Conner Marchell
Alexander Lubrano
CS 340 - Summer 2022

Portfolio Assignment

Step 5

HouBytes Team Members Names

Conner Marchell
Alexander Lubrano

Project Title

Golf Tracker

Website Link

http://flip3.engr.oregonstate.edu:15432/

Please use either Mozilla Firefox or Microsoft Edge to test the application.

Executive Summary

Houbyte's is pleased to present Golf Tracker - a web application for tracking golf data. Golf Tracker was built on a Jinja, HTML, and CSS frontend, with a Python-based Flask backend, all connected to a cloud-hosted MySQL relational database for persistent data storage. Golf Tracker was developed and designed iteratively with multiple rounds of drafts and reviews from peers and collegiate staff.

When initially planning our project in our course's Project Step 1, the team was quick to choose a golf-themed app but had a lot of debate on the overall goals of the app. We asked ourselves questions like: How would it be used? Who would use it? What problems would it solve? As we brainstormed and discussed, our goals became more clear and more refined. These goals are now highlighted in the project outline below. In this initial planning step, we also created an outline for our database. We planned out the entities we wanted to store as tables as well their attributes and their relationships with each other. We made an entity-relationship diagram to visualize these entities, attributes, and relationships. Feedback from our peers helped us in this step by encouraging us to add more statistics and details to the outline and being more consistent with wording across the board.

As we got further into our Project Steps 2, we continued to iterate and improve our database outline, as well as eventually creating a schema diagram to further visualize the entirety of the database. Using our database outline, entity relationship diagram, and schema diagram we were able to create a Data Definition Query (DDQ) file that allowed us to generate a working pass of the database and populate it with example data.

Midway through the development process in Project Step 3, we created Jinja template files for each entity with basic HTML including static tables of data and static forms for implementing Create, Read, Update, and Delete (CRUD) operations in the future. Each of those entity Jinja templates was routed through a Flask application that implemented the routes for our web app. We also created a Data Manipulation Query file that contained initial versions of our CRUD SQL queries that we may use in the project. Peer feedback was critical for Steps 2 and 3 and helped us fix bugs in our DDQ SQL queries and clean up any inconsistencies in all areas of the project.

Toward the end of the project, we began implementing more of the front-end user interface with HTML and CSS then tying it to the backend with Flask for Project Step 4. This process took time but we eventually figured out how to get all of the project's pieces talking to each other and were able to implement all CRUD operations for our players table. Feedback on this step helped find typos and other inconsistencies that popped up. Our peers also gave us ideas for visual and usability improvements.

Now, Golf Tracker is complete and contains all features that we planned to implement for the scope of this course. We've added CRUD operations for every entity, and the capability to search on the players_clubs page. The app runs well and displays live data from a persistent database.

Project Outline

Problem to be solved by web & db

Golf is a popular sport with over 66 million players golfing at 38,864 golf courses around the world. If this is any testament to just how popular of a sport golf is, there were approximately 518 million rounds played in 2021 alone. One of the more popular places to play golf is the United States, which by itself is home to over 16,700 courses played by almost 25 million US players and countless travelers from other countries. With all of these players, courses, and rounds of golf comes a large amount of information to track such as swing counts, scores, course details, details on types of golf clubs used, and many other statistics.

Commonly, golf players track their swing count, also called strokes, on a course-provided piece of paper called a scorecard. This scorecard is fairly standard across courses but does not tie to anything to provide a more global view, it reflects only a minimal summary of a player's round and then is immediately forgotten when the round is complete. If a golf player wants to track their progress over time, they must go out of their way to keep track of all of their scorecards and log all of their rounds' statistics off the course.

Regarding rounds of golf, each round is characterized by a score that reflects the total number of swings a player made in the round. The goal is to have as low a swing count as possible, which heavily depends on a player's skill. Every course will have a number called par, which is the number of swings a proficient player should need to play the entire course, usually 71 or 72 strokes for 18-hole courses. Obtaining a score at or below par is not an easy task and is typically only reached by professional players. The average golf player scores over 100, meaning they swing over 100 times in an 18-hole round of golf. With all of these swings comes even more complexity in relation to the clubs that players can use.

PGA rules dictate that players can have at most 14 clubs in their bag. Even though players are limited to these 14 clubs, there are over 1000 different clubs being sold at major golf stores this year and countless others since modern golf's beginnings in Scotland in the 1400s. Of course, most of the clubs being used today were made in the past 30 to 40 years, but the fact remains that there are a large number of clubs that players might choose from.

