
Member  
Item - before bookmark:



Member  
Item - after bookmark:



Non-member  
No bookmark icon





The bookmark function is for member only, if the member login successful, when they browse the app detail page, they can see there have a bookmark icon, it allow them to click the icon and bookmark the app. Above shown that the status change between the icon, and when a non-member visitor view on this page, they will not see this icon.

```
<div class="app-post-list-item">
  <div class="bookmark-icon">
    ";
    // if member logged in, bookmark will be display, member can bookmark the app
    if(isset($_SESSION['addEmail'])){
      $userQuery = "SELECT * FROM members WHERE memEmail='".$_SESSION['addEmail']."'";
      // echo $userInfoQuery;
      $userResult = mysqli_fetch_assoc($db->query($userQuery));
      $userId = $userResult['memId'];

      $appFavQuery = "SELECT * FROM addfav WHERE appId='".$_GET['id']."'AND "memId='{ $userId}'";
      $appFavResult = $db->query($appFavQuery);
      $rowcount = mysqli_num_rows($appFavResult);
      if ($rowcount > 0) {
        echo"<i class='fas fa-bookmark liked' id='".$row['appId']."' style='color:yellow;'></i>";
      } else{
        echo"<i class='far fa-bookmark addFav' id='".$row['appId']."'></i>";
      }
    }
  }
}
```

To change the bookmark icon status, first I need to check is the member is login or not, if yes, previous bookmark item will fill yellow color; if no, the bookmark color will not be filled. If the visitor is not a member, the bookmark will not be displayed. (appdetails.php)

```
$(".addFav").click(function(){
  // var fav = $(".addFav").is(':checked');

  // alert(this.id);
  var storeId = this.id;
  // alert(storeId);
  if($(this).hasClass("addFav")){
    $.ajax({
      method: "POST",
      url: "editfav.php",
      data:{'store_app_id': storeId}
    }).done(function(data){

      console.log(data);
      if (data) {
        $(".fa-bookmark").css("backgroundColor", "yellow");
        $(".fa-bookmark").removeClass("addFav");
        $(".fa-bookmark").addClass("liked");
      }
    })
  }
});

$(".bookmark-icon").on("click", ".liked", function() {
  alert("You already liked this app");
})
```

(appdetails.php)

The ajax part will change the bookmark color without refresh the page. Here, I first set a variable storeId which used to store the appId, then pass the variable to the editfav.php.

```

<?php
// include('includes/header.php');
include('includes/initialize.php');

// $favapp = $_POST['addfav'];
$storeAppId = $_POST['store_app_id'];

// add app to fav
$userQuery = "SELECT * FROM members WHERE memEmail='".$_SESSION['addEmail']."'";
$userResult = mysqli_fetch_assoc($db->query($userQuery));
$userId = $userResult['memId'];

$insert = '';

if(isset($_POST['store_app_id'])){
    $insert .= "INSERT INTO addfav (appId, memId) VALUES ('$storeAppId', '$userId')";
}

$insertfavResult = $db->query($insert);
echo $insertfavResult;

```


(editfav.php)

In here, I need to know who is the member, so I need to get the memId first. After I got the memId, will insert the bookmark app into the database by using the appId and memId. If the data insert to database successful, the bookmark will change color. The bookmark icon class will change from "addFav" to "liked", and when the member try to click the icon to add the item to bookmark, it will show a message ("You already liked this app") to the user. This can keep the data will not be repeated on the database.

## Dashboard

Welcome, **phoebe**


[Update Information](#)




**Netflix**

Looking for the most talked about TV shows and movies from the around the world? They're all on Netflix.

Entertainment






**Spotify**

With Spotify, you can find music and play millions of songs and podcasts for free.

