

# Submission Worksheet

4.

**CLICK TO GRADE**

<https://learn.ethereallab.app/assignment/IT202-008-S2024/it202-api-project-milestone-2-2024/grade/rr42>

7.

8.

9.

10.

IT202-008-S2024 - [IT202] API Project Milestone 2 2024

## Submissions:

Submission Selection

1 Submission [active] 4/18/2024 9:40:30 PM

## Instructions

**▲ COLLAPSE ▲**

Implement the Milestone 2 features from the project's proposal

document: <https://docs.google.com/document/d/1XE96a8DQ52Vp49XACBDTNCq0xYDt3kF29cO88E>

Make sure you add your ucid/date as code comments where code changes are done

All code changes should reach the Milestone2 branch

Create a pull request from Milestone2 to dev and keep it open until you get the output PDF from this assignment.

Gather the evidence of feature completion based on the below tasks.

Once finished, get the output PDF and copy/move it to your repository folder on your local machine.

Run the necessary git add, commit, and push steps to move it to GitHub

Complete the pull request that was opened earlier

Create and merge a pull request from dev to prod

Upload the same output PDF to Canvas

**Branch name:** Milestone2

**Tasks:** 29 **Points:** 10.00

 Define Appropriate Tables for Data (1 pt.)

**▲ COLLAPSE ▲**

 Task #1 - Points: 1

**Text:** Screenshots of Table SQL

**Checklist**

\*The checkboxes are for your own tracking

\*

Points

Details

#	Points	Details
■ #1	1	Table(s) should have the 3 core columns we'll always be using (id, created, modified) plus additional columns for the incoming API data
■ #2	1	Columns should be logical and thought out (not valid to have a single field of JSON data or similar)
■ #3	1	Clearly caption screenshots

## Task Screenshots:

**Gallery Style: Large View**

---

Small      Medium      Large

	id	label	title	artist	image	lyrics	created	modified	is_api
> 1	2	logic-under-pressure	under pressure	logic	<a href="https://t2.genius.com/i/edited_lyrics">https://t2.genius.com/i/edited_lyrics</a>	2024-04-18 00:05:49	2024-04-18 20:34:33	0	
> 2	3	Drake-company-lyri	Company	Drake (ft. Travis Scott)	<a href="https://images.genius.com/Verse_1_Drake_I_got_it_right_lyrics.jpg">https://images.genius.com/Verse_1_Drake_I_got_it_right_lyrics.jpg</a>	2024-04-18 00:08:23	2024-04-18 02:51:32	1	
> 3	4	Logic-confessions-c	Confessions of a De	Logic	<a href="https://images.genius.com/Intro_Logic_Confessions_of_a_Death_Certified_Killah_lyrics.jpg">https://images.genius.com/Intro_Logic_Confessions_of_a_Death_Certified_Killah_lyrics.jpg</a>	2024-04-18 00:25:05	2024-04-18 00:25:00	1	
> 4	5	rahili-testersong-2-	testersong 2	rahili	<a href="https://imgur.com/Ox_aefhsyseyd_uiges_fulset">https://imgur.com/Ox_aefhsyseyd_uiges_fulset</a>	2024-04-18 00:26:53	2024-04-18 00:26:53	0	
> 5	6	Kanye-west-fade-ly	Fade	Kanye West (ft. Post Mi	<a href="https://images.genius.com/Intro_Rare_Earth_You_Fade_lyrics.jpg">https://images.genius.com/Intro_Rare_Earth_You_Fade_lyrics.jpg</a>	2024-04-18 00:25:26	2024-04-18 02:25:31	1	

SONGS table

## Checklist Items (3)

#1 Table(s) should have the 3 core columns we'll always be using (id, created, modified) plus additional columns for the incoming API data

#2 Columns should be logical and thought out (not valid to have a single field of JSON data or similar)

#3 Clearly caption screenshots

Task #2 - Points: 1

Text: Explain the design



## Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Note the different fields and their purpose
<input checked="" type="checkbox"/> #2	1	Note if you needed to normalize the data into separate tables or if one table fit your needs

### Response:

Only one table was necessary for my API data. My custom columns include label, title, artist, image, and lyrics. Label is the unique identifier for a specific song lyric page, and is always in the format " artist-title-'lyrics' ". Title and artist are self-explanatory. Image holds a URL to the image associated with the song, usually an album cover. And lyrics only holds the songs lyrics.

### Task #3 - Points: 1

Text: Add the pull request link for the branch related to this feature

#### Details:

Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature.

### URL #1

<https://github.com/rRahmanM6/rr42-it202-008/pull/63>

### Data Creation Page (2 pts.)

▲ COLLAPSE ▲

### Task #1 - Points: 1

Text: Screenshots of the creation page

## Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Show potentially valid data filled in for the custom creation page
<input checked="" type="checkbox"/> #2	1	Show how the API data is fetched for API data (must be server-side)
<input checked="" type="checkbox"/> #3	1	Show examples of validation messages
<input checked="" type="checkbox"/> #4	1	Show an example of successful creation message
<input checked="" type="checkbox"/> #5	1	Design/Style should be considered (i.e., bootstrap, custom css, etc)
<input checked="" type="checkbox"/> #6	1	Make sure the heroku dev url is visible in the address bar
<input checked="" type="checkbox"/> #7	1	Clearly caption screenshots

### Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

A screenshot of a web browser window titled "Create Song". The URL is "it202-rr42-dev-fb744d733bee.herokuapp.com/Project/admin/create.php". The page contains a form with the following fields:

- Title: test song
- Artist: rahi
- Image: randomurl.com
- Lyrics: bla bla bla bla

Below the form is a "Create Song" button.

Song creation page form filled out

#### Checklist Items (1)

- #1 Show potentially valid data filled in for the custom creation page

A screenshot of a web browser window titled "Create Song". The URL is "it202-rr42-dev-fb744d733bee.herokuapp.com/Project/admin/create.php". The page contains a form with the following fields:

- Title: test song
- Artist: (empty field)
- Image: (empty field) ! Please fill out this field.
- Lyrics: (empty field)

Below the form is a "Create Song" button.

empty artist field on creation page validation message

#### Checklist Items (1)

#3 Show examples of validation messages

The screenshot shows a web browser window titled "Create Song". The URL is "it202-rr42-dev-fb744d733bee.herokuapp.com/Project/admin/create.php". The page contains navigation links: Home, Profile, Search, Create Role, List Roles, Assign Roles, View Database, Create Song, and Logout. The main title is "Create Song". A success message "Song created successfully!" is displayed above a form. The form fields are: Title (input field), Artist (input field), Image (input field), and Lyrics (input field). Below the form is a green "Create Song" button.

successful creation message

#### Checklist Items (1)

#4 Show an example of successful creation message

Task #2 - Points: 1

Text: Screenshots of creation page code

#### Checklist

\*The checkboxes are for your own tracking

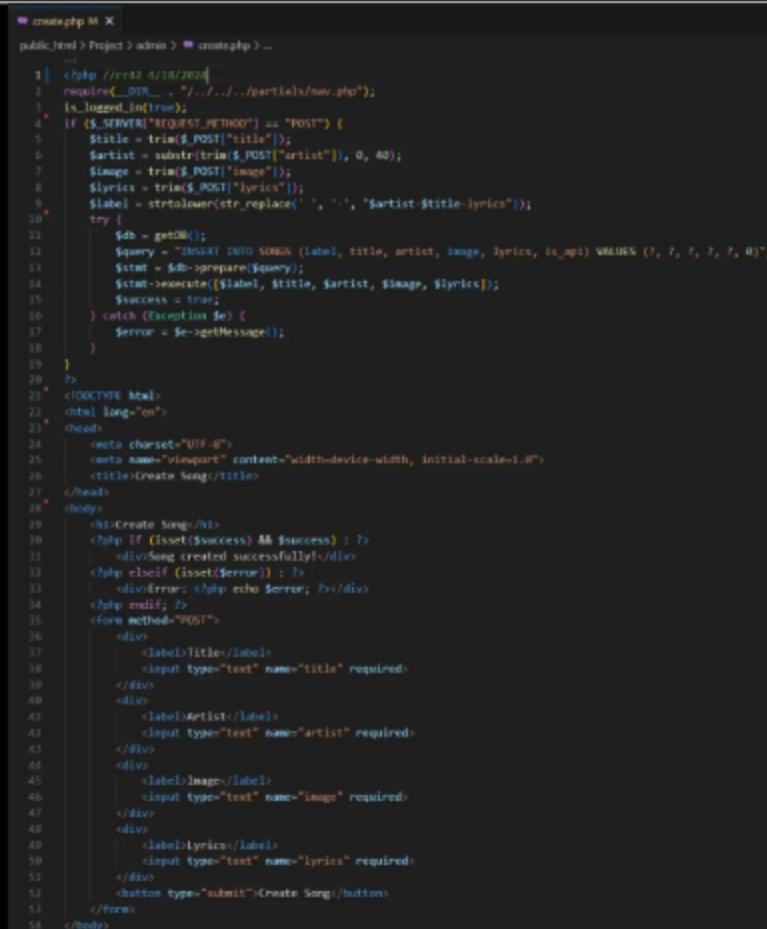
#	Points	Details
<input type="checkbox"/> #1	1	Form should have correct data types for each property being requested
<input type="checkbox"/> #2	1	Form should have correct validation for each field (HTML, JS, and PHP)
<input type="checkbox"/> #3	1	Successful creation should have a user-friendly message
<input type="checkbox"/> #4	1	Any errors should have user-friendly messages
<input type="checkbox"/> #5	1	Include the form/process for fetching API data

		include the form/process for returning API data
<input type="checkbox"/> #6	1	Include some indicator between custom data and API data
<input type="checkbox"/> #7	1	Include any other rules like role guards and login checks
<input checked="" type="checkbox"/> #8	1	Include ucid/date comments for each code screenshot
<input type="checkbox"/> #9	1	Clearly caption screenshots