Due to the enormous amount of people, places, details, and statistics involved in the game of golf, a more persistent and convenient way to view and store this information is through the use of technology, namely web applications, and databases. Our web application and database for the game of golf allows *players* to track not only their scores on the *course*, but also track career stats on every *swing*, for every *club*, for every *hole*, and on every *course* in our database. This allows players to see their score progression over time, analyze trends and make data-informed decisions on how to better perform on the course.

List of numerical facts

- There are 38,864 golf courses in the world, 16,753 in the United States (Golf Monthly)
- There were 518 million rounds of golf played in 2021 (Turfnet)
- 66.6 Million golf player in the world (Golf Course Industry)
- 24.8 Million golf player in the US (CNBC)
- A golfer is allowed to carry up to 14 clubs in their bag (PGA)
- Average golf player swing over 100 times per 18 hole round (PGA)
- The PGA Tour SuperStore is currently selling 1,000+ different clubs (PGA SuperStore)
- Modern golf originated in the 15th Century in Scotland (Historic UK)

Database Outline

- **players**: Records the details of all players using the application
    - Attributes:
        - player_id: int, auto_incremented, unique, not NULL, PK
        - player_name: varchar(50), not NULL
        - player_city: varchar(50), not NULL
        - player_state: varchar(50), not NULL
    - Relationships:
        - 1:M relationship with rounds with player_id being the FK in rounds. Players would be mandatory, rounds would be optional. Deletions of players cascade to delete rounds.
        - 1:M relationship with swings with player_id being the FK in swings. Players would be mandatory, rounds would be optional.
        - 1:M relationship with player_clubs with player_id being the FK in player_clubs. Players would be mandatory, player_clubs would be optional. Deletion of players cascades to delete corresponding rows in the player_clubs table.
- **clubs**: Records details about available clubs
    - Attributes:
        - club_id: int, auto_incremented, unique, not NULL PK
        - brand: varchar(50), not NULL
        - club_name: varchar(50), not NULL
        - club_type: varchar(50), not NULL
    - Relationships:
        - 1:M relationship with player_clubs with club_id being the FK in player_clubs. Both are optional. Deletion of clubs cascades to delete corresponding rows in the player_clubs table.
        - 1:M relationship with swings with club_id being the FK in swings. Clubs would be mandatory, swings would be optional.
- **player_clubs**: Intersection table between the M:M relationship of players and clubs in the database. Records M:M relationship where players can own multiple clubs and clubs can be owned by multiple players.
    - Attributes:
        - player_id: int, not NULL, FK
        - club_id: int, not NULL, FK
    - Relationships:
        - 1:M relationship with players with player_id being the joining field for players. Both would be optional. Players can exist in the database but not have clubs yet. Delete of players cascades to delete player_clubs intersection rows.
        - 1:M relationship with clubs with club_id being the joining field for clubs. Both would be optional. Clubs exist in the database without being connected to a player. Delete of clubs cascades to delete player_clubs intersection rows.
- **rounds**: Records the details of a player's round of golf on a course (transaction table)
    - Attributes:
        - round_id: int, auto_incremented, unique, not NULL PK

        - course_id: int, not NULL, FK
        - player_id: int, not NULL, FK
        - round_date: datetime, not NULL
        - round_score: int, not NULL
    - Relationships:
        - 1:M relationship with courses with course_id being the FK in rounds. Courses would be mandatory, rounds would be optional. Deletions of courses cascade to delete rounds.
        - 1:M relationship with players with player_id being the FK in rounds. Players would be mandatory, rounds would be optional. Deletions of players cascade to delete rounds.
        - 1:M relationship with swings with round_id being the FK in swings. Rounds would be mandatory, swings would be optional. Deletions of rounds cascade to delete swings.

- **swings**: Records each swing of a player's round on a course (transaction table). Swings can be recorded without a club to capture penalty strokes.
    - Attributes:
        - swing_id: int, auto_incremented, not NULL, PK
        - hole_id: int, not NULL, FK
        - round_id: int, not NULL, FK
        - player_id: int, not NULL, FK
        - club_id: int, FK **(NULL allowed)**
        - dist_traveled_yd: int, not NULL
    - Relationships:
        - 1:M relationship with holes with hole_id being the FK in swings. Holes would be mandatory, swings would be optional.
        - 1:M relationship with rounds with round_id being the FK in swings. Rounds would be mandatory, swings would be optional. Deletions of rounds cascade to delete swings.
        - 1:M relationship with players with player_id being the FK in swings. Players would be mandatory, swings would be optional.
        - 1:M relationship with clubs with club_id being the FK in swings. Deletion of clubs would set club_id in swing to NULL.