Music & Audio



On the dashboard page, it will display the app that has been bookmark by member, in each app there have a remove icon, member can remove the app item by clicking the icon, the remove app will not show on the page, and the data will remove on the database.

```

<?php
$userQuery = "SELECT * FROM members WHERE memEmail='".$$_SESSION['addEmail']."'";
// echo $userInfoQuery;
$userResult = mysqli_fetch_assoc($db->query($userQuery));
$userId = $userResult['memId'];
$appFavQuery = "SELECT * FROM addfav WHERE memId='{$userId}'";
$appFavResult = $db->query($appFavQuery);

while($row = mysqli_fetch_assoc($appFavResult)){
    // display all apps
    $aQuery = "SELECT * FROM apps WHERE appId= " . $row['appId'];
    // echo $appQuery;
    $aResult = mysqli_fetch_assoc($db->query($aQuery));
    echo"
    <div class=\"app-container\">
        <div class=\"app-container-list-item\">
            <a href=\"appdetails.php?id=\" . $aResult['appId'] . "\">
                <div class=\"app-image-container\">
                    <img src=\" $aResult['appImgUrl'] .\" alt=\" $aResult['appTitle'] .\" class=\"appImage\">
                </div>
            </a>
            <div class=\"app-container-list-item\">
                <h2 class=\"app-title\">\" . $aResult['appTitle'] .\"</h2>
            </div>
            <div class=\"app-container-list-item\">
                <p class=\"app-container-details\">\" . $aResult['appDescription'] .\"</p>
            </div>
            <div class=\"app-container-list-item\">
                <h4 class=\"app-container-details genres-title\">\" . $aResult['appGenres'] .\"</h4>
            </div>
            <div class=\"app-container-list-item\">
                <i class=\"fas fa-trash-alt remove\" id=\"\" . $row['appId'] . \"\"></i>
            </div>
        </div>
    </div>
    ";
}

```

(dashboard.php)

Above is the code that read the data from database, and display the bookmark item.

```

<script type="text/javascript">
    $(".remove").click(function(e){
        var storeId = this.id;

        $.ajax({
            method: "POST",
            url: "showfav.php",
            data:{"store_app_id": storeId}
        }).done(function(data){
            // var result = $.parseJSON(data);
            // var string='';
            console.log(data);
            $(e.target).parent().parent().parent().remove();

            // $.each(result, function(key, value){
            // })
            // if(data){
            //     $(".app-container").remove();
            // }
        })
    })
</script>

```

(dashboard.php)

When the user click the remove icon, it will run the ajax part and it send the data to showfav.php. When receive the data back from showfav.php, it will remove the container that display the app.

```

<?php
// include('includes/header.php');
include('includes/initialize.php');

$storeAppId = $_POST['store_app_id'];

// get user info
$userQuery = "SELECT * FROM members WHERE memEmail='".$_SESSION['addEmail']."'";
$userResult = mysqli_fetch_assoc($db->query($userQuery));
$userId = $userResult['memId'];

// remove
// $removeQuery = "DELETE FROM addfav WHERE appId= ".$storeAppId. "AND memId=".$userId;
$removeQuery = "DELETE FROM addfav WHERE appId= '$storeAppId' AND memId='$userId'";
$removeResult = $db->query($removeQuery);
echo $removeResult;

```

(showfav.php)

When the showfav.php, receive the request from the dashboard.php, it will start the query to check who is the member and get their memId, than start the remove query to remove the item on the database based on the appld that the user selected and their memId. This data will be removed.

## Part 7 -AJAX

On this project, I used ajax in several parts, it includes comment, filter search result, add bookmark item, and remove bookmark item. The ajax part about add and remove bookmark item I already mentioned above, so here I will focus on the ajax part on the comment and filter search results.

### Comment:

```

<!-- comment display -->
<?php
$commentQuery = "SELECT * FROM comments WHERE appId=".$_GET['id'];
// echo $commentQuery;
$commentResult = $db->query($commentQuery);

while($row = mysqli_fetch_assoc($commentResult)){
    echo "<div class=\"display-comments-container\">
    <div class=\"commenter-container\">
    <p class=\"commenter\">".$row['memFirstName']."</p>
    <p class=\"comment-date\"><br>".$row['comCreateDate']."</p>
    </div>
    <div class=\"display-comments\">
    <p>".$row['comment']."</p>
    </div>
    </div>
    ";
}

```

(appdetails.php)

Above code is to display the comment has been posted, the query will select all the comment on the comments table by using the appId, it find any comment by this appId, the comments will be displayed.



```

<!-- ajax part -->
<script type="text/javascript">
    $(function(){
        // when user click the submit button, it will get the comment value and the appId
        $(".submit-button").on('click', function(){
            var comment = $(".usercomment").val();
            var storeId = $(".app-title").attr("id");

            // the value will pass to getcommon.php
            $.ajax({
                method: "POST",
                url: "getcommon.php",
                data: {"user_comment": comment, "store_app_id": storeId}
            }).done(function(data){
                console.log(data);
                // console.log("here");
                var result = $.parseJSON(data);

                var string='';

                $.each(result, function(key, value){

                    string += '<div class="display-comments-container"><div class="commenter-container"><p
                        class="commenter">'+value['memFirstName']+'</p><p class="comment-date"><br>'+value['comCreateDate']
                        +'</p></div><div class="display-comments"><p>'+value['comment']+'</p></div></div>';

                });

                // add result to the comment-section part
                $(".comment-section").append(string);

            })
        })
    })