## Task Screenshots:

### Gallery Style: Large View

Small      Medium      Large



```

create.php M X
public_html>Project>admin>create.php>...
...
1 | <?php //v42 4/18/2024
2 | require('../partials/nav.php');
3 | if (is_logged_in(true)) {
4 |     if ($_SERVER['REQUEST_METHOD'] == "POST") {
5 |         $title = trim($_POST['title']);
6 |         $artist = substr(trim($_POST['artist']), 0, 40);
7 |         $image = trim($_POST['image']);
8 |         $lyrics = trim($_POST['lyrics']);
9 |         $label = str_replace(str_replace(' ', '-', $artist.$title.$lyrics));
10 |         try {
11 |             $db = getDB();
12 |             $query = "INSERT INTO SONGS (label, title, artist, image, lyrics, is_api) VALUES (?, ?, ?, ?, ?, 0)";
13 |             $stmt = $db->prepare($query);
14 |             $stmt->execute([$label, $title, $artist, $image, $lyrics]);
15 |             $success = true;
16 |         } catch (Exception $e) {
17 |             $error = $e->getMessage();
18 |         }
19 |     }
20 | }
21 | <!DOCTYPE html>
22 | <html lang="en">
23 | <head>
24 |     <meta charset="UTF-8">
25 |     <meta name="viewport" content="width=device-width, initial-scale=1.0">
26 |     <title>Create Song</title>
27 | </head>
28 | <body>
29 |     <h1>Create Song</h1>
30 |     <?php if (isset($success) && $success) : ?>
31 |     <div>Song created successfully!</div>
32 |     <?php elseif (isset($error)) : ?>
33 |     <div>Error: <?php echo $error; ?></div>
34 |     <?php endif; ?>
35 |     <form method="POST">
36 |         <div>
37 |             <label>Title</label>
38 |             <input type="text" name="title" required>
39 |         </div>
40 |         <div>
41 |             <label>Artist</label>
42 |             <input type="text" name="artist" required>
43 |         </div>
44 |         <div>
45 |             <label>Image</label>
46 |             <input type="text" name="image" required>
47 |         </div>
48 |         <div>
49 |             <label>Lyrics</label>
50 |             <input type="text" name="lyrics" required>
51 |         </div>
52 |         <button type="submit">Create Song</button>
53 |     </form>
54 | </body>

```

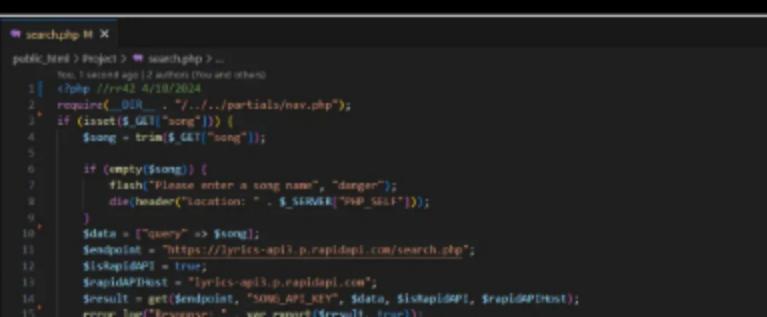
**create.php code**

## Checklist Items (3)

**#1 Form should have correct data types for each property being requested**

**#3 Successful creation should have a user-friendly message**

**#7 Include any other rules like role guards and login checks**



```

search.php M X
public_html>Project>search.php>...
...
1 | <?php //v42 4/18/2024
2 | require('../partials/nav.php');
3 | if (isset($_GET['song'])) {
4 |     $song = trim($_GET['song']);
5 |
6 |     if (empty($song)) {
7 |         flash("Please enter a song name", "danger");
8 |         die(header("Location: " . $_SERVER["PHP_SELF"]));
9 |     }
10 |     $data = ["query" => $song];
11 |     $endpoint = "https://lyrics-api3.p.rapidapi.com/search.php";
12 |     $skipAPI = true;
13 |     $rapidAPIHost = "lyrics-api3.p.rapidapi.com";
14 |     $result = get($endpoint, "SONG_API_KEY", $data, $skipAPI, $rapidAPIHost);
15 |     echo json_encode($result);

```

```

16     if ($e(result, "status", 409, false) == 200 && isset($result["response"])) {
17         $result = json_decode($result["response"], true);
18     } else {
19         $result = [];
20     }
21     if (!empty($result)) {
22         $db = getDB();
23         $firstSong = $result[0];
24         $songName = $firstSong["name"];
25         $artist = $firstSong["artist"];
26         $image = $firstSong["image"];
27         $songId = $firstSong["id"];
28
29         $stmt = $db->prepare("SELECT * FROM SONGS WHERE label = ?");
30         $stmt->execute([$songId]);
31         $song = $stmt->fetch();
32
33         if (!$song) {
34             $query = "INSERT INTO SONGS (label, title, artist, image, lyrics, is_api) VALUES (?, ?, ?, ?, ?, 1)";
35             $stmt = $db->prepare($query);
36             $stmt->execute([$songId, $songName, $artist, $image]);
37
38             $lyricsEndpoint = "https://lyrics-api.p.rapidapi.com/lyrics.php?id=$songId";
39             $lyricsResult = get($lyricsEndpoint, ["SONG_API_KEY", [], true, "Lyrics-api.p.rapidapi.com"]);
40
41             if ($e($lyricsResult, "status", 400, false) == 200 && isset($lyricsResult["response"])) {
42                 $lyricsData = json_decode($lyricsResult["response"], true);
43                 if (isset($lyricsData["lyrics"])) {
44                     $lyrics = $lyricsData["lyrics"];
45                     $query = "UPDATE SONGS SET lyrics = ? WHERE label = ?";
46                     $stmt = $db->prepare($query);
47                     $stmt->execute([$lyrics, $songId]);
48                 }
49             }
50         }
51     }
52 }
53
54 You, yesterday + added create page

```

## search.php code

### Checklist Items (4)

#2 Form should have correct validation for each field (HTML, JS, and PHP)

