Course: IT202-008-S2025
Assignment: IT202 Milestone 2
Student: Dylan W. (dw347)
Status: Submitted | Worksheet Progress: 100%
Potential Grade: 10.00/10.00 (100.00%)
Received Grade: 0.00/10.00 (0.00%)
Grading Link: https://learn.ethereallab.app/assignment/v3/IT202-008-S2025/it202-milestone-2/grading/dw347

Instructions

1. Refer to Milestone2 of this doc: https://docs.google.com/document/d/1XE96a8DQ52Vp49XACBDTNCq0xYDt3kF29cO88EWVwfo/view
2. Ensure you read all instructions and objectives before starting.
3. Ensure you've gone through each lesson related to this Milestone
4. Switch to the Milestone2 branch
    1. `git checkout Milestone2` (ensure proper starting branch)
    2. `git pull origin Milestone2` (ensure history is up to date)
5. Fill out the below worksheet
    - Ensure there's a comment with your UCID, date, and brief summary of the snippet in each screenshot
    - Ensure proper styling is applied to each page
    - Ensure there are no visible technical errors; only user-friendly messages are allowed
6. Once finished, click "Submit and Export"
7. Locally add the generated PDF to a folder of your choosing inside your repository folder and move it to Github
    1. `git add .`
    2. `git commit -m "adding PDF"`
    3. `git push origin Milestone2`
    4. On Github merge the pull request from `Milestone2` to `dev`
    5. On Github create a pull request from `dev` to `prod` and immediately merge. (This will trigger the `prod` deploy to make the heroku prod links work)
8. Upload the same PDF to Canvas
9. Sync Local
10. `git checkout dev`
11. `git pull origin dev`

100%

Section #1: (2 pts.) Data

100%

Task #1 (0.50 pts.) - API Data Table

Combo Task:

