STAGS

Shoot then aim gymnastics scores

Iteration 1 — Planning

Goals for Iteration 1

1. Design what the website will look like, the colours, font and size.
2. Create a database to store all the gymnasts' information and scores and the SQL queries that will be needed.

Website explanation

The site will support and help men's artistic gymnastic judges. It will calculate a series of scores to give a final score and display it on a leaderboard. The leaderboard will display six categories representing each apparatus: floor, pommel, rings, vault, parallel bars, and high bar. My initial plan was to have a separate page to edit all the gymnasts and scores. However, I decided to incorporate the editing process in the corresponding pages to simplify the site. Thus, this site will provide a page where you can enter and edit gymnasts, a page where you can add and edit scores, and a page where you can view the current leaderboard standings. There will also be a homepage with a tutorial on using the website.

How the scores are calculated

In men's artistic gymnastics, a gymnast's routine is judged based on two main components: the difficulty score (D-score) and the execution score (E-score). The D-score measures how challenging the gymnast's routine is, whereas the E-score measures how well the routine performs; the E-score is scored out of 10. The final score is the sum of both the D-score and the E-score.

Routes

- Home page: /
- Register/edit gymnasts: /addgymnast
- Register/edit scores: /addscores
- Leaderboard: /leaderboard

Font

The Font will be Lexend. I got this font from ‘google fonts’ I got an import:

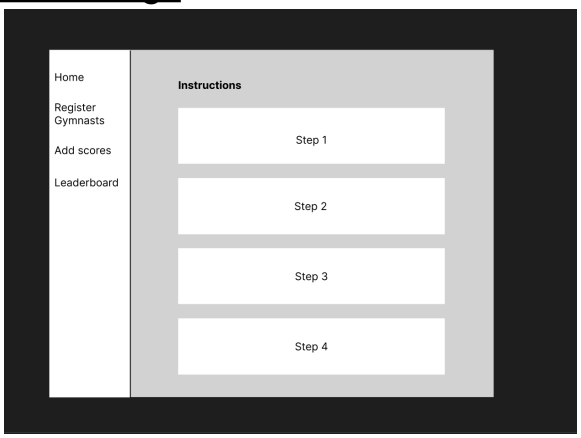
```
@import url('https://fonts.googleapis.com/css2?family=Lexend:wght@100..900&display=swap');
```

Website Colours

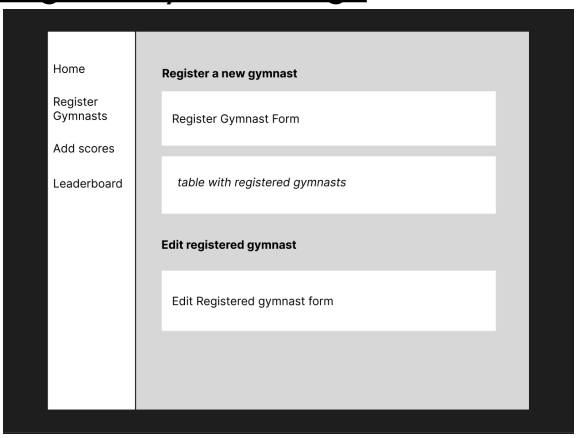
I chose these colours because they are simple, making the website much easier for users to read, navigate, and use.

Website Designs

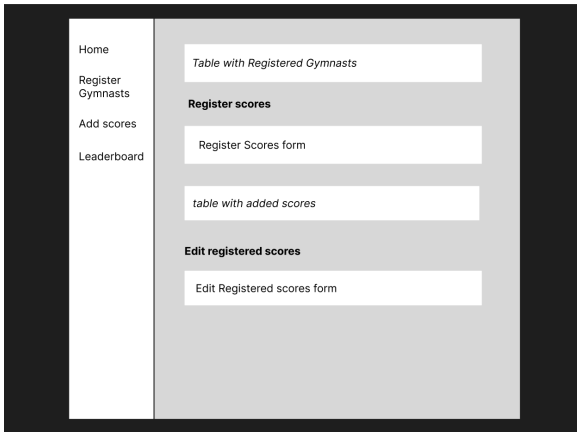
Home Page



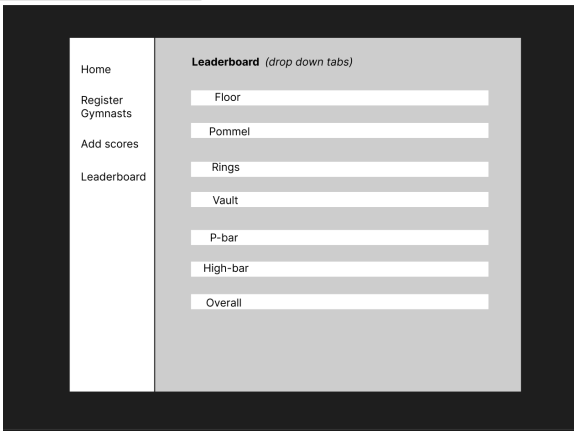
Register Gymnast Page



Register Score Page



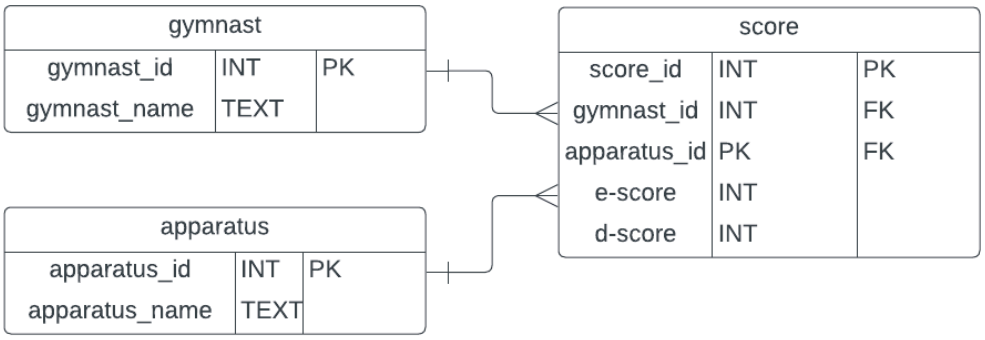
Leaderboard



Initially, I decided to have a table with apparatuses at the top of the register score page, but later, I decided to add a dropdown select in the forms.

Database

ER Diagram



In my database, there will be three tables. The first table will get and store the gymnasts' names and give them each a unique ID. The second table stores all the apparatuses. The third table gets and stores the gymnasts' scores, which are then displayed on the leaderboard.

SQL Entries

What for?	**SQL Entries**
Adding gymnast	INSERT INTO gymnast (gymnast_name) VALUES (?); *This SQL query is used to add data into the gymnasts table, it asks for the gymnast_name.*
Viewing gymnasts	SELECT * FROM gymnast; *This SQL query selects everything from the gymnast table.*
Adding scores	INSERT INTO scores (gymnast_id, apparatus_id, escore, dscore) VALUES (?,?,?,?); *This SQL query is used to add data into the scores table, it asks for the gymnast_id, apparatus_id, escore and dscore.*
Viewing scores	SELECT * FROM score *This SQL query selects everything from the score table.*
Overall Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, SUM(score.dscore + score.escore) AS total

	FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id GROUP BY score.gymnast_id, gymnast.gymnast_name ORDER BY total DESC *This SQL query ranks gymnasts by their total scores (sum of dscore and escore), joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
Floor Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore,score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE apparatus.apparatus_id = 6 ORDER BY total DESC *This SQL query ranks the gymnasts by their total scores (sum of dscore and escore) where the apparatus_id = 6, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
Pommel Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore,score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE apparatus.apparatus_id = 5 ORDER BY total DESC *This SQL query ranks the gymnasts by their total scores (sum of dscore and escore) where the apparatus_id = 5, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
RIngs Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore,score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE apparatus.apparatus_id = 4 ORDER BY total DESC *This SQL query ranks the gymnasts by their total scores (sum of dscore and escore) where the apparatus_id = 4, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*

Vault Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore,score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE apparatus.apparatus_id = 3 ORDER BY total DESC *This SQL query ranks the gymnasts by their total scores (sum of dscore and escore) where the apparatus_id = 3, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
P-bar Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore,score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE apparatus.apparatus_id = 2 ORDER BY total DESC *This SQL query ranks the gymnasts by their total scores (sum of dscore and escore) where the apparatus_id = 2, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
High bar Leaderboard	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore,score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE apparatus.apparatus_id = 1 ORDER BY total DESC *This SQL query ranks the gymnasts by their total scores (sum of dscore and escore) where the apparatus_id = 1, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
Edit registered gymnasts	UPDATE gymnast SET gymnast_name = ? WHERE gymnast_id = ? *This SQL query updates the gymnast table by setting gymnast_name to a new or same values based on the gymnast_id given.*
Edit registered scores	UPDATE score SET apparatus _id = ?, escore = ?, dscore = ? WHERE score_id = ? *This SQL query updates the gymnast table by setting apparatus_id, escore and descore to new or same values based*

	on the score_id given.

Functions

Def homepage()
Def gymnast()
Def scores()
Def leaderboard()

Iteration 2 — Setting up Pages

Goals for Iteration 2

1. Set up all the routes, a template folder, and a static folder. The templates folder will contain all the HTML code for each page. The static folder will hold all the styling, including the CSS and Javascript code.
2. Link up all the pages, so create links in the nav file and connect the HTML pages by putting links on all the pages. These links will be used in the navbar later on.

First, I created a template folder and a static folder, including all the files needed in both folders.

Template Folder:

- layout.html
  This HTML file contains the layout of each page.

- nav.html
  This HTML file contains all the page links for the navbar.

- footer.html
  This HTML file contains what is going to be in the footer.