#4 Any errors should have user-friendly messages

#5 Include the form/process for fetching API data

#6 Include some indicator between custom data and API data

```

# search.php M X
public_html>Project> search.php > div.container-fluid > div.row
55 |   <div class="container-fluid"> <!--r62 4/18/2024-->
56 |     <h1>Song Lyrics</h1>
57 |     <form>
58 |       <div>
59 |         <label>Song</label>
60 |         <input name="song" />
61 |         <input type="submit" value="Fetch Song" />
62 |       </div>
63 |     </form>
64 |     <div class="row">
65 |       <php if (isset($result) && !empty($result)) : ?>
66 |       <php
67 |           $firstSong = $result[0];
68 |           $songName = $firstSong['name'];
69 |           $artist = $firstSong['artist'];
70 |           $image = $firstSong['image'];
71 |           $firstSongId = $firstSong['id'];
72 |           $db = getDB();
73 |           $stmt = $db->prepare("SELECT * FROM SONGS WHERE label = ?");
74 |           $stmt->execute([$firstSongId]);
75 |           $song = $stmt->fetch();
76 |
77 |           If ($song) {
78 |               $lyrics = $song['lyrics'];
79 |           } else {
80 |               $lyricsEndpoint = "https://lyrics-api.p.rapidapi.com/lyrics.php?id=$firstSongId";
81 |               $lyricsResult = get($lyricsEndpoint, ["SONG_API_KEY", [], true, "Lyrics-api.p.rapidapi.com"]);
82 |
83 |               If ($e($lyricsResult, "status", 400, false) == 200 && isset($lyricsResult["response"])) {
84 |                   $lyricsData = json_decode($lyricsResult["response"], true);
85 |                   If (isset($lyricsData["lyrics"])) {
86 |                       $lyrics = $lyricsData["lyrics"];
87 |                       $query = "INSERT INTO SONGS (label, title, artist, image, lyrics, is_api) VALUES (?, ?, ?, ?, ?, 1)";
88 |                       $stmt = $db->prepare($query);
89 |                       $stmt->execute([$firstSongId, $songName, $artist, $image, $lyrics]);
90 |                   } else {
91 |                       $lyrics = "Lyrics not available.";
92 |                   }
93 |               } else {
94 |                   $lyrics = "Failed to fetch lyrics.";
95 |               }
96 |           }
97 |           echo "<h2>$songName</h2>";
98 |           echo "<p>$artist</p>";
99 |           echo "<img src=\"$image\" alt=\"Song Image\" style='max-width: 400px; max-height: 400px;'>";
100 |           echo "<pre>$lyrics</pre>";
101 |           If (has_role("Admin")) {
102 |               $editSongId = $song["id"];
103 |               echo "<div class='admin-actions'>";
104 |               echo "<a href='edit.php?id=$editSongId' class='btn btn-primary'>Edit</a>";
105 |               echo "<form method='POST' class='delete-form'>";
106 |               echo "<input type='hidden' name='delete_id' value='$firstSongId'>";
107 |               echo "<button type='submit' class='btn btn-danger' onclick='return confirm(\"Are you sure you want to delete this song?\")'>Delete</button>";
108 |               echo "</form>";
109 |           }
110 |       }
111 |     }
112 |   </div>

```

## Checklist Items (0)

## Task #3 - Points: 1

Text: Screenshot of records from DB

## Checklist

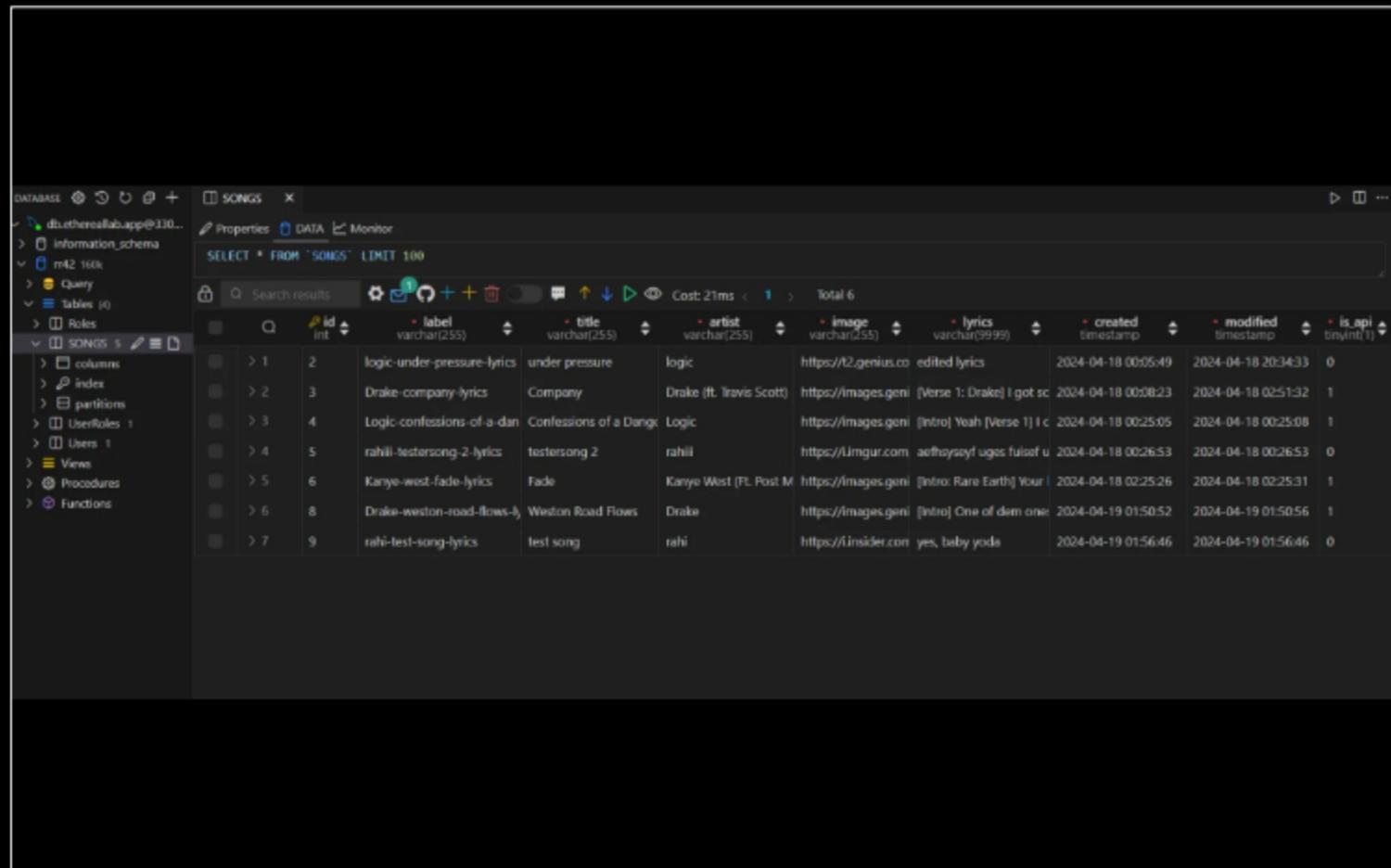
\*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Show at least one record fetched from the API
<input checked="" type="checkbox"/> #2	1	Show at least one record created via the creation form
<input checked="" type="checkbox"/> #3	1	Note what differs
<input checked="" type="checkbox"/> #4	1	Clearly caption screenshots

## Task Screenshots:

## Gallery Style: Large View

Small      Medium      Large



The screenshot shows a PostgreSQL database interface with the following details:

- Database:** dbtheroyalab.app@330..
- Table:** SONGS
- Query:** SELECT \* FROM "SONGS" LIMIT 100
- Results:** 100 rows displayed.

id	label	title	artist	image	lyrics	created	modified	is_api
1	logic-under-pressure-lyrics	under pressure	logic	https://t2.genius.co	edited lyrics	2024-04-18 00:05:49	2024-04-18 20:34:33	0
2	Drake-company-lyrics	Company	Drake (ft. Travis Scott)	https://Images.geni	[Verse 1: Drake] I got sc	2024-04-18 00:08:23	2024-04-18 02:51:32	1
3	Logic-confessions-of-a-dan	Confessions of a Dang	Logic	https://Images.geni	[Intro] Yeah [Verse 1] I c	2024-04-18 00:25:05	2024-04-18 00:25:08	1
4	rahili-testersong-2-lyrics	testersong 2	rahili	https://Imgur.com	aefhsyseyl uges fulsef u	2024-04-18 00:26:53	2024-04-18 00:26:53	0
5	Kanye-west-fade-lyrics	Fade	Kanye West (Fl Post M	https://Images.geni	[Intro: Rare Earth] Your	2024-04-18 02:25:26	2024-04-18 02:25:31	1
6	Drake-winston-road-flows-ly	Winston Road Flows	Drake	https://Images.geni	[Intro] One of dem one	2024-04-19 01:50:52	2024-04-19 01:50:56	1
7	rahili-test-song-lyrics	test song	rahili	https://Insider.com	yes, baby yoda	2024-04-19 01:56:46	2024-04-19 01:56:46	0

Records 2, 5, and 9 are both custom created using the create form. The is\_api field is set to 0 accordingly.

## Checklist Items (3)

#1 Show at least one record fetched from the API

#2 Show at least one record created via the creation form

#3 Note what differs

#### Task #4 - Points: 1

Text: Explain the process

##### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Provide a high-level step-by-step of how fetching API data works and gets added to your DB (include how duplicates are handled)
<input type="checkbox"/> #2	1	Provide a high-level step-by-step of how creating custom data works and gets added to your DB (include how duplicates are handled)
<input type="checkbox"/> #3	1	Briefly describe the validations for the applicable fields
<input type="checkbox"/> #4	1	Describe how duplicate data is handled

##### Response:

My application uses two endpoints, a Search Song endpoint that returns song info (id, title, artist, image), and a Get Song Lyrics endpoint that returns the lyrics to a specified song.

The user first submits a search query with the name of a song and its artist. Then, the song endpoint is invoked with the search query as a parameter. The API returns a number of songs in an array with the first item in the array being the most relevant song. The 'id' key of the first array object is then used to invoke the Get Song Lyrics endpoint, which returns the title and the lyrics in an array.