Weight: *25%*
Objective: *API Data Table*

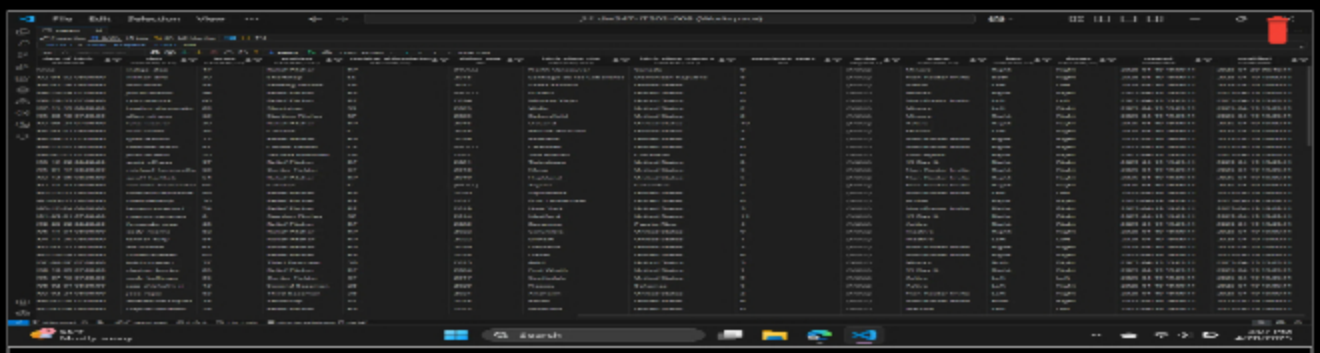
Image Prompt

Weight: *34%*

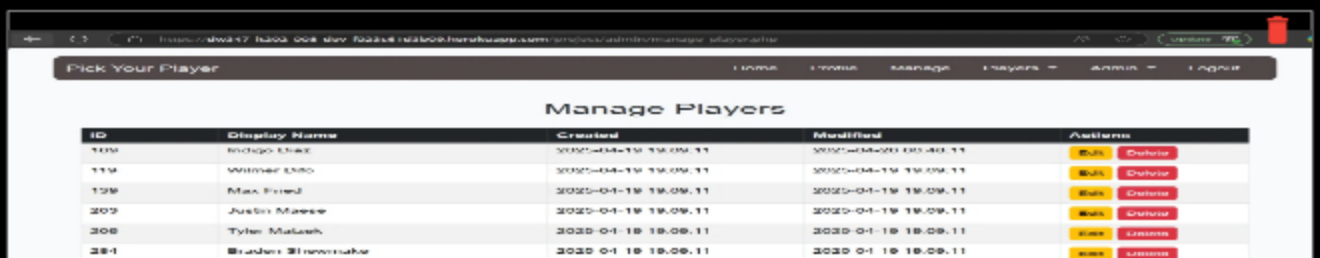
Details:

- Show the table(s) structure of the table made to hold your API
- There should be at least 3 properties/fields beyond `id`, `created`, `modified`, and respective `is_api` or `api_id` columns
- Logical types and constraints should be set
- Show at least a few records from each API table (samples should include both API fetched data and custom entries noted by an `api_id` or `is_api column)

sql table first half



sql table second half with created and modified



dev output table with created, id, modified and records

create table sql

alter table for created and modified

Saved: 4/20/2025 3:29:11 PM

Url Prompt

Weight: *33%*

Details:

- Add a direct link to your sql file from Github (should end in `.sql`)

URL #1

https://github.com/dyylan2018/dw347-IT2021008/Milestone2/public_html/project/sql/009_create_table_players.sql

URL

https://github.com/dyylan2018/dw

URL #2

https://github.com/dyylan2018/dw347-IT2021008/Milestone2/public_html/project/sql/010_alter_table_players.sql

URL

https://github.com/dyylan2018/dw

Task #2 (0.50 pts.) - Data Transformation

Combo Task:

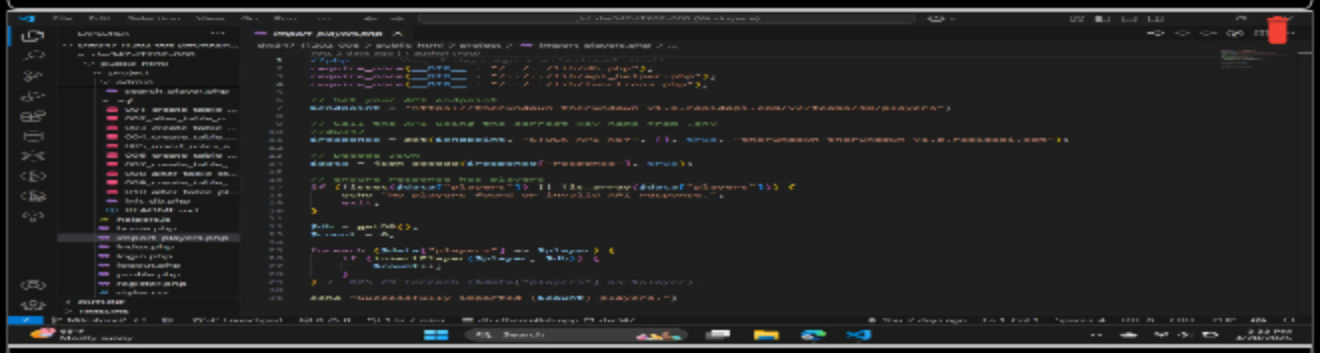
Weight: *25%*
Objective: *Data Transformation*

Image Prompt

Weight: *40%*

Details:

- Show the code that handles the data transformation/conversion (from the API)
- Show a Heroku dev example of API getting fetched and loaded into the DB



script to import plyers

output of imported players

Saved: 4/20/2025 9:03:22 PM

Url Prompt

Weight: *20%*

Details:

- Include Heroku prod link to API fetch page
- Include pull request link for this feature

URL #1
https://dw347-it202-008-prod-39957dda50b0.herokuapp.com/project/import_players.php

https://dw347-it202-008-prod-3995

URL #2
https://github.com/dyylan2018/dw347-IT202-008/Milestone2/public_html/project/import_players.php

https://github.com/dyylan2018/dw

URL #3
https://github.com/dyylan2018/dw347-IT202-008

https://github.com/dyylan2018/dw

Saved: 4/20/2025 9:03:22 PM

Text Prompt

Weight: *40%*

Details:

- Will you be using all the data or just a subset?
- Do you need to convert any data or extract pieces of a property's value?
- Briefly explain the transformation/extraction logic to get the API data to the shape your application intends to use

Your Response:

I imported all the data to the table to avoid issues but i am only going to use some of the selected data like the created, modified, ID, name, display name, weight, height, age, position, status, bats and the important factors. Some are duplicates so no need to use them. i had to convert the date to be executable for users to input. The transformation logic involves sending a request to the MLB API to fetch raw player data, which is returned as nested JSON. This data is looped through, and only relevant fields (like `id`, `first_name`, `last_name`, `team_id`, etc.) are extracted. These fields are then mapped to match the structure of our local `Players` table. Default values (like `active = 1`, `status = 'active'`, `bats = 'R'`) are added for consistency. Finally, each player is inserted into the database using a prepared SQL statement, with an `is_api` flag set to indicate the entry came from the API.

Task #3 (0.50 pts.) - API Duplicates

Text Prompt

Weight: *25%*
Objective: *API Duplicates*

Details:

- Consider how duplicate entries are handled
  - What happens if you get the same result from the API for an entity that exists but has been unmodified?
  - What happens if you get the same result from the API for an entity that exists but has been modified?

Your Response:

Duplicate entries from the API are handled by first checking if the player already exists in the database. If the data is unchanged, the script skips updating to avoid unnecessary writes. However, if the data has been modified, the script updates the existing record to reflect the latest changes. This ensures the local database stays in sync with the API while preventing redundant inserts and preserving performance.

Saved: 4/20/2025 3:58:41 PM

100%

Task #4 (0.50 pts.) - API Fetching

Text Prompt

Weight: *25%*
Objective: *API Fetching*

Details:

- Consider how your application will trigger this data fetch
  - (i.e., periodic via cron job, poor man's cron job, on app start/awake, etc)
  - (i.e., User or Admin triggered via a specific page/action)
  - (i.e., Lazy loaded during searches for stale/missing content)
  - What happens if you get the same result from the API for an entity that exists but has been modified?

Your Response:

The data fetch in this application is triggered manually by an Admin via a specific page/action, allowing controlled synchronization with the API. When the same entity is received but has been modified, the application detects the changes by comparing the current database values with the incoming API data. If differences are found, the existing record is updated to ensure the database reflects the latest available data from the API. This approach keeps the data accurate while preventing unnecessary updates.

Saved: 4/20/2025 4:00:15 PM

Section #2: (1.5 pts.) Data Creation

Task #1 (0.60 pts.) - Validations

Combo Task:

Weight: *40%*
Objective: *Validations*

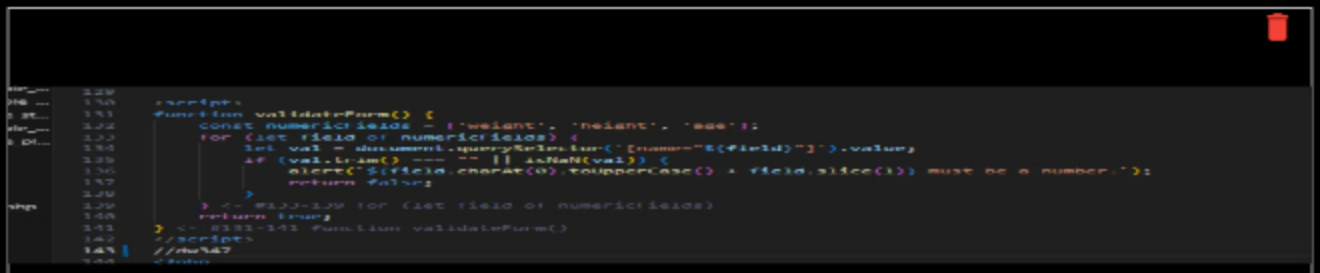
Image Prompt

Weight: *50%*

Details:

- Show the html validations (code)
- Show the javascript validations (code)
- Show the php validations (code)

html validation



is validation

php validation

Saved: 4/20/2025 7:21:52 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain the html validations used
- Briefly explain the javascript validation logic
- Briefly explain the php validation logic

Your Response:

The form uses a combination of HTML, JavaScript, and PHP validations to ensure reliable data entry. HTML validations include the `required` attribute on all fields and specific input types like `number` and `date` to restrict input formats. JavaScript adds client-side checks on numeric fields (`weight`, `height`, `age`) to prevent submission if values are empty or non-numeric, displaying alerts accordingly. PHP provides server-side validation by checking that all required fields are set and not empty, and that numeric fields contain valid numbers using `is_numeric()`. Errors are stored and displayed using flash messages, ensuring data is validated both on the client and server sides before inserting into

Task #2 (0.60 pts.) - Examples

Image Prompt

Weight: *40%*
Objective: *Examples*

Details:

- Show a successful creation
- Demonstrate various validation errors

- Ensure heroku url is visible

create player

created player successfully

Task #3 (0.30 pts.) - Links

Url Prompt

Weight: *20%*
Objective: *Links*

Details:

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1
https://dw347-it202-008-prod-39957dda50b0.herokuapp.com/project/admin/create_player.php

URL
https://dw347-it202-008-prod-39957dd

URL #2

URL

https://github.com/dyylan2018/dw347-IT202-001/98

https://github.com/dyylan2018/dw347

URL #3

https://github.com/dyylan2018/dw347-IT202-001/Milestone2/public_html/project/admin/create_player.php

https://github.com/dyylan2018/dw347

Saved: 4/20/2025 9:04:26 PM

Section #3: (1.5 pts.) Data List Page (Many Items)

Task #1 (0.60 pts.) - Code

Combo Task:

Weight: *40%*
Objective: *Code*

Image Prompt

Weight: *50%*

Details:

- Show the code that handles the filter/sort logic
- Show the code that generates the list/grid output
- Styling must be applied

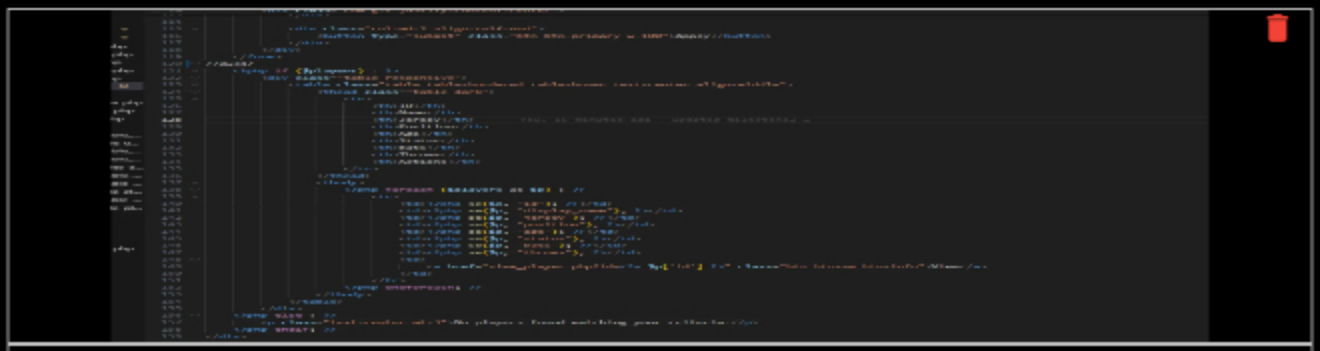
filter/sort logic

```php
    switch ($sort) {
        case 'age':
            $query .= " ORDER BY age DESC";
            break;
        case 'name':
            $query .= " ORDER BY display_name ASC";
            break;
        case 'jersey':
            $query .= " ORDER BY jersey ASC";
            break;
        default:
            $query .= " ORDER BY created DESC";
    } <  #49 61 switch ($sort)

    $db = getDB();
    $stmt = $db->prepare($query);
    $stmt->execute($params);
    $players = $stmt->fetchAll();
    //dw147
    ?>
```

end piece of filter/sort logic

list/grid



other half of list/grid

Saved: 4/20/2025 6:55:48 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain how the filter/sort logic works and formulates the results
- Briefly explain how the list/grid output is generated
- Briefly note the styling choices