- home.html
  This HTML file contains the main content for the home page.

- gymnast.html
  This HTML file contains the main content for the register gymnast page.

- score.html
  This HTML file contains the main content for the adding of scores page.

- leaderboard.html
  This HTML file contains the main content for the leaderboard page

Static Folder

- myscript.js
  This file contains all the javascript code

- style.css
  This file contains all the CSS code.

Next, I started adding some code connecting each of the pages.

In the layout.html file, I created three sections for the layout of my pages: a navbar, the main, and the footer. I used {% include 'nav.html' %} for the navbar. This gets the stored information from the nav.html file. I used {% block content %}{% endblock %} for the page's main content. At the start of the gymnast.html, score.html, leaderboard.html files, I used {% extends 'layout.html' %} and {% block content %}, at the end of the files, I used {% endblock %}. These use the information on each file and put it in the 'main' part of the layout.html file. I used {% include 'footer.html' %} for the footer. This gets the stored information from the footer.html file.

Finally, I added links to each of the pages in the nav.html file, which creates a link on each page that goes to other corresponding pages.

Code on Layout.html file

```
<!DOCTYPE html>
<html>
  <head>
    <title>My website</title>
    <link rel="stylesheet" href="/static/style.css" />
    <script src="/static/myscript.js"></script>
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  </head>
  <body>
    <div>
      <nav>{% include 'nav.html' %}</nav>
      <main>{% block content%}{% endblock %}</main>
      <footer>{% include 'footer.html' %}</footer>
    </div>
  </body>
</html>
```

Code on Gymnast.html, score.html and leaderboard.html files.

```
{% extends 'layout.html' %}
{% block content %}
<!--code-->
{% endblock %}
```

The footer.html and nav.html files do not need any code to connect them to the layout.html file.

Iteration 3 — Forms/Tables/leaderboard

Goals for Iteration 3

1. How many forms will be needed, and what they will have in them?
2. Create the required tables for each form
3. Create all the forms for each page.
4. Create the leaderboard

This website contains four forms, two on the register gymnast page and two on the add scores page. There will also be a form on the register gymnast page and two on the add scores page.

Tables

Register Gymnast Page

1. Registered Gymnasts
    - ID
    - Name

This table displays all the registered gymnasts so you can get the required information to edit them.

```
<!--view registerd gymnasts-->
<div class = "tables">
    <table>
        <tr>
            <th id="tableheader" colspan="3">Registered Gymnasts</th>
        </tr>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Level</th>
        </tr>
        {% for row in regdata %}
        <tr>
            {% for i in row  %}
            <td>{{ i }}</td>
            {% endfor %}
        </tr>
        <tr>
        </tr>
        {% endfor %}
    </table>
```

Add Scores Page

1. Registered Gymnasts
    - ID

    - Name
2. Registered Scores
    - Score ID
    - Gymnast ID
    - Name
    - Apparatus
    - E-score
    - D-score

The first table displays the registered gymnasts so you don't need to keep flicking between pages. The second table displays the Registered scores so you can get the required information to edit the scores.

```
<!--view registerd gymnasts-->
<div class = "tables">
    <table>
        <tr>
            <th id="tableheader" colspan="3">Registered Gymnasts</th>
        </tr>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Level</th>
        </tr>
        {% for row in regdata %}
        <tr>
            {% for i in row  %}
            <td>{{ i }}</td>
            {% endfor %}
        </tr>
        <tr>
        </tr>
        {% endfor %}
    </table>
```

```
<!--table to view the registerd scores-->
<div class="tables">
    <table>
        <tr>
            <th id="tableheader" colspan="6">Registered Scores</th>
        </tr>
        <tr>
            <th>Score ID</th>
            <th>Gymnast ID</th>
            <th>Name</th>
            <th>Apparatus</th>
            <th>E-score</th>
            <th>D-score</th>
        </tr>
        {% for row in scoredata %}
        <tr>
            {% for col in row %}
            <td>{{ col }}</td>
            {% endfor %}
        </tr>
        {% endfor %}
    </table>
```

Forms

Register Gymnast Page

1. Register gymnast:
    - Gymnast Name
2. Edit gymnast
    - Gymnast ID
    - New Name

The first form on this page is to add gymnasts to the database (registering gymnasts for the competition). The second form allows you to edit the registered gymnast's name.

```
<!--Form to add gymnasts to the database-->
<form id = registor>
    <p> Name: <input type="text" name = "registorname" required></p>
    <input type="submit" />
</form>
```

```
<!--edit registered gymnasts-->
<form id="editgymnast">
    <p> ID: <input type="number" name="id" required></p>
    <p> Name: <input type="text" name="newname" required></p>
    <input type="submit">
</form>
```

Add Scores Page

1. Register Score
    - Gymnast ID
    - Apparatus
    - Execution score
    - Difficulty score
2. Edit Score:
    - Gymnast ID
    - Apparatus
    - Execution score
    - Difficulty score

The first form on this page is to add scores to the database for each gymnast and apparatus. The second form allows you to edit the added scores, you can change the scores or even change the apparatus. For the apparatus part of both forms, I initially wanted to have an input box where you enter the apparatus id. But I later on decided to incorporate a drop-down select. I also had to add 0.01 increments because the scores can be in decimals, for example, 8.87.

```
<!--add scores to the database-->
<form>
    <p>Gymnast ID: <input type="number" name = "addgymnastid"required></p>
    <p>
        Apparatus:
        <select name = "app" required>
    </p>
    <option value="" disabled hidden selected>Select</option>
    <option value="1">High Bar</option>
    <option value="2">P-bars</option>
    <option value="3">Vault</option>
    <option value="4">Rings</option>
    <option value="5">Pommel</option>
    <option value="6">Floor</option>
    </select>
    <p>Execution Score: <input type="number" name = "add-escore" min = "0" max = "10" step="0.01" required></p>
    <p>Difficulty Score: <input type="number" name = "add-dscore" min = "0" max = "10" step="0.01" required></p>
    <input type="submit">
</form>
```

```
<!--Form to edit scores-->
<form>
    <p>Score ID: <input type ="number" name = "scoreid" required></p>
    <p>
        Apparatus:
        <select name = "appid" required>
    </p>
    <option value="" disabled hidden selected>Select</option>
    <option value="1">High Bar</option>
    <option value="2">P-bars</option>
    <option value="3">Vault</option>
    <option value="4">Rings</option>
    <option value="5">Pommel</option>
    <option value="6">Floor</option>
    </select>
    <p>Execution Score: <input type ="number" name = "newescore" min = "0" max = "10" step="0.01" required></p>
    <p>Difficulty Score: <input type ="number" name = "newdscore" min = "0" max = "10" step="0.01" required></p>
    <input type = "submit">
</form>
```

Leaderboard

This page will display the leaderboard. I only needed one page to display this. I decided to use dropdown tables to display the leaderboard. I needed seven dropdown tables:

- All around
- Floor
- Pommel
- Rings
- Vault
- Parallel bars
- High bar

I used the same set of code for each of the dropdown tables.

```
<!--Table to display the all around leaderboard-->
<div class = "dropdown">
   <button onclick = "toggleDropdown('allaround')">All Around</button>
   <div id="allaround" class = "dropdown-content">
      <table>
         <tr>
            <th>id</th>
            <th>Name</th>
            <th>Score</th>
         </tr>
         {% for row in overalldata %}
         <tr>
            {% for col in row %}
            <td>{{ col }}</td>
            {% endfor %}
         </tr>
         {% endfor %}
      </table>
   </div>
</div>
```

For the dropdown table to work, I had to incorporate some Javascript.

```javascript
/* When the button is clicked switch between hiding and showing the dropdown content */
function toggleDropdown(dropdownId) {
    document.getElementById(dropdownId).classList.toggle("show");
}
```

The 'toggleDropdown(dropdownID)' allowed me to connect multiple different dropdowns to the one javascript function. This javascript function switches the table from hidden to showing the table when a button is clicked. At the same time, I added some styling to the dropdown table.

```css
dropdown-content {
  display: none;
  position: absolute;
  background-color: #f1f1f1;
  min-width: 160px;
  overflow: auto;
  box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
  z-index: 1;
}

show {
display: block;
}
```

Iteration 4 — CSS Styling/Footer

Navbar

I decided to make the navbar vertical on the left side of the page because I wanted to reduce the page's width. I decided to make the background colour of the Navbar white because I am making the background slightly grey. I added a hover effect to each of the links, making the background of the links slightly grey. I also added a small border on the side of the navbar to help separate the navbar from the background.

```css
ul {
  margin: 0;
  padding: 0;
  width: 10%;
  background-color: #ffffff;
  position: fixed;
  height: 100%;
  border-right-style: ridge;
  border-width: 2px;
  border-color:black;
}
li a {
  font-size: 120%;
  display: block;
  color: #000;
  padding: 20px 16px;
  text-decoration: none;
}
ul :hover {
  background-color: rgb(190, 189, 189);
}
```

Home page

I decided to centre some instructions for the homepage and make them span across the whole page. There are going to be four steps showing how to use my website. I made the background of each step of the instructions white, which will make the instructions more visible. I also added a small border around each of the steps.

```
/* Home Page */
#step1, #step2, #step3, #step4 {
  background-color: #ffffff;
  border: 2px #000 solid;
  margin: 10px;
  padding: 10px;
  width: 100%;
  text-align: center;
```

Register gymnast/Add score pages.

For each section of this page, I added a white background. Each section includes each form and each table.