After invoking the first endpoint, my code checks if the 'id' exists in my DB, if so, the lyrics are pulled from the DB, otherwise, the second endpoint is called, and the new song is stored in the DB. This prevents duplicates from being added into the database.

Creation: The user fills out the form and the data is sent to the server-side with a POST request. Validation is in place to ensure no fields are left blank. The data is processed and a unique label is generated. The song is inserted into the database with an INSERT query.

#### Task #5 - Points: 1

Text: Add related links

##### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Include the heroku prod link for this page
<input type="checkbox"/> #2	1	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

URL #1

<https://it202-rr42-prod-dc703459f313.herokuapp.com/Project/admin/create.php>

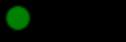
URL #2

<https://github.com/rRahmanM6/rr42-it202-008/pull/64>



Data List Page (many entities) (2 pts.)

[COLLAPSE](#)



Task #1 - Points: 1

[COLLAPSE](#)

Text: Screenshots of the list page

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Show the page of your entities listed (have a reasonable number shown)
<input checked="" type="checkbox"/> #2	1	Show the filter/sort form based on your data and the required limit field
<input type="checkbox"/> #3	1	Demonstrate a few varied filters/sorts
<input type="checkbox"/> #4	1	Demonstrate a filter that doesn't have any records (should show an appropriate message)
<input type="checkbox"/> #5	1	Each list item should have a link of single view (i.e., details), edit, and delete (some of which may only be visible to admin users)
<input type="checkbox"/> #6	1	Each list item should have a summary of the entity (likely won't be the entire entity data)
<input type="checkbox"/> #7	1	Design/Style should be considered (i.e., bootstrap, custom css, etc)
<input type="checkbox"/> #8	1	Make sure the heroku dev url is visible in the address bar
<input type="checkbox"/> #9	1	Clearly caption screenshots

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Title	Artist	Created	Actions
Company	Drake (ft. Travis Scott)	2024-04-18 00:08:23	<a href="#">Edit</a>   <a href="#">Delete</a>
Confessions of a Dangerous Mind	Logic	2024-04-18 00:25:05	<a href="#">Edit</a>   <a href="#">Delete</a>
Fade	Kanye West (Ft. Post Malone & Ty Dolla \$ign)	2024-04-18 02:25:26	<a href="#">Edit</a>   <a href="#">Delete</a>

<a href="#">test song</a>	rahi	2024-04-19 01:56:46	<a href="#">Edit</a>	<a href="#">Delete</a>
<a href="#">testersong_2</a>	rahiii	2024-04-18 00:26:53	<a href="#">Edit</a>	<a href="#">Delete</a>
<a href="#">under pressure</a>	logic	2024-04-18 00:05:49	<a href="#">Edit</a>	<a href="#">Delete</a>
<a href="#">Weston Road Flows</a>	Drake	2024-04-19 01:50:52	<a href="#">Edit</a>	<a href="#">Delete</a>

List page sorted alphabetically by title. Attempted to show 10000 record per page, but warning shown to user to ensure value must be between 0-100. Clicking artist or created will sort the list appropriately.

#### Checklist Items (4)

#1 Show the page of your entities listed (have a reasonable number shown)

#2 Show the filter/sort form based on your data and the required limit field

#3 Demonstrate a few varied filters/sorts

#6 Each list item should have a summary of the entity (likely won't be the entire entity data)

#### Task #2 - Points: 1

Text: Screenshots of the list page code

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show the filter/sort form generation
<input type="checkbox"/> #2	1	Show the DB query and how the filter/sort is handled (including the restriction on the limit field)
<input type="checkbox"/> #3	1	Show how the output is generated and displayed
<input type="checkbox"/> #4	1	Show any restrictions like role guard or login checks
<input type="checkbox"/> #5	1	Include ucid/date comments for each code screenshot
<input type="checkbox"/> #6	1	Clearly caption screenshots

Task Screenshots:

#### Gallery Style: Large View

Small      Medium      Large

```
list.php M X
public_html>Project>admin>list.php >...
You, 1 second ago | 2 authors (yehannamM and others)
1 <?php //ver2 4/18/2024 | You, 1 second ago + Uncommitted changes
2 require __DIR__ . "/../../../../../partials/nav.php";
3 if (!has_role("Admin")) {
4     Flash("You don't have permission to view this page", "warning");
5     die(header("Location: $BASE_PATH" . "/home.php"));
6 }
7 function deleteSong($id)
8 {
9     $db = getDB();
10     $stmt = $db->prepare("DELETE FROM SONGS WHERE id = ?");
```

```

11     $stmt->execute([$id]);
12 }
13 if (isset($_POST['delete_id'])) {
14     $id = $_POST['delete_id'];
15     deleteSong($id);
16     header("Location: $BASE_PATH" . "/list.php");
17     exit;
18 }
19 $perPage = 10;
20 if (isset($_GET['perPage']) && is_numeric($_GET['perPage'])) {
21     $perPage = max(0, min(100, $_GET['perPage']));
22 }
23 $orderBy = "title";
24 if (isset($_GET['sort'])) {
25     $sort = strtolower($_GET['sort']);
26     if (in_array($sort, ['title', 'artist', 'created'])) {
27         $orderBy = $sort;
28     }
29 }
30 $db = getDB();
31 $stmt = $db->prepare("SELECT id, title, artist, created FROM SONGS ORDER BY $orderBy LIMIT $perPage");
32 $stmt->execute();
33 $songs = $stmt->fetchAll(PDO::FETCH_ASSOC);
34 }
35 <!DOCTYPE html>
36 <html lang="en">
37 <head>
38     <meta charset="UTF-8">
39     <meta name="viewport" content="width=device-width, initial-scale=1.0">
40     <title>List Songs</title>
41 </head>
42 <body>
43     <h1>List Songs</h1>
44     <form method="GET">
45         <label for="perPage">Records per page:</label>
46         <input type="number" id="perPage" name="perPage" value="php echo $perPage; ?" min="0" max="100">
47         <button type="submit">Apply</button>
48     </form>
49     <table border="1">
50         <thead>
51             <tr>
52                 <th><a href="?sort=title">Title</a></th>
53                 <th><a href="?sort=artist">Artist</a></th>
54                 <th><a href="?sort=created">Created</a></th>
55                 <th>Actions</th>
56             </tr>
57         </thead>
58         <tbody>
59             <?php foreach ($songs as $song) : ?>
60                 <tr>
61                     <td><a href=".search.php?song=<?php echo urlencode($song['title']) . ' ' . $song['artist']; ?>"><?php echo $song['title']; ?></a></td>
62                     <td><?php echo $song['artist']; ?></td>
63                     <td><?php echo $song['created']; ?></td>
64                     <td>
65                         <a href="edit.php?id=<?php echo $song['id']; ?>">Edit</a> |
66                         <form method="POST" style="display: inline;">
67                             <input type="hidden" name="delete_id" value="<?php echo $song['id']; ?>">
68                             <button type="submit" onclick="return confirm('Are you sure you want to delete this song?')>Delete</button>
69                         </form>
70                     </td>
71                 </tr>
72             <?php endforeach; ?>
73         </tbody>
74     </table>
75 
```

## list.php

### Checklist Items (0)

```

54             <th><a href="?sort=artist">Artist</a></th> <!-- rr42 4/18/2024! -->
55             <th><a href="?sort=created">Created</a></th>
56             <th>Actions</th>
57         </tr>
58         <?php foreach ($songs as $song) : ?>
59             <tr>
60                 <td><a href=".search.php?song=<?php echo urlencode($song['title']) . ' ' . $song['artist']; ?>"><?php echo $song['title']; ?></a></td>
61                 <td><?php echo $song['artist']; ?></td>
62                 <td><?php echo $song['created']; ?></td>
63                 <td>
64                     <a href="edit.php?id=<?php echo $song['id']; ?>">Edit</a> |
65                     <form method="POST" style="display: inline;">
66                         <input type="hidden" name="delete_id" value="<?php echo $song['id']; ?>">
67                         <button type="submit" onclick="return confirm('Are you sure you want to delete this song?')>Delete</button>
68                     </form>
69                 </td>
70             </tr>
71         <?php endforeach; ?>
72     </tbody>
73 </table>
74 </body>
75 </html>

```

## list.php (continued)

### Checklist Items (0)

**Task #3 - Points: 1**

**Text: Explain how the page works**



### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Provide the high-level steps of how the filter/sorting works and how data is displayed
<input type="checkbox"/> #2	1	Summarize what you're showing on the screen
<input type="checkbox"/> #3	1	Mention your design/style choice
<input type="checkbox"/> #4	1	Mention which users can interact with the view, edit, and delete links

### Response:

The user can limit the number of records displayed by filling in the form field. The value is stored in a variable and used in the DB Select statement when listing records. The number entered must be between 0 and 100.