Task #2 (0.60 pts) - Examples

Task #2 (0.60 pts.) - Examples

Image Prompt

Weight: *40%*
Objective: *Examples*

Details:

- Show a few variations of filter/sort applied (including no records)
- Each list item should be a summary
- Each list item should have the following links
  - A link to a single view of the specific entity (i.e., a details page)
  - A link to delete this entity (this may be an admin-only functionality, but it should be present for the respective role)
  - A link to edit this entity (this may be an admin-only functionality, but it should be present for the respective role)
- Ensure heroku url is visible

filtered with button to full details

filtered

Task #3 (0.30 pts.) - Links

Url Prompt

Weight: *20%*
Objective: *Links*

Details:

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1
https://dw347-it202-008-prod-39957dda50b0.herokuapp.com/project/admin/list_players.php

https://dw347-it202-008-prod-39957dd

URL #2
https://github.com/dyylan2018/dw347-IT202-008/tree/01

https://github.com/dyylan2018/dw347

URL #3
https://github.com/dyylan2018/dw347-IT202-008/Milestone2/public_html/project/admin/list_players.php

https://github.com/dyylan2018/dw347

Saved: 4/20/2025 9:05:16 PM

Section #4: (1.5 pts.) View Details Page (Single Item)

Task #1 (0.60 pts.) - Code

Combo Task:

Weight: *40%*
Objective: *Code*

Image Prompt

Weight: *50%*

Details:

- Show the code that handles loading of the record (and how the id retrieved for usage)

- Show the code that handles loading of the record (and how the id retrieved for usage)
  - Include the code that handles invalid/missing ids
- Show the code that generates the entity output
- Styling must be applied

```php
$player = null;
$id = (int)($_GET["id"] ?? 0);
//dw347
if ($id > 0) {
    $db = getDB();
    $stmt = $db->prepare("SELECT * FROM Players WHERE id = :id");
    $stmt->execute([":id" => $id]);
    $player = $stmt->fetch();
} <- #13-18 if ($id > 0)
?>
```

loading of the record

```php
                </form>
            </div>
                <a href="list_players.php" class="btn btn-secondary mt-3 w
            </div>
        <?php else : ?>
            <p class="text-danger text-center">Player not found.</p>
        <?php endif; ?>
    </div>
    //dw347
    <?php
```

invalid id

```php
<div class="container py-5">
    <h1 class="text-center mb-4">Player Details</h1>

    <?php if ($player) : ?>
        <div class="card shadow p-4 mx-auto" style="max-width: 600px;">
            <h4 class="mb-3 text-primary"><?php se($player, "display_name"); ?></h4>
            <ul class="list-group list-group-flush text-start">
                <li class="list-group-item"><strong>ID:</strong> <?php se($player, "id"); ?></li>
                <li class="list-group-item"><strong>Jersey:</strong> <?php se($player, "jersey"); ?></li>
                <li class="list-group-item"><strong>Position:</strong> <?php se($player, "position"); ?></li>
                <li class="list-group-item"><strong>Age:</strong> <?php se($player, "age"); ?></li>
                <li class="list-group-item"><strong>Status:</strong> <?php se($player, "status"); ?></li>
                <li class="list-group-item"><strong>Bats:</strong> <?php se($player, "bats"); ?></li>
                <li class="list-group-item"><strong>Throws:</strong> <?php se($player, "throws"); ?></li>
                <li class="list-group-item"><strong>Date of Birth:</strong> <?php se($player, "date_of_birth"); ?></
                <li class="list-group-item"><strong>Weight:</strong> <?php se($player, "display_weight"); ?></li>
                <li class="list-group-item"><strong>Height:</strong> <?php se($player, "display_height"); ?></li>
                <li class="list-group-item"><strong>Created:</strong> <?php se($player, "created"); ?></li>
                <li class="list-group-item"><strong>Modified:</strong> <?php se($player, "modified"); ?></li>
            </ul>
    //dw347
```