Forms:
I decided to create long input boxes, and when you click on an input box, the border turns black and grey. I made the submit button have a grey background colour,

```css
/* Forms */
input[type=number] {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #000000;
  border-radius: 5px;
  box-sizing: border-box;
}
input[type=text], select {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #000000;
  border-radius: 5px;
  box-sizing: border-box;
}
input[type=submit] {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  display: inline-block;
  border: 1px solid #ffffff;
  border-radius: 5px;
  box-sizing: border-box;
  cursor: pointer;
}
input[type=submit]:hover {
  background-color: #b8b8b8;
}
.forms {
  border-radius: 5px;
  background-color: #ffffff;
  padding: 20px;
}
```

Tables:

For the tables, I decided to keep it simple by using black borders to separate the data.

```css
/* tables */
table, td, th {
  border: 1px solid;
  border-radius: 10px;
}
td, th {
  padding: 15px;
}
table {
  width: 100%;
  border-collapse: collapse;
}
.tables {
  border-radius: 5px;
  background-color: #ffffff;
  padding: 20px;
}
```

Footer:

I put no background behind the footer and added links to each page. I also put a copyright following some text.

```css
/* Footer */

.footer {
  width: 80%;
  margin: 0 14%;
  padding: 10px;
  text-align: center;
  color: #000000;
}

.footer-links a {
  text-decoration: none;
  color: #000000;
  padding: 10px;
}
```

Iteration 5 — Improvements

After making the prototype website, I made a few improvements around the website. These included:

- Apparatus section of the forms

  Initially, I had a table to view each of the apparatuses with their corresponding IDs, and when you selected an apparatus for entering scores, you had to input the ID. Later, I removed the table and replaced the input field with a dropdown select.

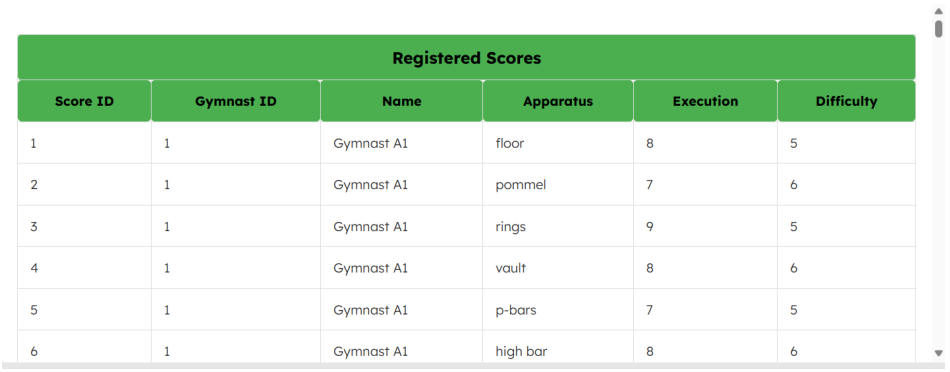
  This makes it easier for the user to select an apparatus without the complications.

- Website colours

  I changed the website colours to make it more aesthetically pleasing. These are the new main website colours:

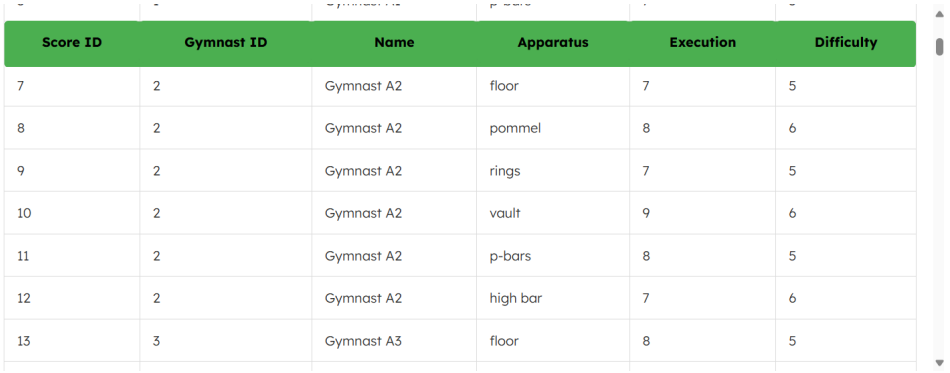
- Tables

I changed the designs of all the tables. For viewing gymnasts and scores, I have made it so that when you scroll, the heading follows. This will make it easier for the user to see which column is which. I also changed some colours to make it more readable.



Registered Scores					
Score ID	Gymnast ID	Name	Apparatus	Execution	Difficulty
1	1	Gymnast A1	floor	8	5
2	1	Gymnast A1	pommel	7	6
3	1	Gymnast A1	rings	9	5
4	1	Gymnast A1	vault	8	6
5	1	Gymnast A1	p-bars	7	5
6	1	Gymnast A1	high bar	8	6

When scrolling



Score ID	Gymnast ID	Name	Apparatus	Execution	Difficulty
7	2	Gymnast A2	floor	7	5
8	2	Gymnast A2	pommel	8	6
9	2	Gymnast A2	rings	7	5
10	2	Gymnast A2	vault	9	6
11	2	Gymnast A2	p-bars	8	5
12	2	Gymnast A2	high bar	7	6
13	3	Gymnast A3	floor	8	5

I also removed the deletion form and added buttons to each of the rows in the tables for deleting. So now you just need to click the buttons, and then a confirmation message will appear. I needed to create a function for the new delete button, that sorts out all the deletion:
Def delete_gymnast

Registered Gymnasts			
ID	Name	Level	Delete
2	Test 2	1	Delete
3	Test 3	1	Delete
4	Test 4	2	Delete
5	Test 5	2	Delete

- Improved Leaderboard

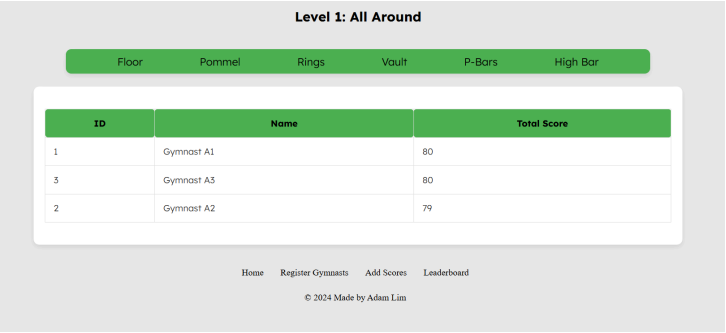
  The previous leaderboard had dropdown buttons for each apparatus to view the corresponding scores. But with this improvement I added links to a corresponding page for each apparatus to view the scores.

  To do this I had to create a new route and function:
  - @app.route("/apparatuslead/<int:level>/<int:apparatus_id>")
  - def apparatus_leaderboard(level, apparatus_id)
  - apparatuslead.html

  This page contains all the apparatus leaderboards while also keeping the level categories.

  I also originally had an sql query for each apparatus, but I combined these into one sql statement.

  Later on I actually changed the design of the leaderboard, this is what it looks like.

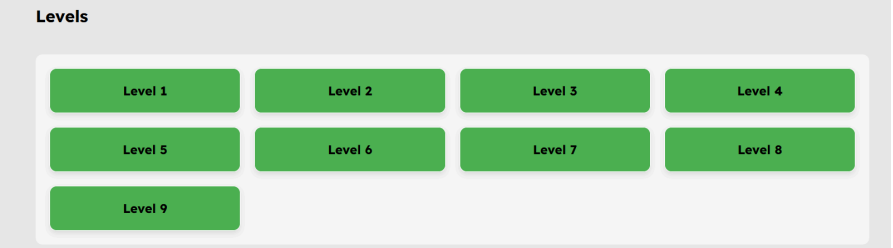


- Level Section

  In New Zealand gymnastics, there are different levels to help athletes progress. I have added level categories for competition.
  Because of the new level category, I have to add a new column to the gymnast table in my database:

gymnast		
gymnast_id	INT	PK
gymnast_name	TEXT	
Level	INT	

I styled the links so they are readable while keeping them aesthetically pleasing.



I have also needed to modify and add SQL queries for this improvement. Due to this new category I had to modify the leaderboard. I created a page with links so then you can select which level leaderboard you want to view. To do this I had to create 2 new route, function and html files:

- @app.route("/scoredata")
- def scoredata()
- scoredata.html

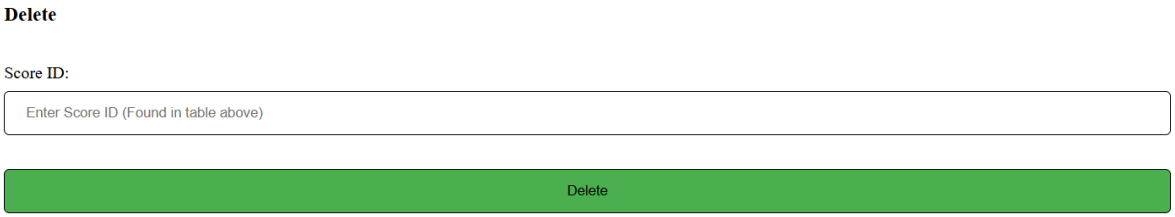
This page contains the links for each level.

- @app.route("/scorelead/<int:level>")
- def score_leaderboard(level)
- scorelead.html

This page contains the all around leaderboard and links to the other apparatuses.

- Other

I created a new form on both the register gymnast and add scores pages. This form allows you to delete the gymnast (which will also delete corresponding existing scores) and scores.



I also added placeholders to all of the forms. These placeholders are text that appear on the input boxes. I used them to give a small idea for the user of what they are supposed to be inputting.

The code for my website was quite messy and repetitive. So i cleaned up the cleaned up the code and created a function for connecting to the database and getting data from the database.

```
def db_query(query_string, params=(), single=True, commit=False):
    conn = sqlite3.connect("database.db")
    cur = conn.cursor()
    cur.execute(query_string, params)

    if single:
        result = cur.fetchone()
    else:
        result = cur.fetchall()
    if commit:
        conn.commit()

    conn.close()
    return result
```

- Final SQL queries

  I have added and changed sql queries throughout the project, this is the final set of SQL queries.