The user can also click on each column to sort alphabetically or by date.

The songs are displayed in an HTML table with columns title, artist, and creation date. There are also buttons to edit or delete each record, which can only be used by an admin. Clicking the edit button redirects to edit.php, and clicking delete shows a confirmation dialog, and deletes the record from the DB via a POST request, if the user confirms.

### Task #4 - Points: 1

Text: Add related links

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Include the heroku prod link for this page
<input type="checkbox"/> #2	1	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

### URL #1

<https://it202-rr42-prod-dc703459f313.herokuapp.com/Project/admin/list.php>

### URL #2

<https://github.com/rRahmanM6/rr42-it202-008/pull/65>

### View Details Page (single entity) (1 pt.)



### Task #1 - Points: 1

Text: Screenshots of the details page

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details

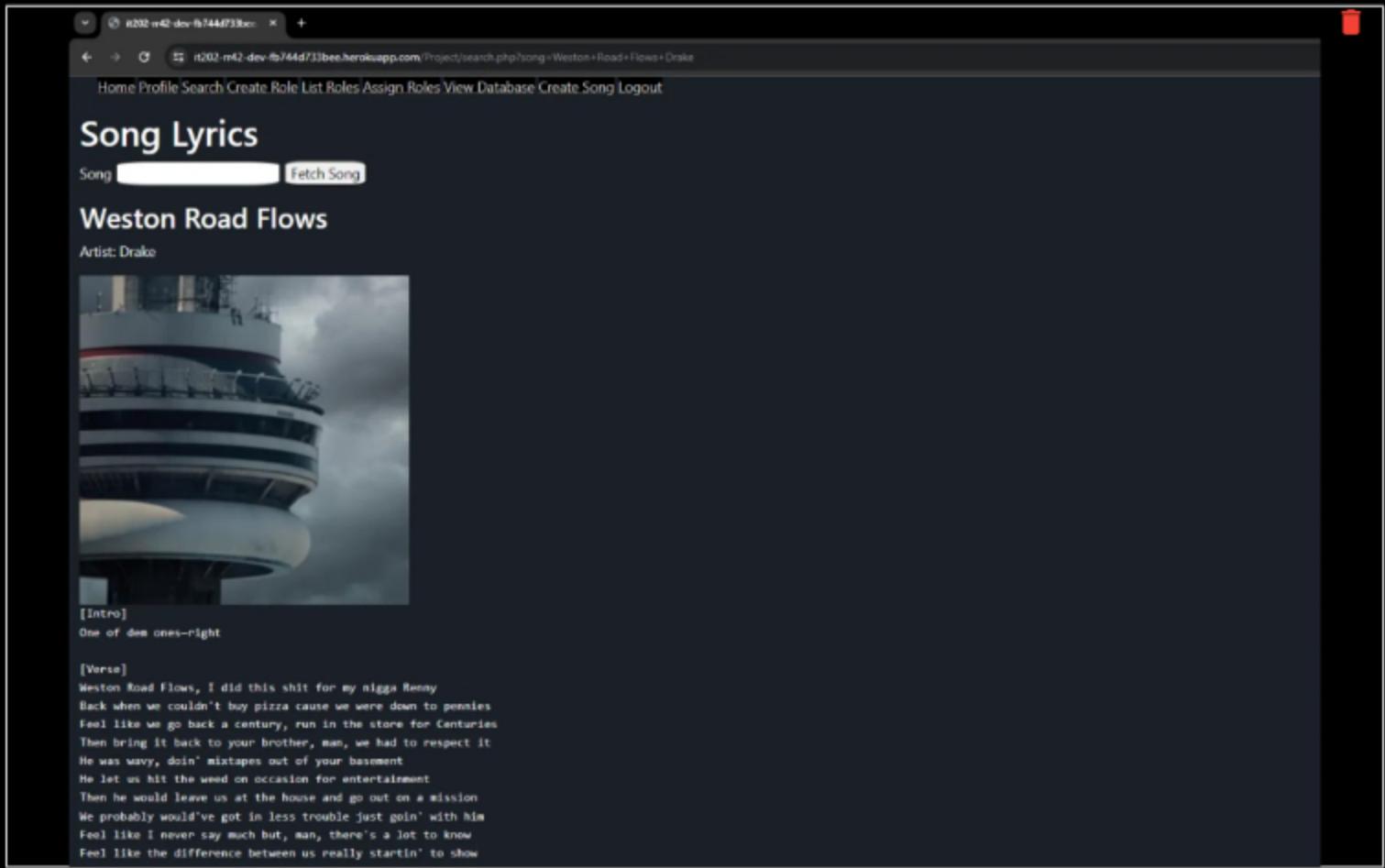
<input type="checkbox"/> #1	1	Entity should be fetch by id (via the url)
<input type="checkbox"/> #2	1	A missing id should redirect back to the list page with an applicable message
<input type="checkbox"/> #3	1	Design/Style should be considered (i.e., bootstrap, custom css, etc)
<input type="checkbox"/> #4	1	Data shown should be more detailed/inclusive than the summary view
<input type="checkbox"/> #5	1	There should be a link to edit the entity (this may be an admin-only thing, but it should be present for the respective role)
<input type="checkbox"/> #6	1	There should be a link to delete the entity (this may be an admin-only thing, but it should be present for the respective role)
<input type="checkbox"/> #7	1	Make sure the heroku dev url is visible in the address bar
<input type="checkbox"/> #8	1	Clearly caption screenshots

#### Task Screenshots:

**Gallery Style: Large View**

---

Small      Medium      Large



The screenshot shows a dark-themed web application interface. At the top, there's a navigation bar with links for Home, Profile, Search, Create Role, List Roles, Assign Roles, View Database, Create Song, and Logout. Below the navigation, the main content area has a title "Song Lyrics" and a subtitle "Song [redacted] Fetch Song". The main title of the song is "Weston Road Flows" and the artist is "Drake". There's a large, blurry image of a modern building. The lyrics are displayed in a monospaced font, starting with "[Intro]" and "[Verse]". The lyrics describe a past event where they couldn't buy pizza because they were down to pennies, and they bring it back to their brother. They mention being wavy and doing mixtapes. The [Verse] section continues with a narrative about a friend named Remy who was浪子 (wavy) and had mixtapes. The lyrics end with a reference to the difference between them and their friend.

Clicking on a title from the list page links the user to the relevant search page with the lyrics.

#### Checklist Items (3)

#4 Data shown should be more detailed/inclusive than the summary view

#5 There should be a link to edit the entity (this may be an admin-only thing, but it should be present for the respective role)

#6 There should be a link to delete the entity (this may be an admin-only thing, but it should be present for the respective role)

i202-m42-dev-b744d733be.herokuapp.com/Project/search.php?song=Weston+Road+Flows+Drake

```

Nowadays, they just shakin' my hand to hide the tension
A lot of people just hit me up when my name is mentioned
Shout out to KD, we relate, we get the same attention
It's rainin' money, Oklahoma City Thunder
The most successful rapper 35 and under
I'm assumin' everybody's 35 and under
That's when I plan to retire, man, it's already funded
Yeah, I brought your wifey out to Saint Martin
She violated, I sent her back where it all started
How quick they are to forget about their bachelor apartment
Leave it to niggas like you to show em' light in the darkness
Told my momma that I found a lady in the east
Concur, when I got signed, they upgraded the suite
Don Julio in the freezer that they gave us for free
I get you all you can eat, just have some patience with me
You wouldn't tell me you loved me, started seein' Monique
Last time I heard from Monique, T-Minus was makin' beats
I used to hit the corner store to get to Tahiti Treat
Now the talk at the corner store is I'm TME
The best ever, don't even question, you know better
But shit ain't always how it seems when it's sewn together
Yeah, I let that last line breathe, it take a second to get it
Weston Road Flows, my confidence level gettin' settled
Don't get hyped for the moment then start to backpedal
Don't let your newfound fame fool you
Or cloud up your judgement to talk loosely, I really do this
Been flyin' stupid since Vince Carter
Was on some through the legs arm in the hoop shit
Drinkin' Henny with Glenn Lewis, I been through it
Y'all was so afraid to lay claim to it
Too busy face screwin' on waste movements
You was ridin' TTC metro, I had the place boomin'
First take Drake, you know I rarely have to take two it
And they still take to it
Big Apple had the white Hummer parked right in front of Fluid
And we be walkin' in that bitch like we already knew it
But money can't buy happiness, Jellee talkin' truthful
But I'm happiest when I can buy what I want
Get high when I want

[Outro]
Yeah, that's right, yeah

```

[Edit](#)

[Delete](#)

Edit and delete buttons at the bottom of the page only visible to admins.