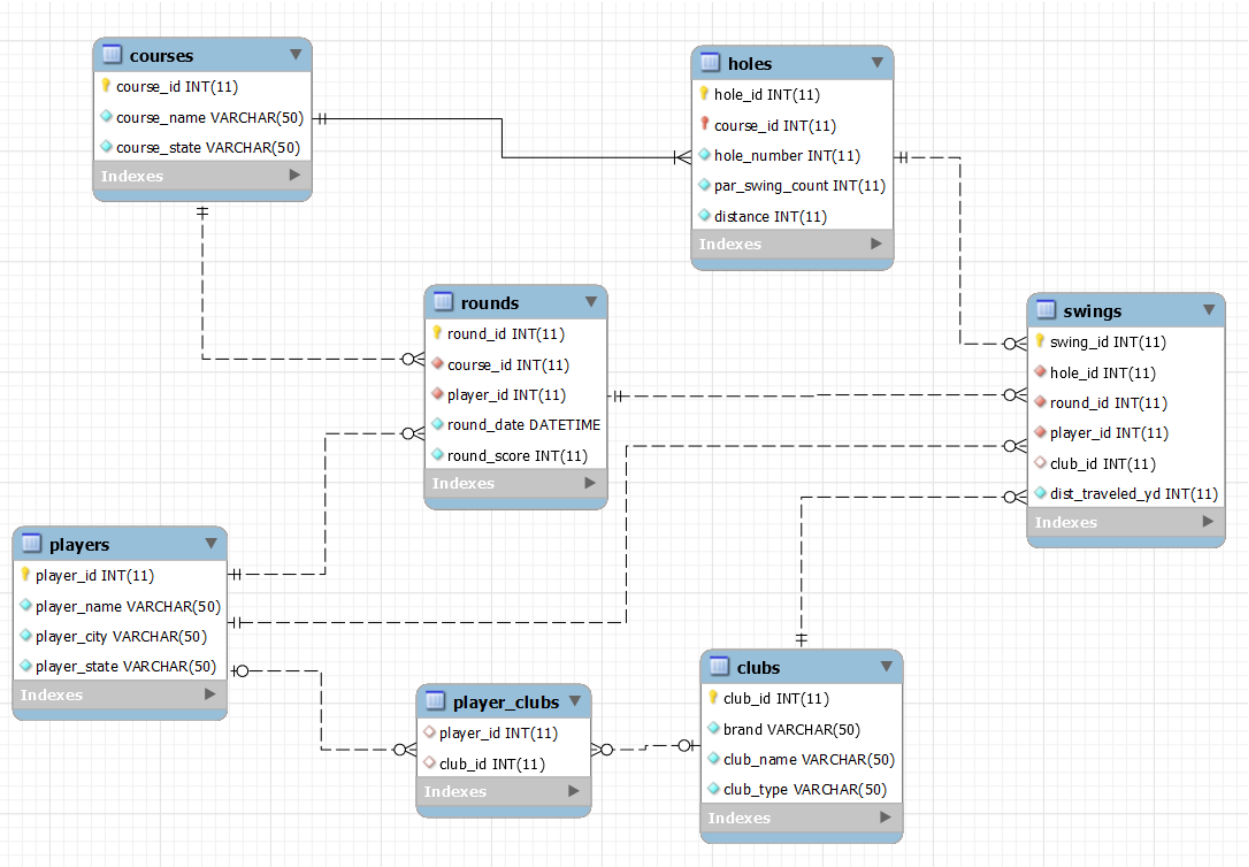
- **courses**: Records the details of all courses tracked in the application
    - Attributes:
        - course_id: int, auto_incremented, unique, not NULL, PK
        - course_name: varchar(50), not NULL
        - course_state: varchar(50), not NULL
    - Relationships:
        - 1:M relationship with holes with course_id being the FK in holes. Both would be mandatory.
        - 1:M relationship with rounds with course_id being the FK in rounds. Courses would be mandatory, rounds would be optional. Deletions of courses cascade to delete rounds.