generates entity output

```html
<!-- Action Buttons -->
<style>
...
.action-btn
{
    height: 40px;
    display: flex;
    align-items: center;
    justify-content: center;
}//dw347 <- #45-50 .action-btn
</style>
```

styling

Saved: 4/20/2025 7:37:27 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain how the loading logic works
- Briefly explain how the entity output is generated
- Briefly note the styling choices

Task #2 (0.60 pts.) - Examples

Image Prompt

Weight: *40%*
Objective: *Examples*

Details:

- Show a few examples of different entities
- Each entity item should have more details than the summary view (if possible)
- Each entity should have the following links
  - A link to edit the single entity (this may be an admin-only thing, but it should be present for the respective role)
  - A link to delete the single entity (this may be an admin-only thing, but it should be present for the respective role)

one players viewing info

another player info

Task #3 (0.30 pts.) - Links

Url Prompt

Weight: *20%*
Objective: *Links*

Details:

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1
https://dw347-it202-008-prod-39957dda50b0.herokuapp.com/project/admin/view_player.php?id=2640

URL #2
https://github.com/dyylan2018/dw347-IT202-008/Milestone2/public_html/project/admin/view_player.php

URL #3
https://github.com/dyylan2018/dw347-IT202-008/pull/102

Saved: 4/20/2025 9:06:21 PM

Section #5: (1.5 pts.) Edit Page

Task #1 (0.60 pts.) - Code

Combo Task:

Weight: *40%*
Objective: *Code*

Image Prompt

Weight: *50%*

Details:

- Show the code that handles loading of the record (and how the id retrieved for usage)
  - Include the code that handles invalid/missing ids
- Show the code that generates the form with the correct data types and populates it
- Show the html, javascript, and php validation (should be similar to create)
- Styling must be applied

```php
// GET: Load player by ID
if ($_SERVER["REQUEST_METHOD"] === "GET" && isset($_GET["id"])) {
    $db = getDB();
    $stmt = $db->prepare("SELECT * FROM Players WHERE id = :id");
    $stmt->execute([":id" => $_GET["id"]]);
    $player = $stmt->fetch();
}//dwdd7 <- [illegible]
```

loading of data

```php
<?php
$fields = [
    "sport_id" => "Sport ID", "team_id" => "Team ID", "first_name" => "First Name",
    "last name" as "last name", "display name" as "display name", "weight" as "weight",
    "height" => "height", "age" => "Age", "date of birth" => "date of birth", "position" => "position"
]; <- [illegible]
foreach ($fields as $field => $label):
?>
    <div class="mb-3">
        <label for="<?= $field ?>" class="form-label"><?= $label ?></label>
        <input
            type="<?= ($field === 'date of birth') ? 'date' : (is_numeric($player[$field]) ? 'number' : 'text') ?>"
            class="form-control mx-auto"
            style="max-width: 400px;"
            name="<?= $field ?>"
            id="<?= $field ?>"
            value="<?= htmlspecialchars($player[$field]) ?>">
    </div></div></div>
<?php endforeach; ?>
```

generates form of data

```php
    // handle date_of_birth separately to retain value if not changed
    $dob = $_POST["date_of_birth"] ?? "";
    if (!empty($dob)) {
        $updates[] = "date of birth = :date of birth";
        $params[":date_of_birth"] = $dob;
    }//dwdd

    $params[":id"] = $id;

    $db = getDB();
    $stmt = $db->prepare("UPDATE Players SET " . implode(", ", $updates) . ", updated at = NOW() WHERE id = :id");
    $stmt->execute($params);

    // Redirect back to this page with success flag
    header("Location: edit_player.php?id=" . $id . "&success=1");
    exit;
}; <- [illegible] if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST["id...
```

validation

```
<head>
    <title>Edit Player</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <style>
        ...
        .centered-form {
            max-width: 600px;
            margin: auto;
            text-align: center;
        } <-- #/2/6/ .centered-form
        ...
        .form-control {
            text-align: center;
        }
    </style>
    //dw244
</head>
<body>
```

styling

Saved: 4/21/2025 6:15:31 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain how the loading logic works
- Briefly explain how the form is generated and submission is handled
- Briefly note the styling choices
- Note what fields are editable and why the choice was made for them (it's likely not all fields are editable, especially `id`, `created`, `modified`, etc)