What for?	SQL Entries + Explanation
Adding gymnast with their level	INSERT INTO gymnast (gymnast_name, level) VALUES (?, ?) *This SQL query is used to add data into the gymnast table. It asks for the gymnast_name and level, which gets added to the database.*
Get gymnasts	SELECT * FROM gymnast *This SQL query selects everything from the gymnast table*
Checking if gymnast id exists	SELECT * FROM gymnast WHERE gymnast_id = ? *This SQL query selects everything from the gymnast table based on the gymnast_id given.*
Delete score depending on gymnast	DELETE FROM score WHERE gymnast_id = ? *This SQL query deletes everything from the table score based on the gymnast_id given.*
Delete gymnast	DELETE FROM gymnast WHERE gymnast_id = ? *This SQL query deletes everything from the table gymnast based on the gymnast_id given.*
Delete score	DELETE FROM score WHERE score_id = ? *This SQL query deletes everything from the table score based on the score_id given.*
Edit gymnast	UPDATE gymnast SET gymnast_name = ?, level = ? WHERE gymnast_id = ? *This SQL query updates the gymnast table by setting the gymnast_name and level to new or same values based on the gymnast_id given.*
Viewing gymnast	SELECT * FROM gymnast *This SQL query selects everything in the gymnast table*
Adding scores	INSERT INTO score (gymnast_id, apparatus_id, escore, dscore) VALUES (?,?,?,?) *This SQL query is used to add data into the score table. It asks for the gymnast_id, apparatus_id, escore and dscore, which gets added to the database.*
Viewing scores	SELECT score_id, score.gymnast_id, gymnast.gymnast_name,

	apparatus.apparatus_name, escore, dscore FROM score INNER JOIN gymnast ON score.gymnast_id=gymnast.gymnast_id INNER JOIN apparatus ON score.apparatus_id=apparatus.apparatus_id *This SQL query retrieves data from the table score, gymnast and apparatus. It selects the score_id, gymnast_id from the score table, gymnast_name from the gymnast table, and both escore and dscore from the score table. The query uses INNER JOIN to combine these tables based on gymnast_id and apparatus_id.*
Checking if score_id exists	SELECT * FROM score WHERE score_id = ? *This SQL query selects everything from the table score based on the score_id.*
Edit scores	UPDATE score SET apparatus_id = ?, escore = ?, dscore = ? WHERE score_id = ? *This SQL query updates the score table by setting the apparatus_id, escore and dscore to new or same values based on score_id given.*
Get gymnast levels	SELECT * FROM (SELECT gymnast.*, ROW_NUMBER() OVER (PARTITION BY gymnast.level ORDER BY gymnast.gymnast_id) AS num FROM gymnast JOIN score ON gymnast.gymnast_id = score.gymnast_id) AS ranked_gymnasts WHERE num = 1 ORDER by level *This SQL query retrieves the first gymnast from each level, based on their gymnast_id. It does this by first joining the gymnast and score tables on gymnast_id, then assigning a unique row number to each gymnast within the same level using the Row_number() function. The outer query then selects all columns from this result set where the row number is 1. Filtering to the first gymnast in each level. Finally, the results are ordered by gymnast level.*
Get apparatus	SELECT DISTINCT apparatus.apparatus_id, apparatus.apparatus_name FROM apparatus JOIN score ON apparatus.apparatus_id = score.apparatus_id JOIN gymnast ON score.gymnast_id = gymnast.gymnast_id WHERE gymnast.level = ? ORDER BY apparatus.apparatus_id *This SQL query retrieves distinct apparatus IDs and names for gymnasts at a specific level. It joins the apparatus, score and gymnast tables getting the results to only include gymnasts at the specified level. The DISTINCT ensures that each apparatus appears only once in the results. Then the apparatus are ordered by apparatus_id.*

Get all around results	SELECT score.gymnast_id, gymnast.gymnast_name, SUM(score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id WHERE gymnast.level = ? GROUP BY score.gymnast_id, gymnast.gymnast_name ORDER BY total DESC *This SQL query ranks gymnasts by their total scores (sum of dscore and escore) for a specific level, joining the gymnast and score tables on gymnast_id, and orders the results in descending order.*
Get apparatus results	SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore, score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id WHERE score.apparatus_id = ? AND gymnast.level = ? ORDER BY total DESC *This SQL query ranks gymnasts by their total score (sum of dscore and escore) for a specific level and apparatus, joining the gymnast and score tables on gymnast_id and orders the results in descending order.*
Get apparatuses	SELECT apparatus_name FROM apparatus WHERE apparatus_id = ? *This SQL query selects the apparatus name from the apparatus table based on the apparatus_id given*
Get apparatus used	SELECT DISTINCT apparatus.apparatus_id, apparatus.apparatus_name FROM apparatus JOIN score ON apparatus.apparatus_id = score.apparatus_id JOIN gymnast ON score.gymnast_id = gymnast.gymnast_id WHERE gymnast.level = ? ORDER BY apparatus.apparatus_id *This SQL query retrieves distinct apparatus_id and apparatus_name for gymnasts of a specific level, joining the apparatus, score and gymnast tables and orders the results by apparatus_id*

Final Functions:

def page_not_found()
def url_too_long()
def internal_server_error
def home_page
def db_query(query_string, params=(), single = True, commit = False)

def delete_gymnast(delete_id, delete_gymnast=False)
def gymnast()
def scores()
def score_data()
def level_leaderboard(level)
def apparatus_leaderboard(level, apparatus_id)

Implications

Usability

- Usability means that the website needs to be easy to use, for example, easy navigation between pages, etc…

  To help make the website easier to use, I added a navigation bar on the left side of the screen. This allowed the user to click on the links to go to a different page. I also added instructions on the home page to give the user an idea of how to use my website. As a small reminder I have added small instructions on all the forms. I have added drop-down selections for level and apparatus, this will help simplify things for the user, so they do not need to type in like an id or the name, They just need to select one of the given options. For the level selection page, I put the 9 links into a 3 by 3 grid, this will make it easier for the user to see and select a level. I removed the delete form and added buttons to each of the rows in the table for deleting instead. This will make it much easier for the user to find and delete a gymnast if they want to.

Aesthetics

- Aesthetics refers to how the website looks and how easy it is to see things around the site.

  I have used simple colours that look aesthetically pleasing. I have made sure everything is readable. I used the font Lexend. I think this font works well with my website, allowing for easy readability. For tables, I have added scroll bars to help prevent the page from getting too long and confusing. I have made it so when you scroll down the tables, the table headers follow, this will make it easier for the user to see what each column is for, instead of having to keep scrolling back up and down.

Sustainability and future-proofing

- Sustainability and future-proofing refer to a website's load speed to save energy and users' experience. A website that can adapt to future changes.

  My website's load speed is normal. There are no delays in the load speed, which means the user does not need to wait for a long time for their data to be shown in the tables. So when entering gymnasts, almost straight after they submit, they can find their gymnast in a table. My website is constantly future-proofing when a user uses it. When they add scores, the leaderboard gets automatically updated, as the other tables on the website. They all get updated as the user inputs data.

Social

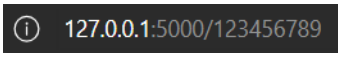
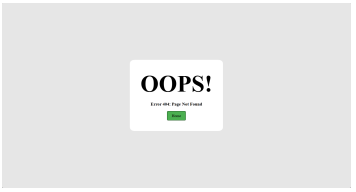
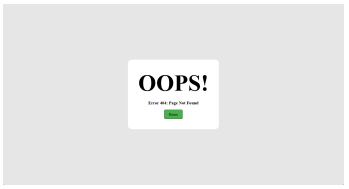
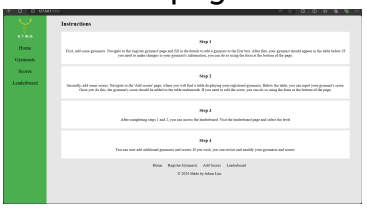
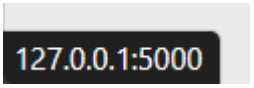
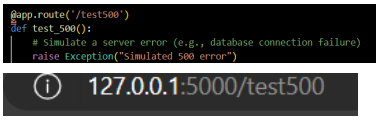
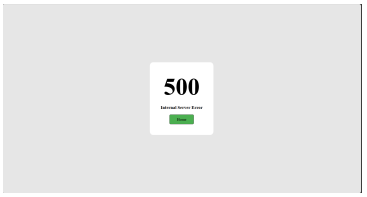
- Your database is important for social reasons because it helps create fairness, openness, community, and inclusivity. It makes sure that every gymnast's performance is recorded in the same way, so everyone is treated equally and fairly in the competition. The way the database is organised makes it easy to see and understand how scores are given, which keeps things clear for gymnasts, coaches, and spectators. By recording each gymnast's name and level, it also helps build a sense of community where athletes can compare scores, learn from each other, and improve. Finally, the database includes gymnasts of all skill levels, making sure everyone's performance is recorded equally, which creates an inclusive environment where every gymnast has the same chance to succeed.

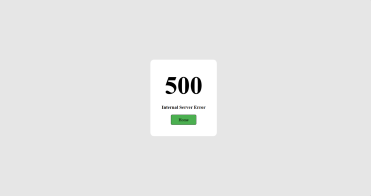
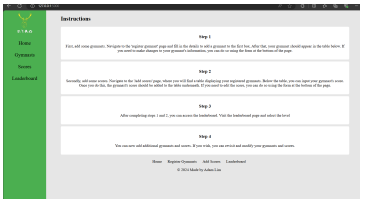
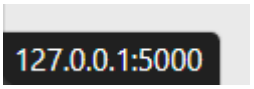
Testing

When I almost completed my testing, I changed the website's look, fonts, table designs, instructions, and leaderboard. So, some of the images may not match the designs. But even though the design has changed, the backend code has not. Everything will function as stated in the testing.