- **holes**: Records the details for all holes on all courses tracked in the application
    - Attributes:
        - hole_id: int, auto_incremented, not NULL, PK
        - course_id: int, not NULL, PK, FK
        - hole_number: int, not NULL

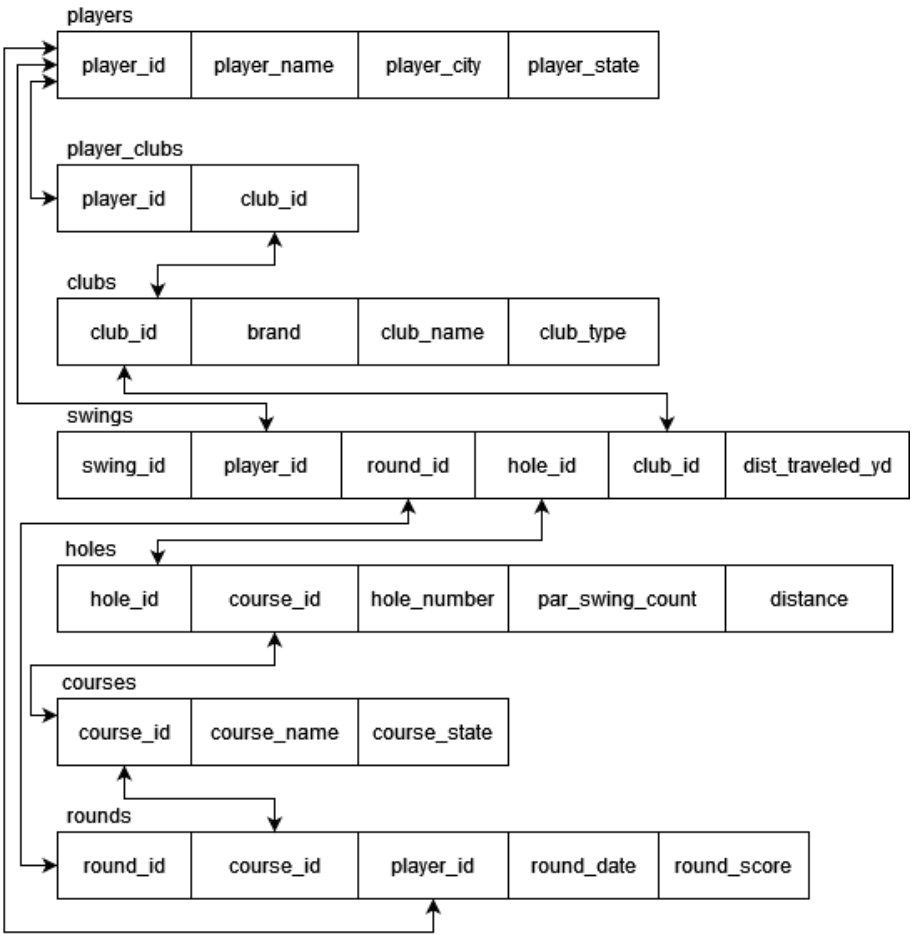
- par_swing_count: int, not NULL
- distance: int, not NULL

- Relationships:
  - 1:M relationship with courses with course_id being the FK in holes. Both would be mandatory.
  - 1:M relationship with swings with hole_id being the FK in swings. Holes would be mandatory, swings would be optional.