## Checklist Items (0)

### Task #2 - Points: 1

**Text:** Screenshots of the details page code

**Checklist** \*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show how id is fetched
<input type="checkbox"/> #2	1	Show the DB query to get the record
<input type="checkbox"/> #3	1	Show the code related to presenting the data and showing the links
<input type="checkbox"/> #4	1	Include uid/date comments for each code screenshot
<input type="checkbox"/> #5	1	Clearly caption screenshots

## Task Screenshots:

### Gallery Style: Large View

Small      Medium      Large

search.php M X

```

public: C:\Users\rahulg42\i202-008\public_html\Project\search.php - Modified
You've seconded ego | 1 author(s) (you and others)
1 | .php //rev42 6/18/2024   You've seconded ego + Uncommitted changes
2 | require __DIR__ . '/../../partials/nav.php';
3 | if (isset($_GET['song'])) {
4 |     $song = trim($_GET['song']);
5 |     if ($song) {

```

```
1 if (empty($song)) {
2     flash("Please enter a song name", "danger");
3     die(header("Location: " . $_SERVER["PHP_SELF"]));
4 }
5
6 $data = ["query" => $song];
7 $endpoint = "https://lyrics-api.p.rapidapi.com/search.php";
8 $isRapidAPI = true;
9 $rapidAPIHost = "lyrics-api.p.rapidapi.com";
10 $result = get($endpoint, "SONG_API_KEY", $data, $isRapidAPI, $rapidAPIHost);
11 error_log("Response: " . var_export($result, true));
12 if ($result["status"] == 400, false) == 200 && isset($result["response"]) {
13     $result = json_decode($result["response"], true);
14 } else {
15     $result = [];
16 }
17
18 if (empty($result)) {
19     $db = getDB();
20     $firstSong = $result[0];
21     $songName = $firstSong['name'];
22     $artist = $firstSong['artist'];
23     $image = $firstSong['image'];
24     $songId = $firstSong['id'];
25     $stmt = $db->prepare("SELECT * FROM SONGS WHERE label = ?");
26     $stmt->execute([$songId]);
27     $song = $stmt->fetch();
28     if (!$song) {
29         $query = "INSERT INTO SONGS (label, title, artist, image, lyrics, is_api) VALUES (?, ?, ?, ?, ?, ?, ?)";
30         $stmt = $db->prepare($query);
31         $stmt->execute([$songId, $songName, $artist, $image]);
32
33         $lyricsEndpoint = "https://lyrics-api.p.rapidapi.com/lyrics.php?id=$songId";
34         $lyricsResult = get($lyricsEndpoint, "SONG_API_KEY", [], true, "lyrics-api.p.rapidapi.com");
35
36         if ($lyricsResult["status"] == 400, false) == 200 && isset($lyricsResult["response"]) {
37             $lyricsData = json_decode($lyricsResult["response"], true);
38             if (isset($lyricsData['lyrics'])) {
39                 $lyrics = $lyricsData['lyrics'];
40                 $query = "UPDATE SONGS SET lyrics = ? WHERE label = ?";
41                 $stmt = $db->prepare($query);
42                 $stmt->execute([$lyrics, $songId]);
43             }
44         }
45     }
46 }
47
48 }
49
50 /*
```

search.php

## Checklist Items (0)

### **search.php (continued)**

## Checklist Items (0)

COLLAPSE

### TASK #3 - POINTS: 1

Text: Explain how the page works

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Provide the high-level steps for handling the DB lookup and presenting the data	

Response:

From the list page, clicking on the title of the song redirects to search.php with the song query filled in.



COLLAPSE

### Task #4 - Points: 1

Text: Add related links

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Include the heroku prod link for this page	
<input type="checkbox"/> #2	1	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature	

URL #1

<https://it202-rr42-prod-dc703459f313.herokuapp.com/Project/search.php>



COLLAPSE

Edit Data Page (2 pts.)



COLLAPSE

### Task #1 - Points: 1

Text: Screenshots of the edit page

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Show before and after screenshots of data you'll edit	
<input type="checkbox"/> #2	1	Show examples of validation messages	
<input type="checkbox"/> #3	1	Show an example of successful edit messages	
<input type="checkbox"/> #4	1	Design/Style should be considered (i.e., bootstrap, custom css, etc)	
<input type="checkbox"/> #5	1	Make sure the heroku dev url is visible in the address bar	
<input type="checkbox"/> #6	1	Clearly caption screenshots	

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Edit Song

Title

Artist Dr ! Please fill out this field.

Image <https://images.genius.com>

Lyrics

```
[Intro]
One of dem ones—
right

[Verse]
```

Update Song

edit page with empty title form.

#### Checklist Items (2)

#1 Show before and after screenshots of data you'll edit

#2 Show examples of validation messages

Song updated successfully

## Song Lyrics

Song  Fetch Song

### Weston Road Flows

Artist: Drake



lyrics edited using edit page

Edit

Delete

## after editing lyrics of song

### Checklist Items (2)

#1 Show before and after screenshots of data you'll edit

#3 Show an example of successful edit messages

### Task #2 - Points: 1

Text: Screenshots of edit page code

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Form should have correct data types for each property being requested
<input type="checkbox"/> #2	1	Form should have correct validation for each field (HTML, JS, and PHP)
<input type="checkbox"/> #3	1	Successful edit should have a user-friendly message
<input type="checkbox"/> #4	1	Any errors should have user-friendly messages
<input type="checkbox"/> #5	1	Include any other rules like role guards and login checks
<input type="checkbox"/> #6	1	Include ucid/date comments for each code screenshot
<input type="checkbox"/> #7	1	Clearly caption screenshots

#### Task Screenshots:

##### Gallery Style: Large View

Small

Medium

Large

```
edit.php
public_html > Project > admin > edit.php > html
...
1 |  <?php //rr42 4/19/2024
2 |  require(__DIR__ . "/../../../../partials/nav.php");
3 |  if (!has_role("Admin")) {
4 |      flash("You don't have permission to view this page", "warning");
5 |      die(header("Location: $BASE_PATH" . "/home.php"));
6 |  }
7 |  if (!isset($_GET['id']) || !is_numeric($_GET['id'])) {
8 |      flash("Invalid song ID", "danger");
9 |      die(header("Location: $BASE_PATH" . "/list.php"));
10 |
11 |  $songId = $_GET['id'];
12 |  $db = getDB();
13 |  $stmt = $db->prepare("SELECT * FROM SONGS WHERE id = ?");
14 |  $stmt->execute([$songId]);
15 |  $song = $stmt->fetch(PDO::FETCH_ASSOC);
16 |  if (!$song) {
17 |      flash("Song not found", "danger");
18 |      die(header("Location: $BASE_PATH" . "/list.php"));
19 |  }
20 |  if ($_SERVER["REQUEST_METHOD"] == "POST") {
```

```

21     $title = trim($_POST["title"]);
22     $artist = trim($_POST["artist"]);
23     $image = trim($_POST["image"]);
24     $lyrics = trim($_POST["lyrics"]);
25
26     $query = "UPDATE SONGS SET title = ?, artist = ?, image = ?, lyrics = ? WHERE id = ?";
27     $stmt = $db->prepare($query);
28     $stmt->execute([$title, $artist, $image, $lyrics, $songId]);
29
30     flash("Song updated successfully", "success");
31     header("Location: list.php");
32 }
33 ?>

```

## edit.php

### Checklist Items (0)

```

34 <!DOCTYPE html> <!--rr42 4/19/2024!-->
35 <html lang="en">
36
37   <head>
38     <meta charset="UTF-8">
39     <meta name="viewport" content="width=device-width, initial-scale=1.0">
40     <title>Edit Song</title>
41   </head>
42
43   <body>
44     <h1>Edit Song</h1>
45     <form method="POST">
46       <div>
47         <label>Title</label>
48         <input type="text" name="title" value=<?php echo htmlspecialchars($song['title']); ?>" required>
49       </div>
50       <div>
51         <label>Artist</label>
52         <input type="text" name="artist" value=<?php echo htmlspecialchars($song['artist']); ?>" required>
53       </div>
54       <div>
55         <label>Image</label>
56         <input type="text" name="image" value=<?php echo htmlspecialchars($song['image']); ?>" required>
57       </div>
58       <div>
59         <label>Lyrics</label>
60         <textarea name="lyrics" rows="5" required><?php echo htmlspecialchars($song['lyrics']); ?></textarea>
61       </div>
62       <button type="submit">Update Song</button>
63     </form>
64   </body>
65
66 </html>

```

## edit.php (continued)

### Checklist Items (0)

#### Task #3 - Points: 1

**Text:** Screenshot of records from DB

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Show a before and after screenshot of the record
<input checked="" type="checkbox"/> #2	1	Note what differs
<input checked="" type="checkbox"/> #3	1	Clearly caption screenshots