100%

Task #2 (0.60 pts.) - Examples

Image Prompt

Weight: *40%*
Objective: *Examples*

Details:

- Show a before and after of updating an entity (clearly caption)
- Successful update should have an appropriate message shown
- The updated data should be shown in the form
- Any errors should have user-friendly messages shown
- Show some examples of validation errors (html, javascript, php)

before update

Task #3 (0.30 pts.) - Links

Url Prompt

Weight: *20%*
Objective: *Links*

Details:

- Include the heroku prod link to this page
- Include pull request link for this feature

URL #1
https://dw347-it202-008-prod-39957dda50b0.herokuapp.com/project/admin/edit_player.php
URL: https://dw347-it202-008-prod-39957dd

URL #2
https://github.com/dyylan2018/dw347-IT202-008/Milestone2/public_html/project/admin/edit_player.php
URL: https://github.com/dyylan2018/dw347

URL #3
https://github.com/dyylan2018/dw347-IT202-008/04
URL: https://github.com/dyylan2018/dw347

Saved: 4/21/2025 6:18:32 PM

Section #6: (1.5 pts.) Delete Logic

Task #1 (0.60 pts.) - Code

Combo Task:

Weight: *40%*
Objective: *Code*

Image Prompt

Weight: *50%*

Details:

- Show the code that handles deletion of the record (and how the id retrieved for usage)
  - Include the code that handles invalid/missing ids
- Handle any necessary roles/permissions (i.e., it's not likely that anyone can delete any entity they choose)
- Show the redirect logic that goes back to the previous page (where the route came from), if it was a list page, it should maintain the filter/sort criteria

```php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['display_name'] ?? '');

    if ($name === '') {
        flash("Please enter a player's display name.", "danger");//dwj4
    } else {
        try {
            $db = getDB();
            $stmt = $db->prepare("DELETE FROM Players WHERE display_name = :display_name");
            $stmt->execute([":display_name" => $name]);
```

deletion of record

```php
            $stmt->execute([":display_name" => $name]);

            if ($stmt->rowCount() > 0) {
                flash("Player deleted successfully!", "success");
            } else {
                flash("No player found with that display name.", "warning");
            }
        } catch (PDOException $e) {
            error_log("Delete player error: " . var_export($e, true));
            flash("Error deleting player: " . htmlspecialchars($e->getMessage()), "danger");//dwj4?
        }
    } // missing else
} // end if ($_SERVER['REQUEST_METHOD'] === 'POST')
```

invalid/missing IDs

```php
require_once(__DIR__ . "/../../../lib/db.php");
```

```php
if (!has_role("Admin")) {
    flash("You don't have permission to view this page", "warning");
    die(header("Location: " . get_url("home.php")));
}//dw447
```

roles/permissions

```php
$redirect = $_GET['redirect'] ?? get_url("list_players.php");
//dw447
```

redirect logic

Saved: 4/21/2025 9:01:39 PM

Task #2 (0.60 pts.) - Examples

Image Prompt

Weight: *40%*
Objective: *Examples*

Details:

- Show a successful delete message
- Show a before and after database view of the record being deleted (clearly caption)
- Successful update should have an appropriate message shown
- The updated data should be shown in the form
- Any errors should have user-friendly messages shown

successful delete message

Task #3 (0.30 pts.) - Links

Url Prompt

Weight: *20%*
Objective: *Links*

Details:

- Include the heroku prod link to this page (even though it's not a full page, it's more like how logout works)
- Include pull request link for this feature