Entity-Relationship Diagram

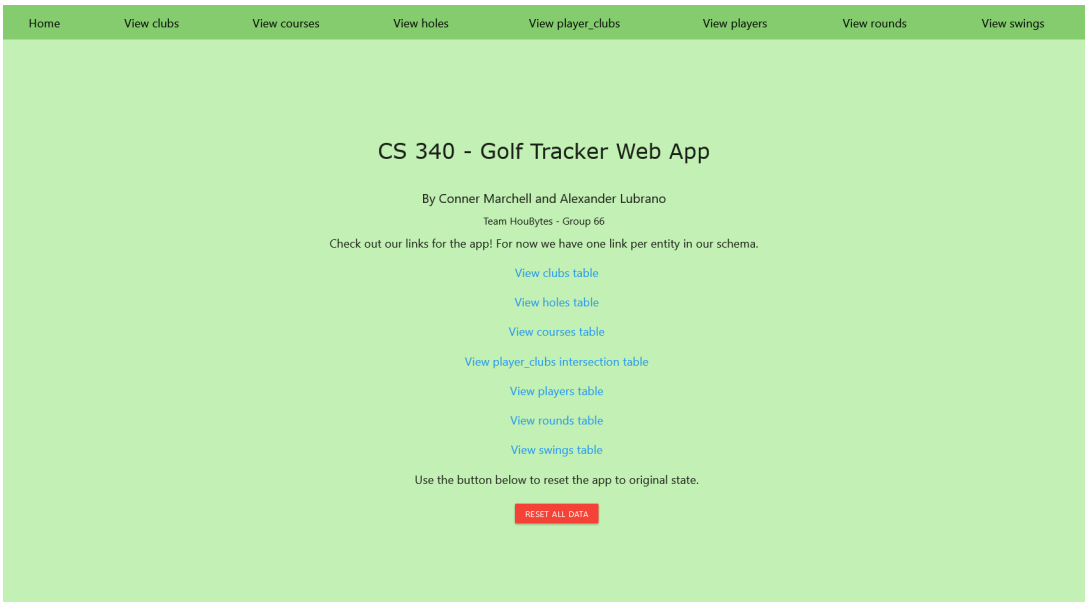


Schema

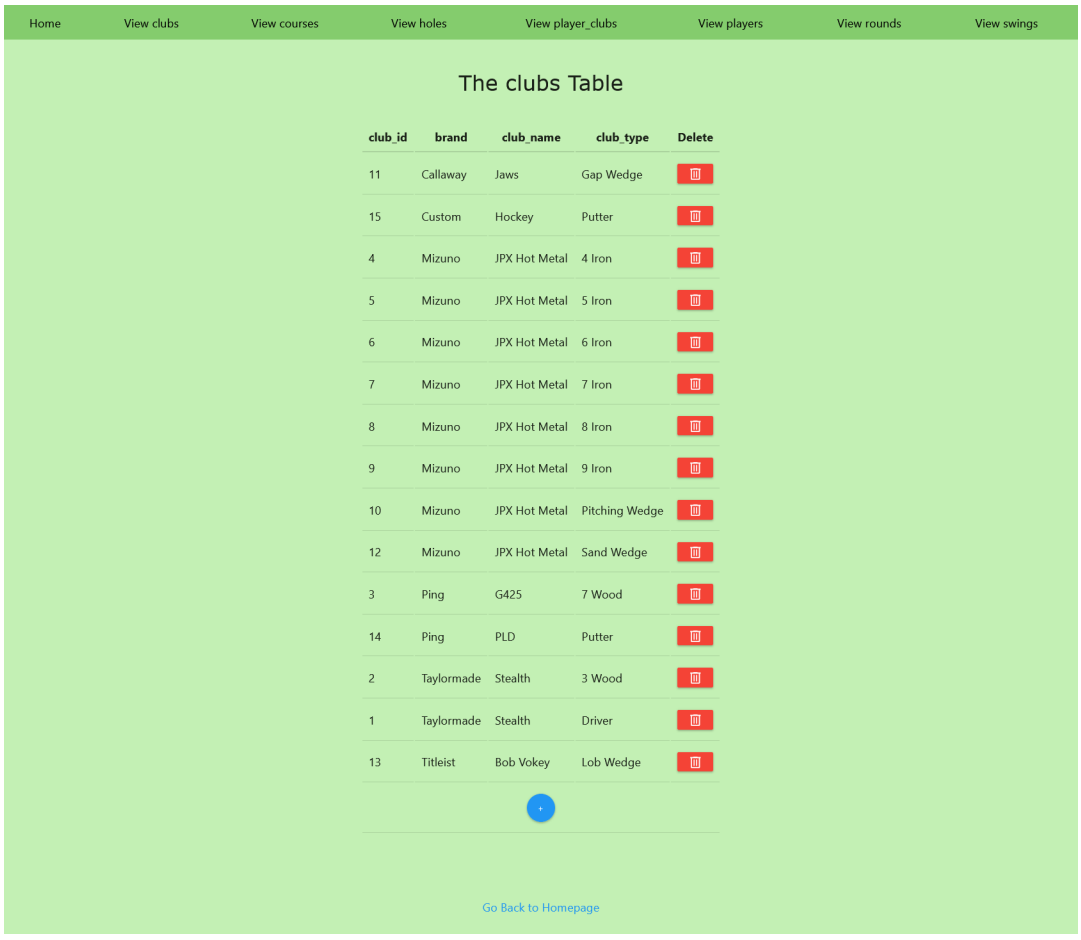

players
player_id
player_name
player_city
player_state
player_clubs
player_id
club_id
clubs
club_id
brand
club_name
club_type
swings
swing_id
player_id
round_id
hole_id
club_id
dist_traveled_yd
holes
hole_id
course_id
hole_number
par_swing_count
distance
courses
course_id
course_name
course_state
rounds
round_id
course_id
player_id
round_date
round_score