```

(appdetails.php)

When the member after fill the comment and click the submit button it will run the ajax part, here I set 2 variable which is comment this will be used to get the value about what the comment is and the storeId is to get the appId. After getting the variable, then pass those data to the getcommon.php. After receive the data back from getcommon.php, by using the key and value to fill the data into the comment container. The append used to add the string into the comment-section part.

```

<?php
// include('includes/header.php');
include('includes/initialize.php');

$userComment = $_POST['user_comment'];
$storeAppId = $_POST['store_app_id'];

// add comment
$userQuery = "SELECT * FROM members WHERE memEmail='".$$_SESSION['addEmail']."'";
$userResult = mysqli_fetch_assoc($db->query($userQuery));
$userId = $userResult['memId'];
$userName = $userResult['memFirstName'];
$date = date('Y-m-d H:i:s');
$insertCommentQuery = "INSERT INTO comments (appId, memId, comment, comCreateDate, memFirstName) VALUES ('$storeAppId', '$userId', '$userComment', '$date', '$userName')";
$insertCommentResult = $db->query($insertCommentQuery);

// list comment
$result_array = array();
$commentsql = "SELECT * FROM comments WHERE appId='$storeAppId'";
$commentsqlResult = $db->query($commentsql);

if ($commentsqlResult->num_rows > 0) {
    while($row = $commentsqlResult->fetch_assoc()) {
        array_push($result_array, $row);
    }
}
echo json_encode($result_array);
?>

```

(getcommon.php)

When receive the request from appdetails.php, it will insert the comment to the database with the attribute of “appId”, “memId”, “comment”, “comCreateDate” and “memFirstName”. After insert the comment in the database, I run another query to select all the comment based on this appId. Then put those query result into array, then pass it back to the appdetails.php and display the comment based on this query result.

Filter search results:

### Price

☐ Free

☐ Paid


### Genres

☐ Beauty

☐ Books & Reference

☐ Business


☐ Communication



### TikTok

TikTok is THE destination for mobile videos.


Social



### SkipTheDishes

Food delivery when you want – whatever you're in the mood for.

Food & Drink



### Netflix

Looking for the most talked about TV shows and movies from the around the world? They're all on Netflix.

Entertainment

I apply another ajax in the filter part, in previous when user trying to filter the search result, after click the search button the page will be get refresh and display the search result. After apply ajax, the reload of search result on the page will not get refreshed.

```

<script type="text/javascript">
    $(function(){
        $("#filterSearchBtn").on('click', function(){
            var priceFree = $("#price-free").is(':checked');
            var pricePaid = $("#price-paid").is(':checked');
            var generBeauty = $("#gener-beauty").is(':checked');
            var generBooks = $("#gener-books").is(':checked');
            var generBusiness = $("#gener-business").is(':checked');
            var generCommunication = $("#gener-communication").is(':checked');
            var generEducation = $("#gener-education").is(':checked');
            var generEntertainment = $("#gener-entertainment").is(':checked');
            var generEvents = $("#gener-events").is(':checked');
            var generFinance = $("#gener-finance").is(':checked');
            var generFood = $("#gener-food").is(':checked');
            var generHealth = $("#gener-health").is(':checked');
            var generMusic = $("#gener-music").is(':checked');
            var generNews = $("#gener-news").is(':checked');
            var generPhotography = $("#gener-photography").is(':checked');
            var generShopping = $("#gener-shopping").is(':checked');
            var generSocial = $("#gener-social").is(':checked');
            var generTravel = $("#gener-travel").is(':checked');
            var generVideo = $("#gener-video").is(':checked');

            var apps = $(".app-search-container").empty();

            $.ajax({
                method: "POST",
                url: "getappsresult.php",
                data: {
                    "pFree": priceFree, "pPaid": pricePaid, "gBeauty": generBeauty, "gBooks": generBooks, "gBusiness":
                    generBusiness, "gCommunication": generCommunication, "gEducation": generEducation, "gEntertainment":
                    generEntertainment, "gEvents": generEvents, "gFinance": generFinance, "gFood": generFood, "gHealth":
                    generHealth, "gMusic": generMusic, "gNews": generNews, "gPhotography": generPhotography, "gShopping":
                    generShopping, "gSocial": generSocial, "gTravel": generTravel, "gVideo": generVideo
                }
            }).done(function(data){
                console.log(data);
                var result = $.parseJSON(data);

                var string='';

                $.each(result, function(key, value){