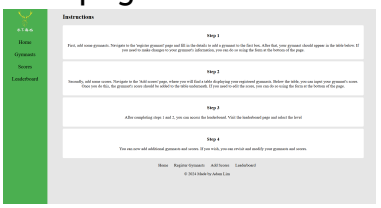
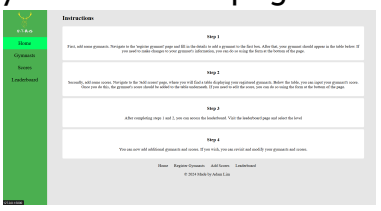
There is no upper bound for all of my forms. Because the level selection is a dropdown selection, the user cannot select any lower or higher. The number of scores and gymnasts can be as long as the user wants.

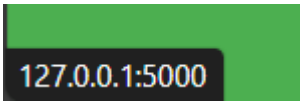
I have created 'test' gymnasts to help show if the program works.

What is being tested?	How?	Did it work?
404 Error	I added random numbers into the web address to test if a 404 page appears if there is an error in the web address. 	Yes, if there is an error in the web address, a 404 page will appear. 
Does the Home button work on the 404 page?	Clicked the homepage button. 	Yes, the button takes you to the homepage.  When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the homepage URL link. 
500 Error	To test whether the 500 error page works, I had to create a route that creates a server error. I then entered the route into the url. 	Yes, if there is a server error, a 500 page will appear. 
Does the Home button work on the 500 page?	I clicked the Home button	Yes, the button takes you to the homepage.

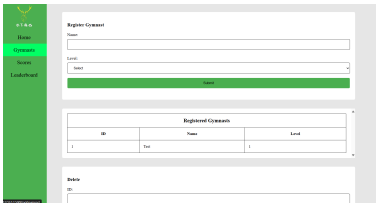
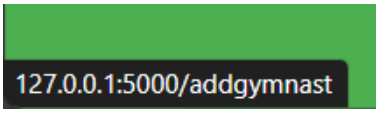
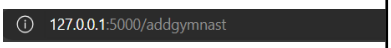
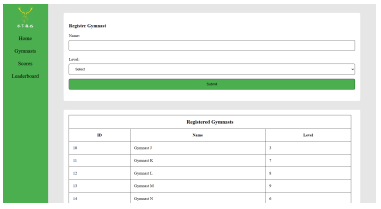
		 When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the homepage URL link. 

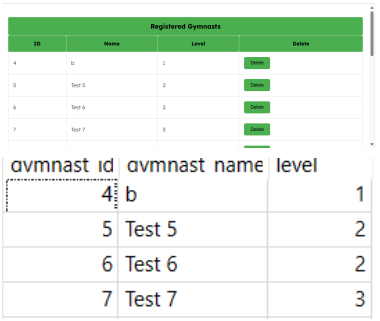
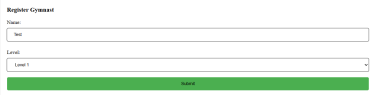
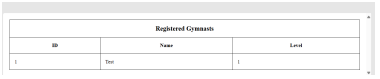
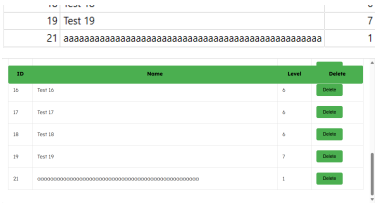
Home Page

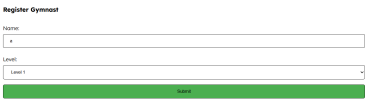
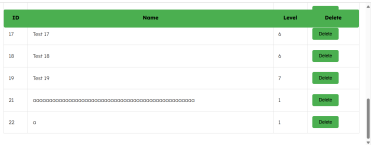
What is being tested	How?	Did it work?
Home page route	I entered the home page route: 	Yes, the homepage route takes you to the homepage. 
Home page navbar link	I clicked the home page navbar link. 	Yes, the navbar link takes you to the homepage.  When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the homepage URL link.

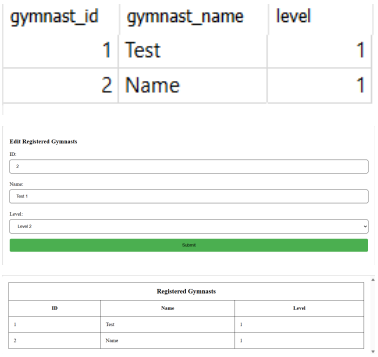
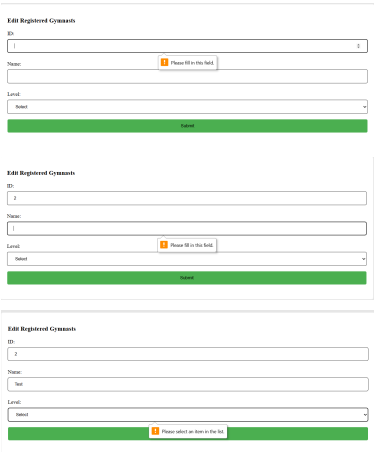
		

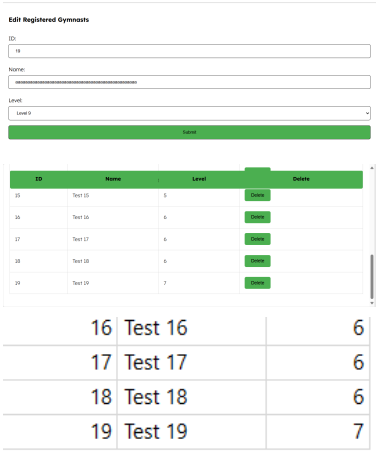
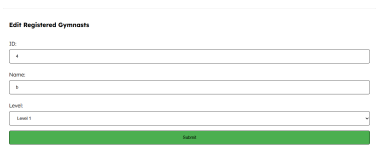
Register Gymnast Page

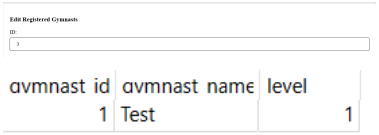
What is being tested?	How?	Did it work?
Gymnast page navbar link	I clicked the gymnast page navbar link: 	Yes, the navbar link takes you to the register gymnast page.  When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the gymnast page URL link. 
Register Gymnast route	I entered the addgymnast route into the URL. 	Yes, the route takes you to the register gymnast page. 
Does the registered gymnasts table show the correct information?	Compared it with the gymnast table in the database.	Yes, the table shows the correct information (i am only showing the first 4 in

		the table. Since these are correct the rest will be:  Registered Gymnasts ID Name Level Delete 4 b 1 Delete 5 Test 5 2 Delete 6 Test 6 2 Delete 7 Test 7 3 gymnast_id gymnast_name level 4 b 1 5 Test 5 2 6 Test 6 2 7 Test 7 3
Register Gymnast Form Does adding gymnasts work?	I registered the gymnast name as Test and level as 1.  Register Gymnast Name: Test Level: Level 1 Submit	Yes, a new gymnast with a level has been added to the database, shown in the table. gymnast_id gymnast_name level 1 Test 1  Registered Gymnasts ID Name Level 1 Test 1
What happens if the register gymnast form is left empty?	I left the form empty.	An error message says the field needs to be filled out. This will prevent any null data in the database.  Register Gymnast Name: Test Level: Select Please select an item in the list.  Register Gymnast Name: Level: Select Please fill in this field. Submit
Upper bound	Testing the max amount of characters in entering the name. I set the max as 50 characters, so the user cannot type any more than 50 characters. I typed 50 a's to test this.  Register Gymnast Name: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa Level: Level 1 Submit	Yes, the gymnasts still gets added to the database and table.  19 Test 19 7 21 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa 1 ID Name Level Delete 16 Test 16 6 Delete 17 Test 17 6 Delete 18 Test 18 6 Delete 19 Test 19 7 Delete 21 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa 1 Delete

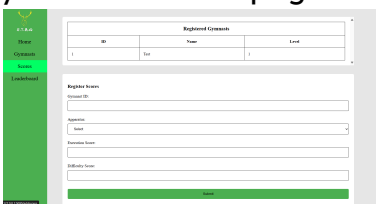
Lower bound	Testing lowest amount of characters in entering the name. I set the min as 1 character, so the user has to type at least 1 character. 	Yes, the gymnast still gets added to the database and table. 
What happens if the name inputted is whitespace?	Indented in the input field. (spacebutton)	An error message says please match the format, just whitespace is not allowed. 
Delete Buttons Does the delete buttons work?	I am going to delete Test 2 with the gymnast ID 2 avmnast id | avmnast name | level 2 | Test 2 | 1 3 | Test 3 | 1 4 | Test 4 | 2 5 | Test 5 | 2 	Yes, the buttons works. The gymnast, Test 2 with the gymnast ID 2 gets removed from the database.  avmnast id | avmnast name | level 3 | Test 3 | 1 4 | Test 4 | 2 5 | Test 5 | 2

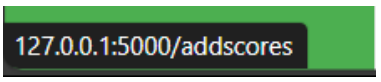
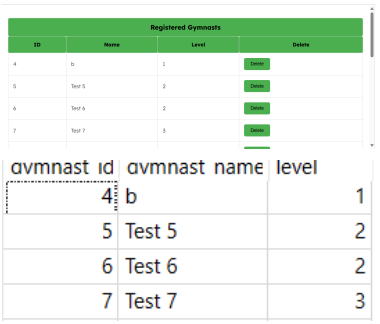
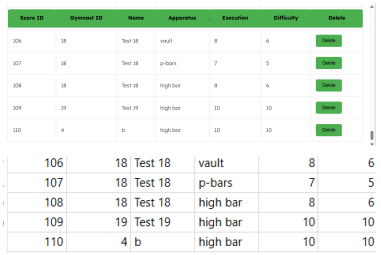
Editing Form Does editing the gymnast name and level work?	I inputted ID 2, changed the corresponding name to “Test 1,” and also changed the level to 2.  	Yes, gymnast name and level changes.  
Does it work if the input fields for ID, Name, and Level are left empty?	I left the input fields empty.	Yes, an error message appears saying that the field must be filled out. This is needed to prevent any null inputs into the database. 