Screen Captures

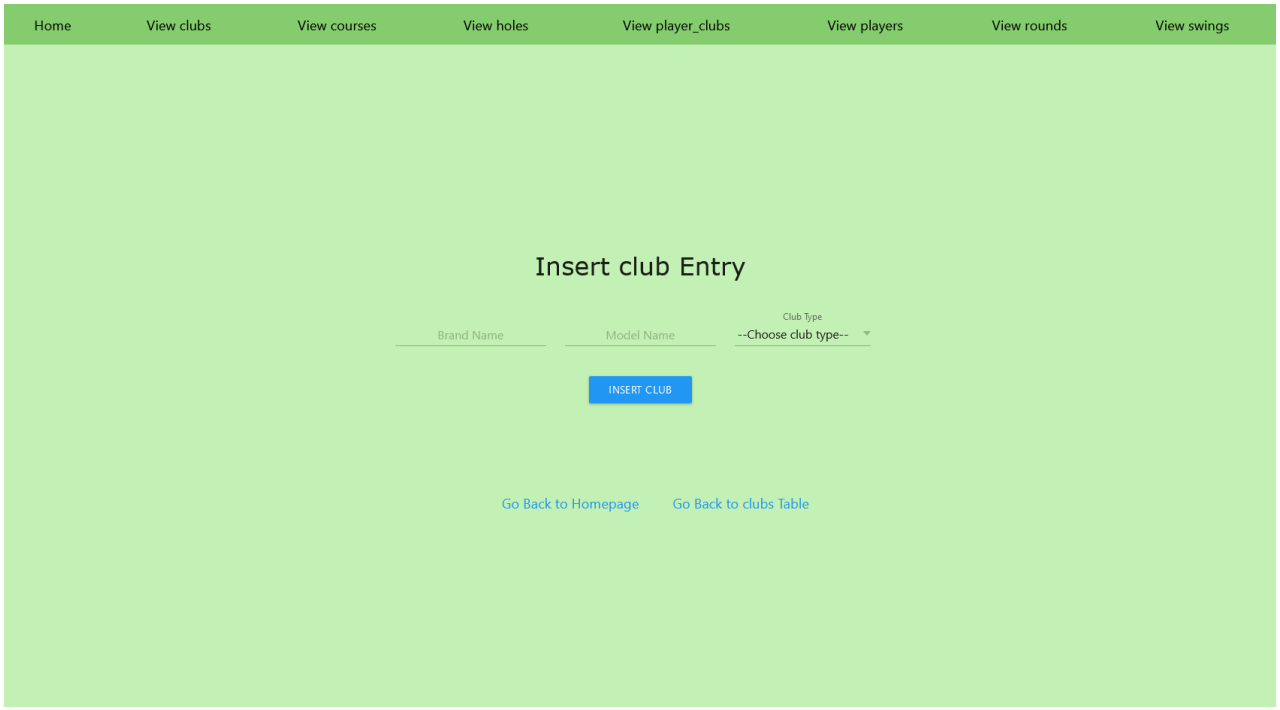
Home Page