## Task Screenshots:

### Gallery Style: Large View

Small

Medium

Large

The screenshot shows a database interface with a sidebar containing database navigation and schema information. The main area displays a table named 'SONGS' with the following data:

	id	label	title	artist	image	lyrics	created
1	2	logic-under-pressure-lyrics	under pressure	logic	<a href="https://i2.genius.com/unsaf">https://i2.genius.com/unsaf</a>	edited lyrics	2024-04-18 00:05:49
2	3	Drake-company-lyrics	Company	Drake (ft. Travis Scott)	<a href="https://images.genius.com/l">https://images.genius.com/l</a>	[Verse 1: Drake] I got some shit for you to come and g	2024-04-18 00:08:23
3	4	Logic-confessions-of-a-dan	Confessions of a Dang	Logic	<a href="https://images.genius.com/l">https://images.genius.com/l</a>	[Intro] Yeah [Verse 1] I can't get no better, can't get no	2024-04-18 00:25:05
4	5	rashii-testersong-2-lyrics	testersong 2	rashii	<a href="https://imgur.com/Qx1lv6K">https://imgur.com/Qx1lv6K</a>	aefheysseyf uges fuisef usdf ygefgyu yis fy	2024-04-18 00:26:53
5	6	Kanye-west-fade-lyrics	Fade	Kanye West (Ft. Post M	<a href="https://images.genius.com/l">https://images.genius.com/l</a>	[Intro: Rare Earth] Your love is fadin' Your	2024-04-18 02:25:26
6	8	Drake-weston-road-flows-l	Weston Road Flows	Drake	<a href="https://images.genius.com/l">https://images.genius.com/l</a>	[Intro] One of dem ones—right [Verse] V	2024-04-19 01:50:52

### Before editing last record

#### Checklist Items (1)

#1 Show a before and after screenshot of the record

The screenshot shows the same database interface after editing the last record. The 'lyrics' column for the last row now contains the text 'lyrics edited using edit page'.

## After edit of last entry. Lyrics data different

### Checklist Items (2)

#1 Show a before and after screenshot of the record

#2 Note what differs

#### Task #4 - Points: 1

Text: Explain the process

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Provide a high-level step-by-step of how the fetching of the record, populating the form, and the update works and gets changed in your DB
<input type="checkbox"/> #2	1	Briefly describe the validations for the applicable fields

### Response:

The edit.php page is accessed using a specific song id. The corresponding song data is fetched from the DB and the information is used to fill in the edit form. When the user submits the form, the values in the form fields are used to create an SQL UPDATE query. No form field can be left blank or a validation message will appear on the empty form field.

#### Task #5 - Points: 1

Text: Add related links

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Include the heroku prod link for this page
<input type="checkbox"/> #2	1	Add the pull request link for the branch related to this feature Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

### URL #1

<https://it202-rr42-prod-dc703459f313.herokuapp.com/Project/admin/edit.php?id=8>

### URL #2

<https://github.com/rRahmanM6/rr42-it202-008/pull/66>

## Delete Handling (1 pt.)

[COLLAPSE](#)

### Task #1 - Points: 1

Text: Screenshots related to delete

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show the success message of a delete
<input type="checkbox"/> #2	1	Show any error messages of a failed delete (like id not being passed)
<input type="checkbox"/> #3	1	Show the code related to the delete processing

Task Screenshots:

#### Gallery Style: Large View

Small

Medium

Large

The screenshot shows a web browser window titled "List Songs". The URL is "it202-rr42-dev-fb744d733bee.herokuapp.com/Project/admin/list.php". The page displays a table of songs with columns "Title", "ArtistCreated", and "Actions". Two rows are visible: "Confessions of a Dangerous Mind" by Logic (2024-04-18 00:25:05) and "under pressure" by logic (2024-04-18 00:05:49). The "Actions" column for each row contains "Edit" and "Delete" links. A modal dialog box is overlaid on the page, centered over the second song row. The dialog has a dark background and contains the text "it202-rr42-dev-fb744d733bee.herokuapp.com says" and "Are you sure you want to delete this song?". It features two buttons: "OK" (highlighted in blue) and "Cancel". In the top right corner of the main content area, there is a small red trash can icon.

delete confirmation

Checklist Items (0)

This screenshot is identical to the one above it, showing the same "List Songs" page and the modal confirmation dialog for deleting the song "under pressure" by logic. The "OK" button is highlighted in blue. A small red trash can icon is also present in the top right corner of the main content area.

Home Profile Search Create Role List Roles Assign Roles View Database Create Song Logout

Song deleted successfully

## List Songs

Records per page: 10

Title	Artist Created	Actions
Confessions of a Dangerous Mind	Logic 2024-04-18 00:25:05	<a href="#">Edit</a>   <a href="#">Delete</a>

successful delete

### Checklist Items (1)

#1 Show the success message of a delete

```
list.php
```

public\_html > Project > admin > list.php > deleteSong

You, 3 minutes ago | 2 authors (rRahmanM6 and others)

```
1 <?php //rr42 4/18/2024
2 require(__DIR__ . "/../../partials/nav.php");
3 require(__DIR__ . "/../../partials/flash.php");
4
5 if (!has_role("Admin")) {
6     flash("You don't have permission to view this page", "warning");
7     die(header("Location: $BASE_PATH" . "/home.php"));
8 }
9 function deleteSong($id)
10 {
11     $db = getDB();
12     $stmt = $db->prepare("DELETE FROM SONGS WHERE id = ?"); rRah
13     $stmt->execute([$id]);
14     flash("Song deleted successfully", "success");
15 }
16
17 if (isset($_POST['delete_id'])) {
18     $id = $_POST['delete_id'];
19     deleteSong($id);
20     header("Location: list.php");
21     exit;
22 }
```

delete function in list.php

### Checklist Items (1)

#3 Show the code related to the delete processing

```

63 | <?php foreach ($songs as $song) : ?><!--rr42 4/18/2024!-->
64 |     <tr>
65 |         <td><a href="../search.php?song=<?php echo urlencode($song['title']) . ' ' . $song['artist']; ?>"><?php echo $song['title']; ?></a></td>
66 |         <td><?php echo $song['artist']; ?></td>
67 |         <td><?php echo $song['created']; ?></td>
68 |         <td>
69 |             <a href="edit.php?id=<?php echo $song['id']; ?>">Edit</a> |
70 |             <form method="POST" style="display: inline;">
71 |                 <input type="hidden" name="delete_id" value=<?php echo $song['id']; ?>">
72 |                 <button type="submit" onclick="return confirm('Are you sure you want to delete this song?')>Delete</button>
73 |             </form>
74 |         </td>
75 |     </tr>
76 | <?php endforeach; ?>
-->
```

### delete button in list.php

#### Checklist Items (1)

#3 Show the code related to the delete processing

#### Task #2 - Points: 1

**Text:** Screenshots of the data

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show a before and after screenshot of the DB data
<input type="checkbox"/> #2	1	Note what changed (i.e., record removed or soft delete value changed)
<input type="checkbox"/> #3	1	Clearly caption screenshots

#### Task Screenshots:

Gallery Style: Large View

Small      Medium      Large

	<input type="checkbox"/>	Q	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> Cost: 14ms	<input type="checkbox"/> 1	<input type="checkbox"/> Total 1
			id	label	title	artist	image	lyrics	created	modified			
			> 1	2	logic-under-pressure-lyrics	under pressure	logic	https://t2.genius.com/unsafe_edited_lyrics	2024-04-18 00:05:49	2024-04-18 20:34			
			> 2	4	Logic-confessions-of-a-dan	Confessions of a Dangerous Logic	Logic	https://images.genius.com/i/[Intro] Yeah [Verse 1] I can't	2024-04-18 00:25:05	2024-04-18 00:25			

before deleting last record

### Checklist Items (1)

#1 Show a before and after screenshot of the DB data

#### Task #3 - Points: 1

Text: Explain the delete logic

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Is it a soft or hard delete
<input type="checkbox"/> #2	1	Are there any necessary roles or restrictions? (can only delete their data, can only be done by admin, etc)
<input type="checkbox"/> #3	1	Provide the high-level steps for handling the DB lookup and handling the delete

#### Response:

A hard delete is done meaning the record is fully deleted from the database with no backups in place. Only administrators have the ability to delete a record.

An SQL DELETE statement is used to delete a song from the database. The delete statement deletes record where the id matches the song the user wishes to delete.

#### Task #4 - Points: 1

**Text:** Add the pull request link for the branch related to this feature

**Details:**

Note: the link should end with /pull/#. Same pull request shouldn't be used for each feature

**URL #1**

<https://github.com/rRahmanM6/rr42-it202-008/pull/66>

Misc (1 pt.)