```

```

                    string += '<div class="app-container"><div class="app-container-list-item"><a href="appdetails.php?id='
                    +value['appId']+'><div class="app-image-container"><img src='+value['appImgUrl']+' alt='+value['
                    appTitle']+' class="appImage"></div><div class="app-container-list-item"><h2 class="app-title">'+
                    value['appTitle']+'</h2></a></div><div class="app-container-list-item"><p
                    class="app-container-details">'+value['appDescription']+'</p></div><div
                    class="app-container-list-item"><h4 class="app-container-details genres-title">'+value['appGenres']
                    +'</h4></div></div></div>';
                });

```

```

        $(".app-search-container").append(string);
    });

```

(hotapps.php)

On the ajax part, first I set the variable for each checkbox, if the checkbox was clicked then it will set checked. Then those variable will save inside the data and pass to the getappsresult.php. After receive the result back from the getappsresult.php, it set an empty string and input the result value in the html and display the search result to the user.

```

k?php
// include('includes/header.php');
include('includes/initialize.php');

$appFree = $_POST['pFree'];
$appPaid = $_POST['pPaid'];
$appBeauty = $_POST['gBeauty'];
$appBooks = $_POST['gBooks'];
$appBusiness = $_POST['gBusiness'];
$appCommunication = $_POST['gCommunication'];
$appEducation = $_POST['gEducation'];
$appEntertainment = $_POST['gEntertainment'];
$appEvents = $_POST['gEvents'];
$appFinance = $_POST['gFinance'];
$appFood = $_POST['gFood'];
$appHealth = $_POST['gHealth'];
$appMusic = $_POST['gMusic'];
$appNews = $_POST['gNews'];
$appPhotography = $_POST['gPhotography'];
$appShopping = $_POST['gShopping'];
$appSocial = $_POST['gSocial'];
$appTravel = $_POST['gTravel'];
$appVideo = $_POST['gVideo'];

$result_array = array();

// generate the select string
$select = "SELECT * FROM apps WHERE ";

//generate the where string
$where = "";

// check genres - Photography
if ($_POST['gPhotography'] == 'true') {
    $where .= "appGenres = 'Photography' OR ";
}

// check genres - Shopping
if ($_POST['gShopping'] == 'true') {
    $where .= "appGenres = 'Shopping' OR ";
}

// check genres - Social
if ($_POST['gSocial'] == 'true') {
    $where .= "appGenres = 'Social' OR ";
}

// check genres - Travel & Local
if ($_POST['gTravel'] == 'true') {
    $where .= "appGenres = 'Travel & Local' OR ";
}

// check genres - Video Players & Editors
if ($_POST['gVideo'] == 'true') {
    $where .= "appGenres = 'Video Players & Editors' OR ";
}

$where = substr($where, 0, -4);

$appQuery = $select . $where;

$appResult = $db->query($appQuery);

if ($appResult->num_rows > 0) {
    while($row = $appResult->fetch_assoc()) {
        array_push($result_array, $row);
    }
}

echo json_encode($result_array);

```

(getappresult.php)

When the getappresult.php receive a request from hotapps.php, it will first get the value of those checkbox, if the checkbox is checked that it will add into the query and search the result based on which checked was checked. After the query is done it will save it into an array and the array result will pass back to the hotapps.php.

## Part 8 - Learning Experience, Challenge and Personal Reflection

I think I learn a lot when working on this project, when building this web application, I faced a lot of challenges, such as on the PA2, I need to have a clear idea to build the tables on the database and the relationship between each entity. To insert the comment data into database, I need to know who is the creator, this part is a little challenging to myself, because I have to use the session to know which member login and get the information of that member on the database, then insert the new information into another entity, this part let me have clearer concept about the relationship of each entity in the database. On the PA3, the ajax part is hard for me, because it is about logic how the web server work, it takes time for me to understand how the ajax run, the structure inside the ajax, how the data sent back and forth. The bookmark part with using ajax take me a lot of time to make it work. Because the bookmark was shown on two part, I have to create two different ajax part from display and remove the bookmark item. Overall, I think this project is hard and I'm glad that I can solve those challenges which help me a lot of my problem solving, and the most important thing is I think I build a portfolio quality project.