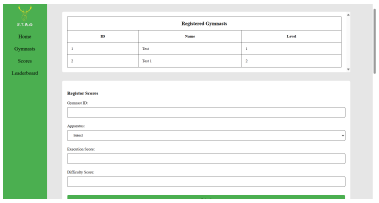
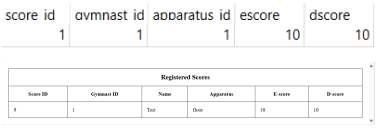
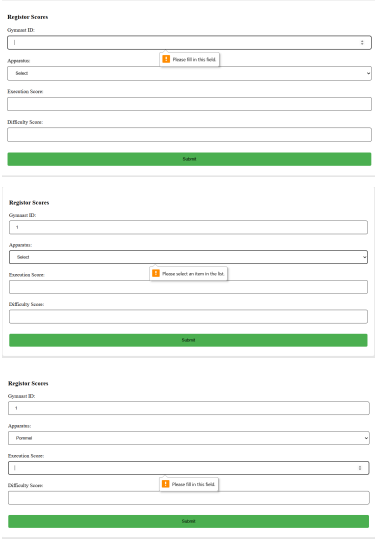
Upper bound of ID, name and level. I added a max length in my code, so the user cannot actually type anymore than 50 characters.	I inputted the highest ID that is in the database which is 19. I inputted 50 a's in the name field, this is the max number of characters allowed and selected level 9, which is the highest level.  16 Test 16 6 17 Test 17 6 18 Test 18 6 19 Test 19 7	Yes, the gymnast with the ID 19, name changed to the 50 a's and the level changed to 9.  16 Test 16 6 17 Test 17 6 18 Test 18 6 19 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa 9
Lower bound of ID, name and level.	I inputted the lowest ID that is in the database which is 4. I inputted 1 b in the name field, this is the min number of characters allowed and selected level 1, which is the lowest level.   avmnast id avmnast name level 4 Test 4 2 5 Test 5 2 6 Test 6 2 7 Test 7 3	Yes, the gymnast with the ID 4, name changed to b and the level changed to 1. avmnast id avmnast name level 4 b 1 5 Test 5 2 6 Test 6 2 7 Test 7 3 

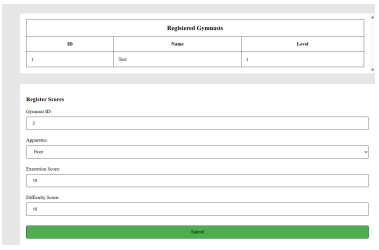
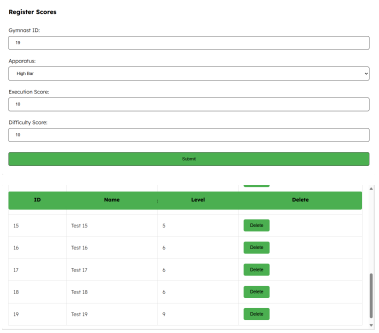
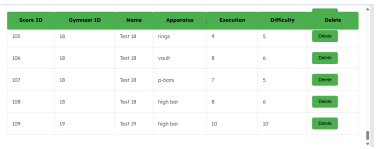
What happens if the name inputted is whitespace?	Indented in the input field. (spacebutton)	An error message says please match the format, just whitespace is not allowed. 
What happens if the ID inputted to edit does not exist?	I inputted the ID 3, which does not exist in the database. 	Nothing changes. An error message appears saying that the ID does not exist. 

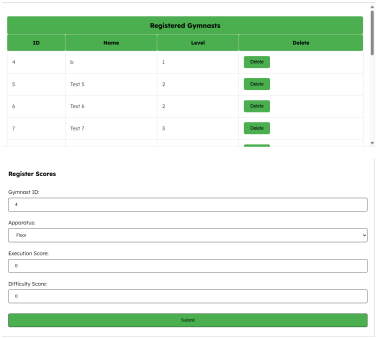
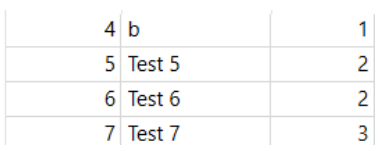
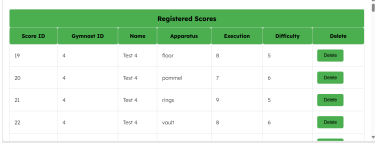
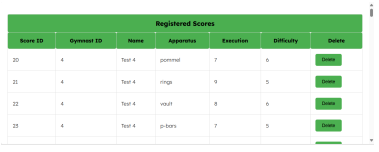
Scores Page

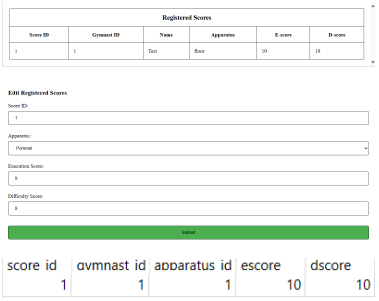
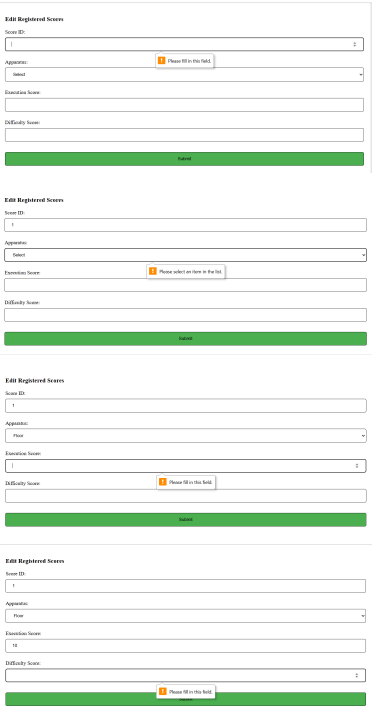
What is being tested?	**How?**	**Did it work?**
Add scores page navbar link.	I clicked on the scores navbar link. 	Yes, the navbar link takes you to the scores page  When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the scores page URL link.

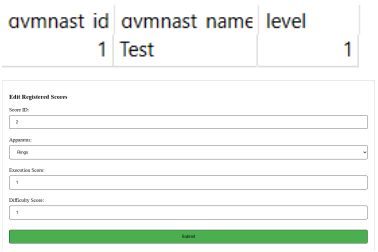
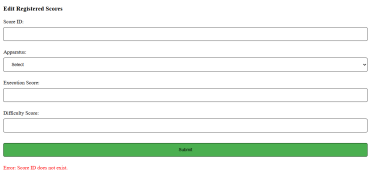
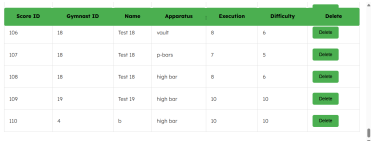
		
Does the registered gymnasts table show the correct information?	Compared it with the gymnast table in the database.	Yes, the table shows the correct information (i am only showing the first 4 in the table. Since these are correct the rest will be:  gymnast_id gymnast_name level 4 b 1 5 Test 5 2 6 Test 6 2 7 Test 7 3
Does the registered scores table show the correct information?	Compared it with a table generated from a SQL query in the database. An SQL query is required because the score table does not hold all the columns that the registered scores table has. Sql: SELECT score_id, score.gymnast_id, gymnast.gymnast_name, apparatus.apparatus_name, escore, dscore FROM score INNER JOIN gymnast ON score.gymnast_id=gymnast.gymnast_id INNER JOIN apparatus ON score.apparatus_id=apparatus.apparatus_id	Yes, the table shows the correct information (I am only showing the last set of 5 scores in the table. Since these are correct the rest will be:  106 18 Test 18 vault 8 6 107 18 Test 18 p-bars 7 5 108 18 Test 18 high bar 8 6 109 19 Test 19 high bar 10 10 110 4 b high bar 10 10

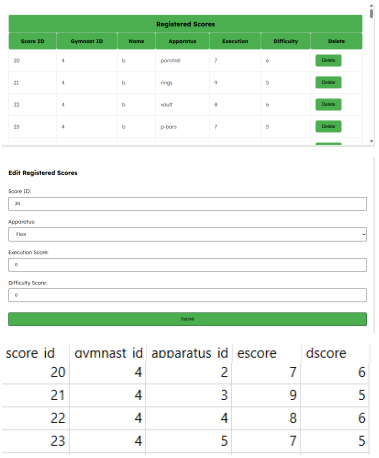
Scores route	I entered the addscores route into the URL. 	Yes, the scores route takes you to the scores page 
Register Score Form Does adding scores work?	I inputted the gymnast with the ID 1, selected floor and gave the gymnast the execution score of 10 and difficulty score of 10. 	Yes, the score gets added to the database and also to a table on the page. 
What happens if the register score form is left empty?	I left the input fields empty.	Nothing gets added to the database. An error message appears telling the user to fill out that field. This is good because I do not want null values in the database. 