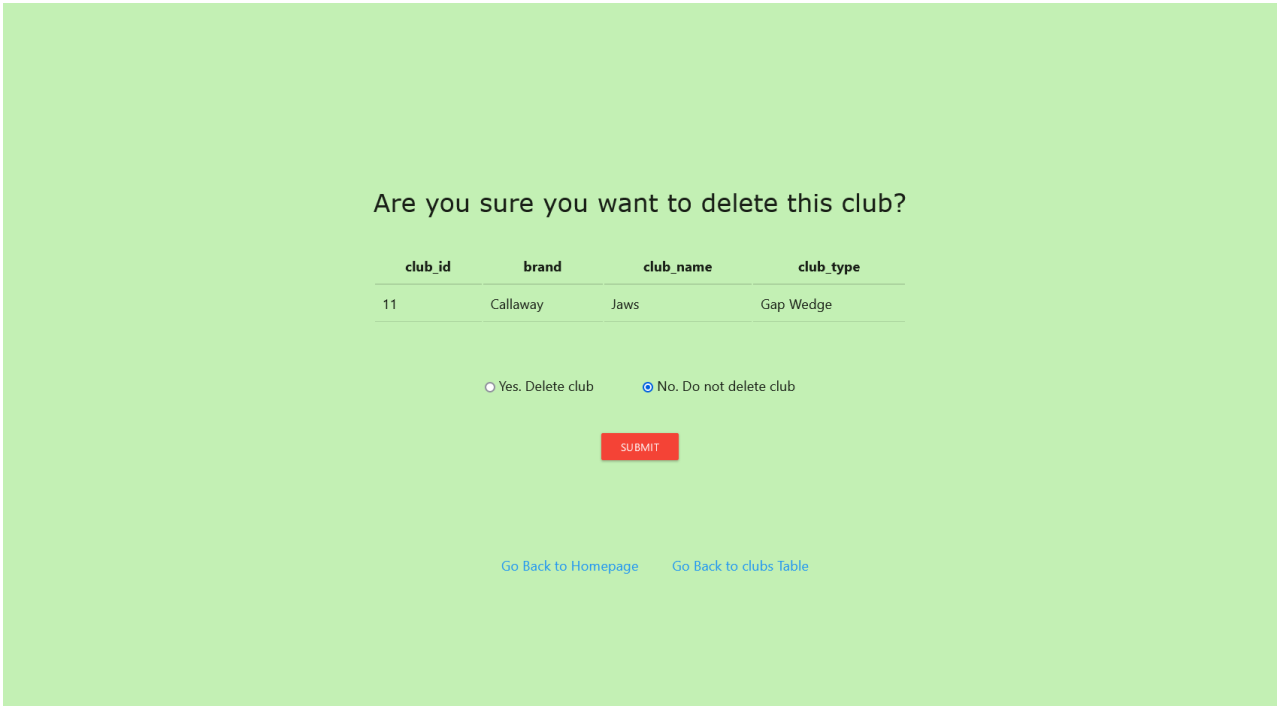
View/Read clubs



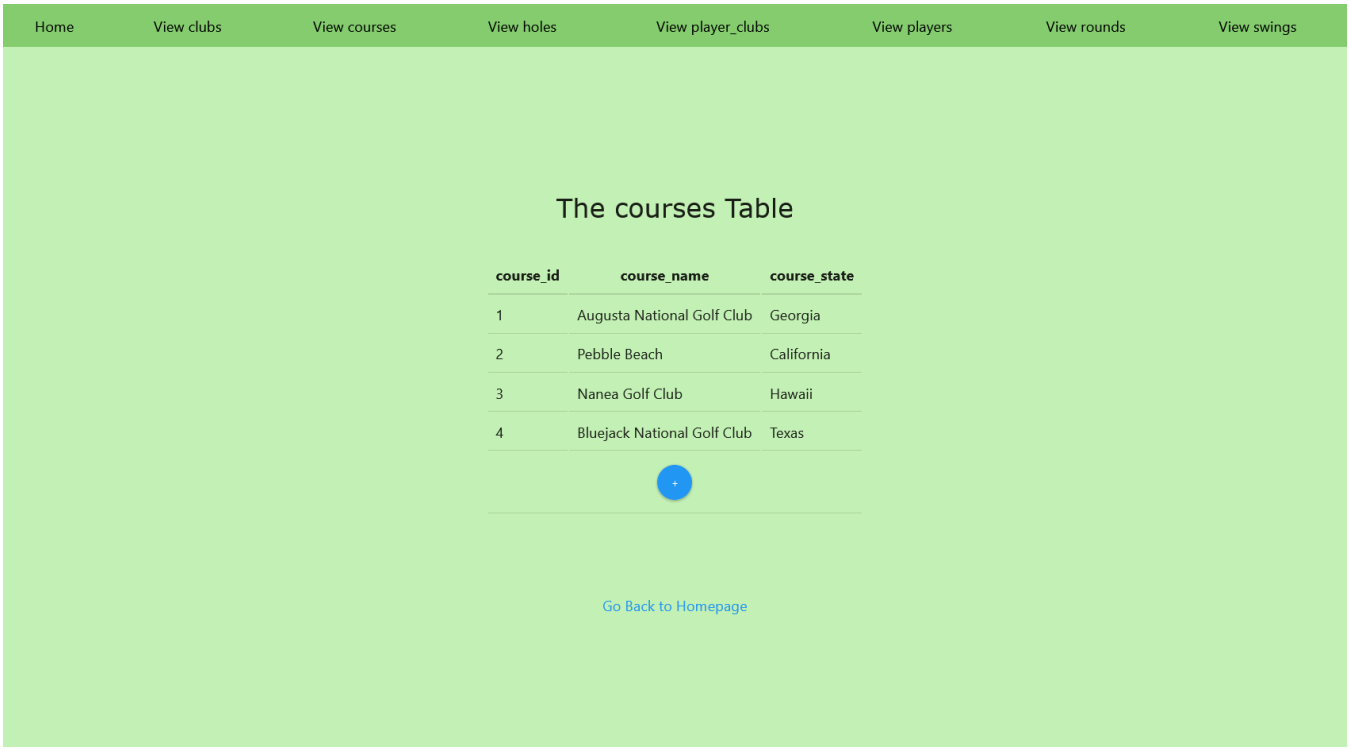