[COLLAPSE](#)

**Task #1 - Points: 1**

**Text:** Screenshot of your project board from GitHub (tasks should be in the proper column)

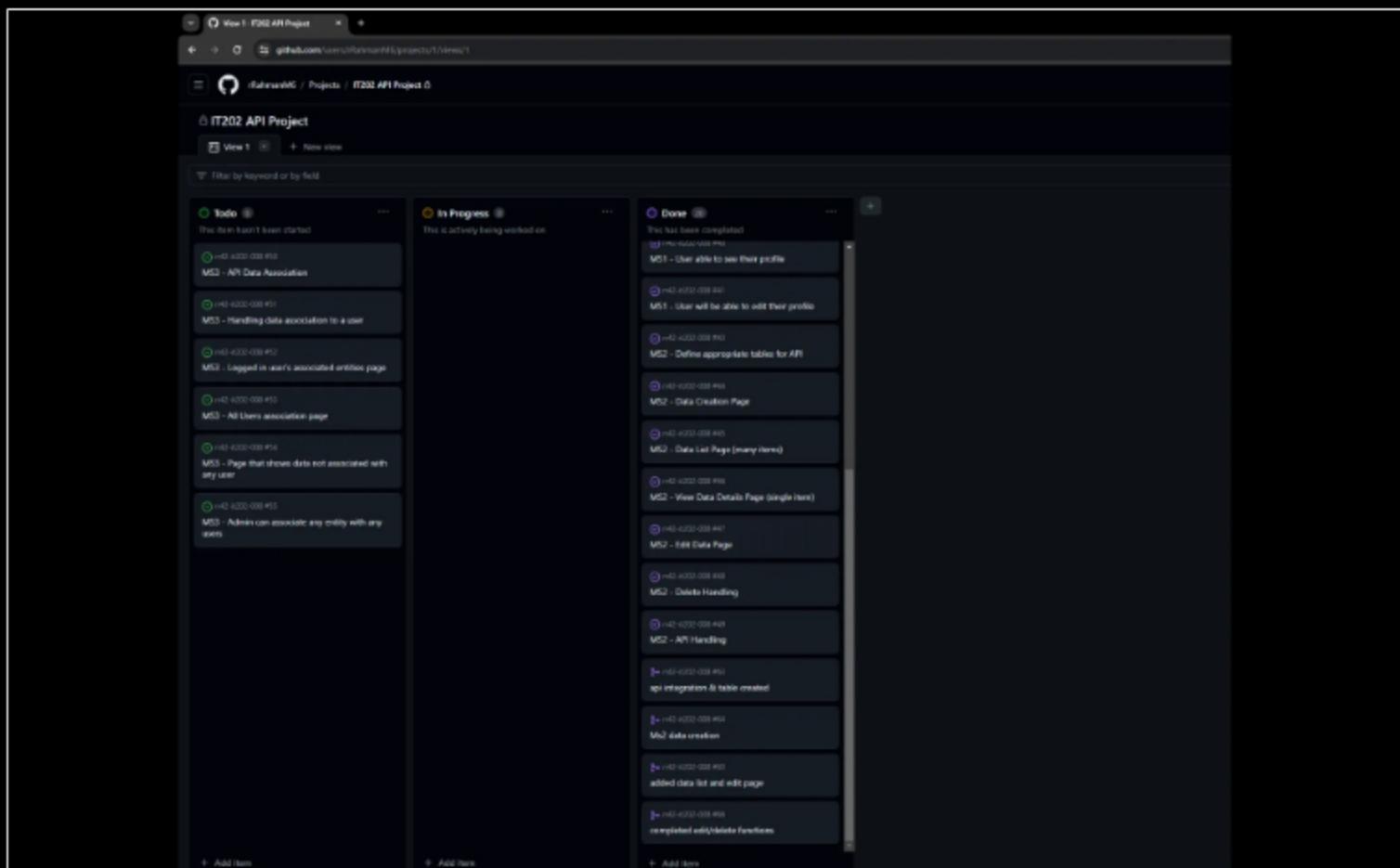
**Task Screenshots:**

**Gallery Style: Large View**

**Small**

**Medium**

**Large**



**project board**

**Task #2 - Points: 1**

[COLLAPSE](#)

## Task #2 - Points: 1

**Text:** Provide a direct link to the project board on GitHub

**URL #1**

<https://github.com/users/rRahmanM6/projects/1/views/1>



[COLLAPSE](#)

## Task #3 - Points: 1

**Text:** Talk about any issues or learnings during this assignment

**Response:**

I will say the most difficult part of this project so far has been figuring out how to fetch and work with the API data. I had to do a lot of testing and trial & error to get it to work how I wanted just because of the way my API returns data. After working through that, most other features were relatively straightforward.



[COLLAPSE](#)

## Task #4 - Points: 1

**Text:** WakaTime Screenshot

### Details:

Grab a snippet showing the approximate time involved that clearly shows your repository. The duration isn't considered for grading, but there should be some time involved

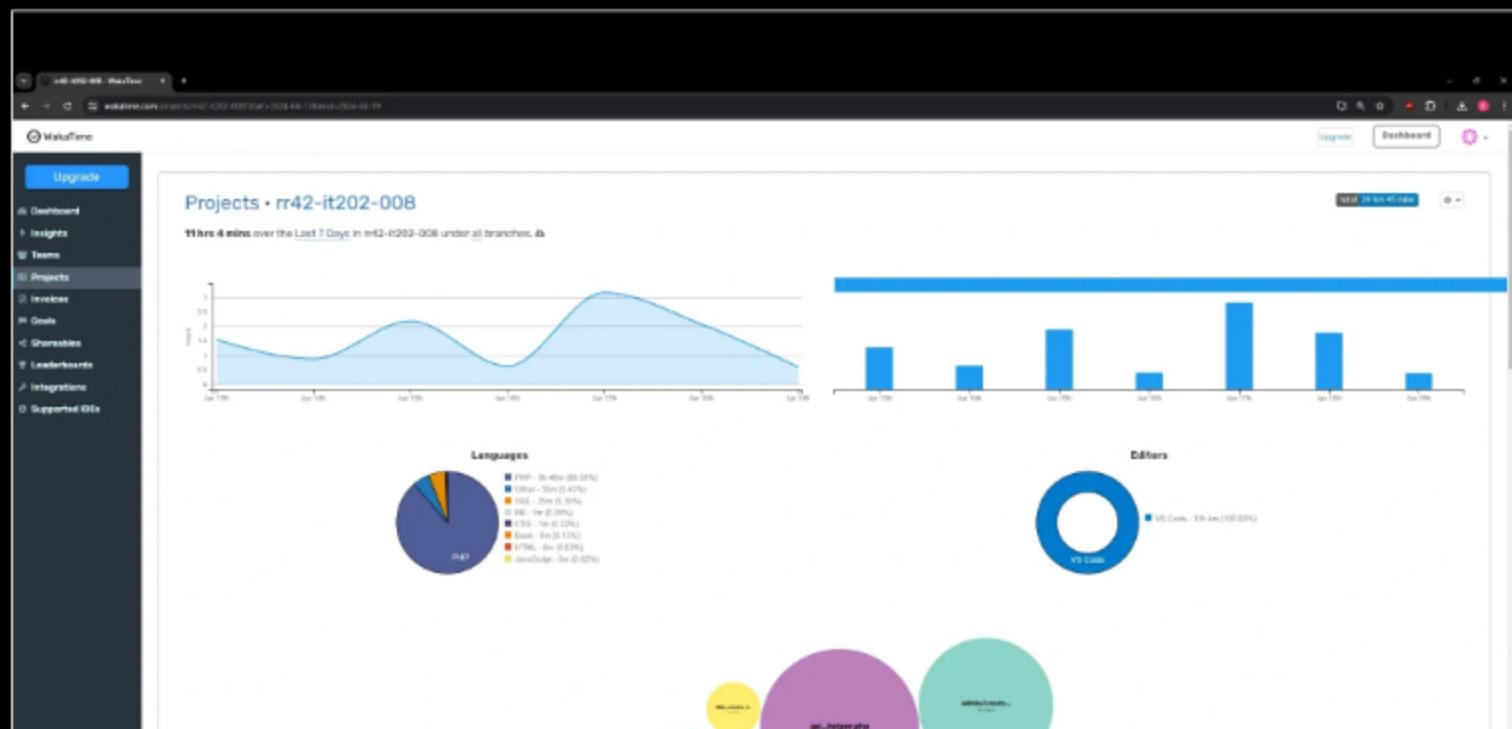
**Task Screenshots:**

#### Gallery Style: Large View

Small

Medium

Large



Bug  
API Data  
Pulling  
Community  
FAQ  
Plugin Status  
About

login.php

admin/edit.php

wakatime

End of Assignment