URL #1
https://dw347-it202-008-prod-39957dda50b0.herokuapp.com/project/admin/delete_player.php
URL
https://dw347-it202-008-prod-39957dd

URL #2
https://github.com/dyylan2018/dw347-IT202-008/Milestone2/public_html/project/admin/delete_player.php
URL
https://github.com/dyylan2018/dw347

Section #7: (0.5 pts.) Misc

Task #1 (0.17 pts.) - Github Details

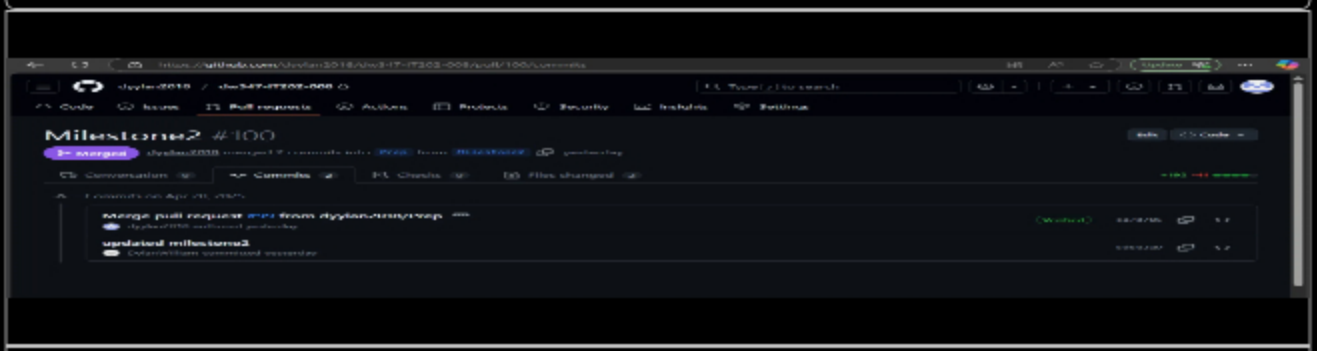
Combo Task:

Weight: *33.33%*
Objective: *Github Details*

Image Prompt

Weight: *60%*

Details:
From the Commits tab of the Pull Request screenshot the commit history



commits

Saved: 4/21/2025 9:18:22 PM

Url Prompt

Weight: *40%*

Details:
Include the link to the Pull Request (should end in `/pull/#`)

URL #1
https://github.com/dyylan2018/dw347-IT202-008/100

URL
https://github.com/dyylan2018/dw:

Saved: 4/21/2025 9:18:22 PM

Task #2 (0.17 pts.) - WakaTime - Activity

Image Prompt

Weight: *33.33%*
Objective: *WakaTime - Activity*

Details:

- Visit the WakaTime.com Dashboard

- Click `Projects` and find your repository
- Capture the overall time at the top that includes the repository name
- Capture the individual time at the bottom that includes the file time
- Note: The duration isn't relevant for the grade and the visual graphs aren't necessary

overall

Task #3 (0.17 pts.) - Reflection

Weight: *33.33%*
Objective: *Reflection*

Sub-Tasks:

Task #1 (0.33 pts.) - What did you learn?

Text Prompt

Weight: *33.33%*
Objective: *What did you learn?*

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

ALOT: i learned to navigate pages, delete, import list, fetch, filter & more

ALOT. I learned to navigate pages, delete, import, list, fetch, filter, & more.

Saved: 4/21/2025 9:21:33 PM

100%

Task #2 (0.33 pts.) - What was the easiest part of the assignr

Text Prompt

Weight: *33.33%*
Objective: *What was the easiest part of the assignment?*

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

making the lists and allowing users to view entities

Saved: 4/21/2025 9:21:59 PM

100%

Task #3 (0.33 pts.) - What was the hardest part of the assign

Text Prompt

Weight: *33.33%*
Objective: *What was the hardest part of the assignment?*

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

making the script to import the data from the api to the database

Saved: 4/21/2025 9:22:21 PM