		
What happens if the ID inputted does not exist in the database?	I inputted the ID 2 which does not exist in the database. 	Nothing gets added to the database. An error message appears saying that the ID does not exist. 
Upper Bound of ID, apparatus, execution score and difficulty score.	I inputted the gymnast ID 19, which is the highest gymnast ID in the database. I selected high-bar which has the highest ID of 6. I inputted 10 in both execution and the difficulty score, which is the highest score possible.  16 Test 16 6 17 Test 17 6 18 Test 18 6 19 Test 19 9	Yes, the gymnast with the ID 19's execution and difficulty score of 10 in the high bar category gets added to the database.  106 18 4 8 6 107 18 5 7 5 108 18 6 8 6 109 19 6 10 10
Lower bound of ID, apparatus, execution	I inputted the gymnast ID 4, which is the lowest	Yes, the gymnast with the ID 4's execution and

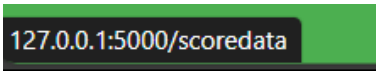
score and difficulty score.	gymnast ID in the database. I selected floor which has the lowest ID of 1. I inputted 0 in both execution and the difficulty score, which is the lowest possible score.   4 b 1 5 Test 5 2 6 Test 6 2 7 Test 7 3	difficulty score of 0 in the floor category gets added to the database.  107 18 5 7 5 108 18 6 8 6 109 19 6 10 10 110 4 1 0 0
Delete Buttons - Registered gymnast table - Score table	Clicked the delete button for the gymnast with the ID 3. avmnast id avmnast name level 3 Test 3 1 4 Test 4 2 5 Test 5 2 6 Test 6 2  Clicked the delete button for the score with the score id 19 score id avmnast id apparatus id escore dscore 19 4 1 8 5 20 4 2 7 6 21 4 3 9 5 22 4 4 8 6 	Yes, the buttons works, the gymnast, Test 3 with the gymnast ID 3 gets removed from the database.  avmnast id avmnast name level 4 Test 4 2 5 Test 5 2 6 Test 6 2 7 Test 7 3 Yes, the buttons works, the score with the score ID 19 gets removed from the database. 

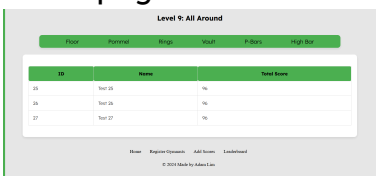
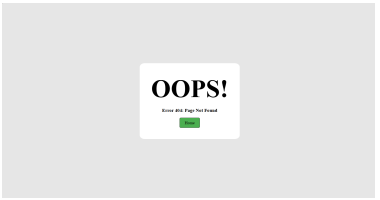
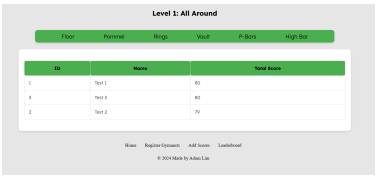
		score id | gymnast id | apparatus id | escore | dscore 20 | 4 | 2 | 7 | 6 21 | 4 | 3 | 9 | 5 22 | 4 | 4 | 8 | 6 23 | 4 | 5 | 7 | 5
Editing Form Does editing the gymnast's scores work?	I inputted the ID 1 and changed the apparatus to pommel, execution and difficulty score to 9.  score id | gymnast id | apparatus id | escore | dscore 1 | 1 | 1 | 10 | 10	Yes, the gymnast with the ID 1's score apparatus changed to pommel, the execution and difficulty score to 9. score id | gymnast id | apparatus id | escore | dscore 1 | 1 | 2 | 9 | 9 
What happens if the input fields are left empty?	I left the input fields empty.  	Nothing happens, and an error message appears saying to fill out the field. This is good because this prevents any null values getting added to the database 

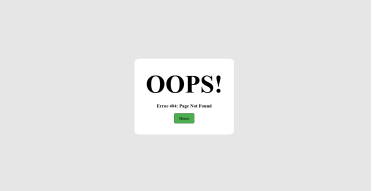
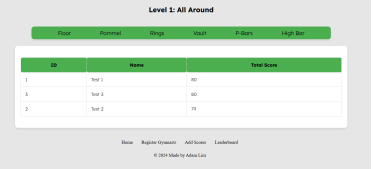
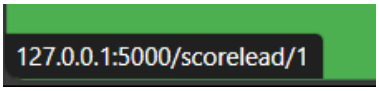
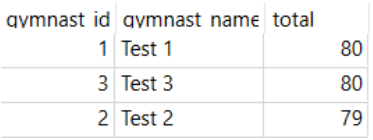
What happens if the ID does not exist?	I have inputted the ID 2, which does not exist in the database.  gymnast id | gymnast name | level 1 | Test | 1	Nothing happens, An error message appears saying that the ID does not exist. 
Upper bound of score ID, apparatus, execution score and difficulty score.	I inputted the score ID 110, which is the highest score ID in the database. I selected high bar which has the highest ID of 6. I inputted 10 in both execution and the difficulty score, which is the highest possible score.   107 | 18 | 5 | 7 | 5 108 | 18 | 6 | 8 | 6 109 | 19 | 6 | 10 | 10 110 | 4 | 1 | 0 | 0	Yes, the score ID 110's apparatus changed to high bar and their execution score and difficulty score changed to 10.  107 | 18 | 5 | 7 | 5 108 | 18 | 6 | 8 | 6 109 | 19 | 6 | 10 | 10 110 | 4 | 6 | 10 | 10
Lower bound of score ID, apparatus, execution score and difficulty score.	I inputted the score ID 20, which is the lowest score ID in the database. I selected floor which has the lowest ID of 1. I inputted 0 in both	Yes, the score ID 20's apparatus changed to floor and their execution score and difficulty score changed to 0.

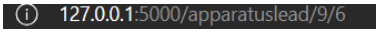
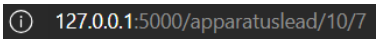
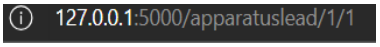
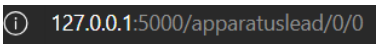
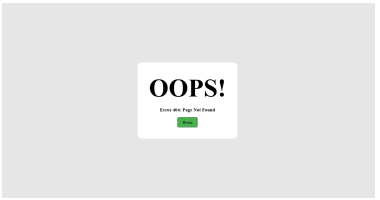
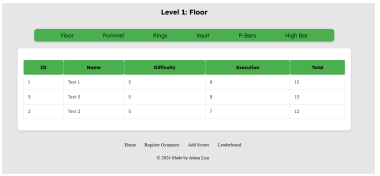
	execution and the difficulty score, which is the lowest possible score.  Registered Scores Score ID / Gymnast ID / Name / Apparatus / Execution / Difficulty / Delete 20 4 b pommel 7 6 Delete 21 4 b rings 9 5 Delete 22 4 b vault 8 6 Delete 23 4 b p-bars 7 5 Delete Edit Registered Scores Score ID: Apparatus: Execution Score: Difficulty Score: Submit score id / gymnast id / apparatus id / escore / dscore 20 4 2 7 6 21 4 3 9 5 22 4 4 8 6 23 4 5 7 5	 Registered Scores Score ID / Gymnast ID / Name / Apparatus / Execution / Difficulty / Delete 20 4 b floor 0 0 Delete 21 4 b rings 9 5 Delete 22 4 b vault 8 6 Delete 23 4 b p-bars 7 5 Delete score id / gymnast id / apparatus id / escore / dscore 20 4 1 0 0 21 4 3 9 5 22 4 4 8 6 23 4 5 7 5

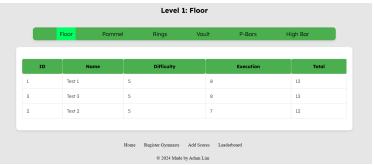
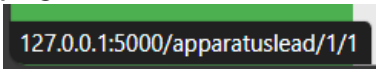
Leaderboard Page

What is being tested?	How?	Did it work?
Leaderboard navbar link	I clicked the leaderboard navbar link.	Yes, the navbar link takes you to the leaderboard page, where you can select a level.  Levels Level 1 Level 2 Level 3 Level 4 Level 5 Level 6 Level 7 Level 8 Level 9 Home Gymnasts Scores Leaderboard When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the level selection page URL link.  127.0.0.1:5000/scoredata
Leaderboard route	I entered the scoredata route into the URL.	Yes, the scoredata route takes you to the levels

	127.0.0.1:5000/scoredata	option page. 
Upper bounds and lower bounds of level selection in the scorelead route. Upper bound: - Level 9 - Level 10 Lower bound - Level 1 - Level 0	I entered the level as 9 in the URL. 127.0.0.1:5000/scorelead/9 I entered the level as 10 in the URL. 127.0.0.1:5000/scoredata/10 I entered the level as 1 in the URL. 127.0.0.1:5000/scorelead/1 I entered the level as 0 in the URL. 127.0.0.1:5000/scorelead/0	Yes, the highest level, 9 works. It takes you to the level 9 page.  No, Level 10 does not work, because there is no level that can be higher than 9. A 404 page appears.  Yes, the lowest level, 1 works. It takes you to the level 1 page.  No, Level 0 does not work because there is no level 0, there can not be any level lower than 1. A 404 page appears.

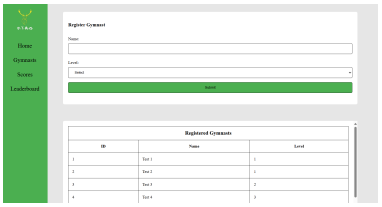
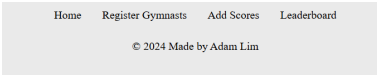
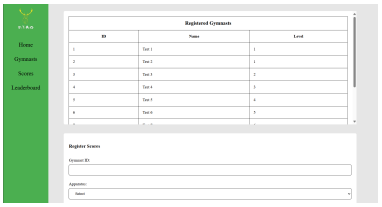
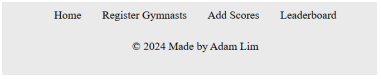
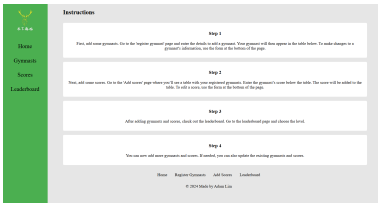
		
Levels link Does this show the correct information?	I selected level 1  In this example I selected level 1. To show that the data is showing the category level 1, I have used an SQL query. SELECT score.gymnast_id, gymnast.gymnast_name, SUM(score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id WHERE gymnast.level = 1 GROUP BY score.gymnast_id, gymnast.gymnast_name ORDER BY total DESC This SQL statement gets the total scores from the category level 1.	Yes, each level link takes you to the corresponding level leaderboard page. I will only show the first level because if it works, all the others will also work.  When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the level 1 page URL link.  Yes, the level 1 link shows the scores of gymnasts who are level 1.  gymnast id | gymnast name | total 1 | Test 1 | 80 3 | Test 3 | 80 2 | Test 2 | 79

Upper bounds and lower bounds of apparatus selection in the apparatus lead route. Do they work, and do they show the correct information? Upper bound: - Level 9 with the apparatus ID of 6 - Level 10 with the apparatus ID of 7 Lower bound - Level 1 with the apparatus ID 1 - Level 0 with the apparatus ID of 0 -	I entered the level as 9 and apparatus ID as 6 in the URL.  I entered the level as 10 and apparatus ID as 7 in the URL.  I entered the level as 1 and apparatus ID as 1 in the URL.  I entered the level as 0 and apparatus ID as 0 in the URL.  To show that the information shown on each page, I am using an sql query that gets the scores based on the level and the apparatus ID. I replaced the ? with the corresponding level and apparatus ID. SELECT score.gymnast_id, gymnast.gymnast_name, score.dscore, score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id WHERE score.apparatus_id = ? AND gymnast.level = ? ORDER BY total DESC	Yes, the ‘highest’ apparatus route of 6, high bar, in the level 9 category, works. It takes you to the leaderboard that displays high bar under the category level 9. The information is correct as it matches the database info.  gymnast_id | gymnast_name | dscore | escore | total 25 | Test 25 | 8 | 8 | 16 26 | Test 26 | 8 | 8 | 16 27 | Test 27 | 8 | 8 | 16 No, the apparatus ID 7 with the level 10 takes you to a 404 page. This is because the highest apparatus ID is 6 and the highest level is 9.  Yes, the lowest apparatus ID 1 with the level 1 works, it takes you to the leaderboard that displays floor under the category level 1. The information is correct as it matches the database info. 

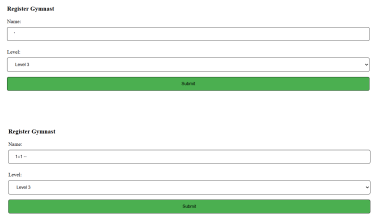
		gymnast_id | gymnast_name | dscore | escore | total 1 | Test 1 | 5 | 8 | 13 3 | Test 3 | 5 | 8 | 13 2 | Test 2 | 5 | 7 | 12 No, the apparatus ID 0 with the level 0 takes you to a 404 page. This is because the lowest apparatus ID is 1 and the lowest level is 1. 
Do each apparatus leaderboard links work?  Does the link show the correct information?	In this example: does the table show the floor scores. I used an sql query to show the scores for level 1, floor. SELECT score.gymnast_id, apparatus.apparatus_name, gymnast.gymnast_name, score.dscore, score.escore, (score.dscore + score.escore) AS total FROM gymnast JOIN score ON score.gymnast_id = gymnast.gymnast_id JOIN apparatus ON score.apparatus_id = apparatus.apparatus_id WHERE score.apparatus_id = 1 AND gymnast.level = 1 ORDER BY total DESC;	Yes, each of the links takes you to the corresponding apparatus and level. I am only showing one apparatus because if it works, then all others should work as well.  When hovering over the link, this appears at the bottom left corner, showing where the link takes you. This is the apparatus ID 1 (floor) page URL link.  gymnast_ic | apparatus | gymnast_n | dscore | escore | total 1 | floor | Test 1 | 5 | 8 | 13 3 | floor | Test 3 | 5 | 8 | 13 2 | floor | Test 2 | 5 | 7 | 12

	This SQL query gets the scores from the scores table, the apparatus name from the apparatus table.	

Footer

What is being tested?	How?	Did it work?
Does the register gymnast footer link work?	I clicked the register gymnast footer link. 	Yes, it takes you to the register gymnast page. 
Does the add scores footer link work?	I clicked the add scores footer link. 	Yes, it takes you to the add scores page. 
Does the Leaderboard footer link work?	I clicked the leaderboard footer link. 	Yes, it takes you to the leaderboard page, where you can select the levels. 
Does the home footer link work?	I clicked on the home footer link. 	Yes, it takes you to the home page. 

Other

What is being tested?	How?	Did it work?
SQL injection	I prevented this by adding a question mark in the SQL queries that require input. For example: SELECT * FROM gymnast WHERE id = ?.  I tested this by inputting things like ‘ and 1=1 –.	Nothing happens. The program works like usual. 

Feedback

Name/feedback	How I have acted with the feedback
Vishnu - Programmer - Having to keep flipping through pages to view instructions.	I will figure out how to improve my website so you don’t need to keep flipping the page to view the instructions. However, due to time constraints, I cannot do this now. So, I will continue to develop this in the future.
Akira - Programmer - Add flash messages, for example when an input was successful. - Style the welcome message.	I will figure out how to improve my website so that there are better and improved flash messages. However, due to time constraints, I cannot do this. So, I will continue to develop this in the future. I took in the styling of the welcome message by making this bigger and bolding the text.
Michael - Gymnast - Change the style of the level selection into a 3 by 3 grid. - Delete buttons	I applied both of these feedbacks. I replaced the delete form with delete buttons and changed the style of the level selection into a 3 by 3 grid.

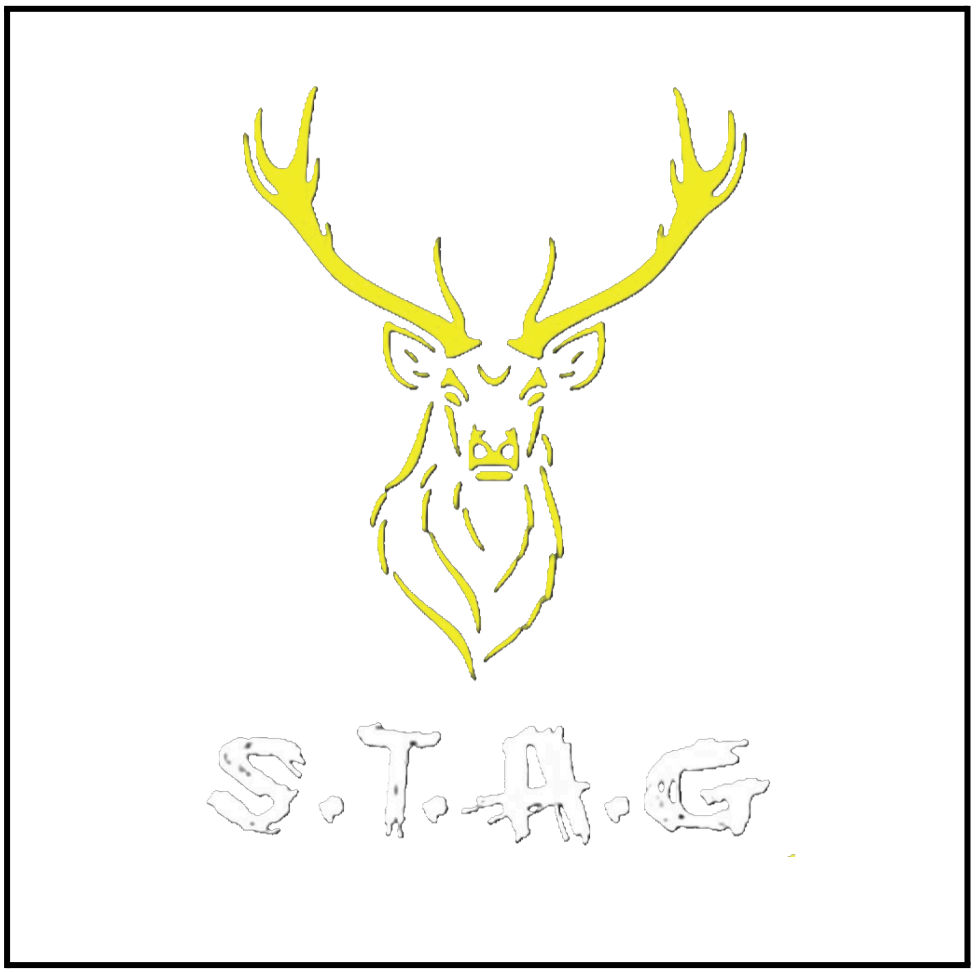
Charlie - Gymnast - Can create new competitions - Search bar to search a gymnast and see their record from all of the competitions.	I won't be able to apply this due to time constraints, but I will definitely apply this in the future

Conclusion

In conclusion I would have made this website if I did this again. I would improve the form process by a lot, I would have liked to have had a step by step process for the forms. I would have added a sign up and login. I would have done the feedback I could not do due to time. To finish I cleared out the data in the score and gymnast tables because in a competition the score and gymnast tables would start out empty.

Thanks to the coach who let me use his logo for my website.

Logo:



End of documentation