Insert/Create clubs



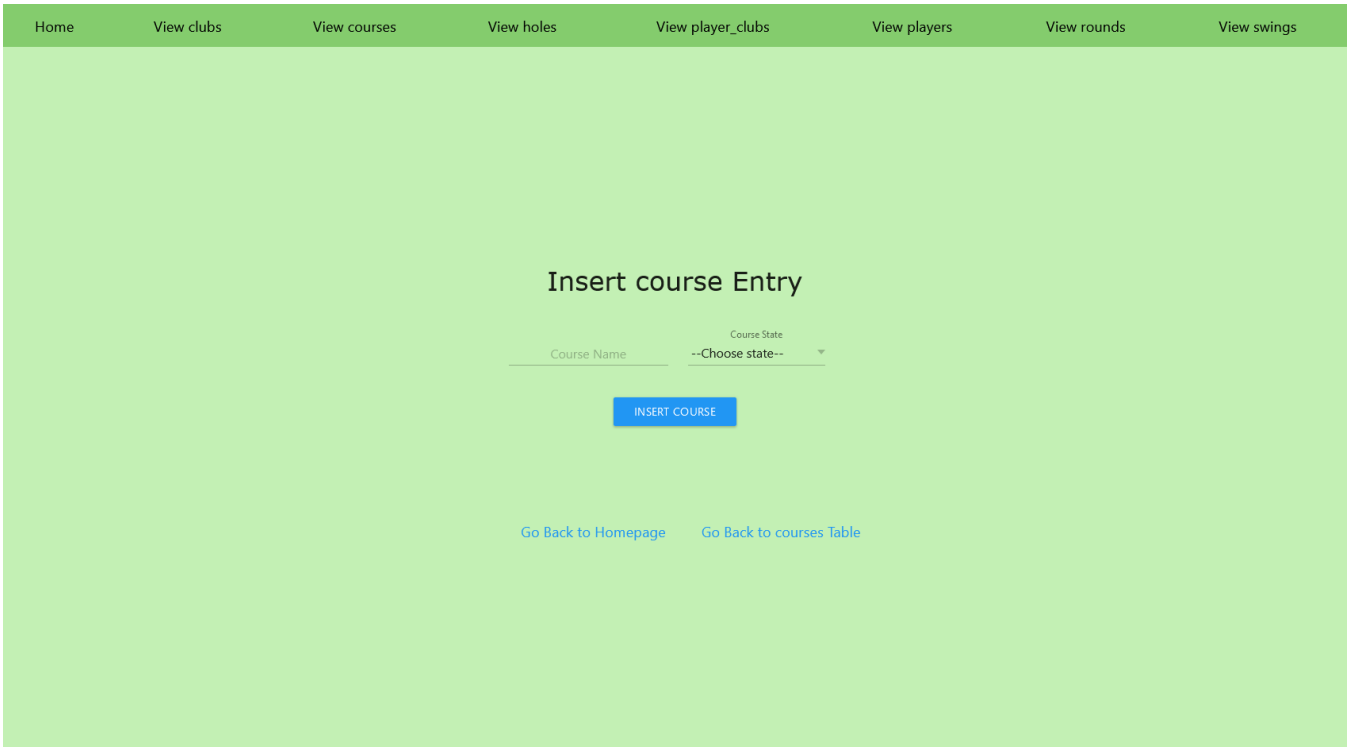
Delete club



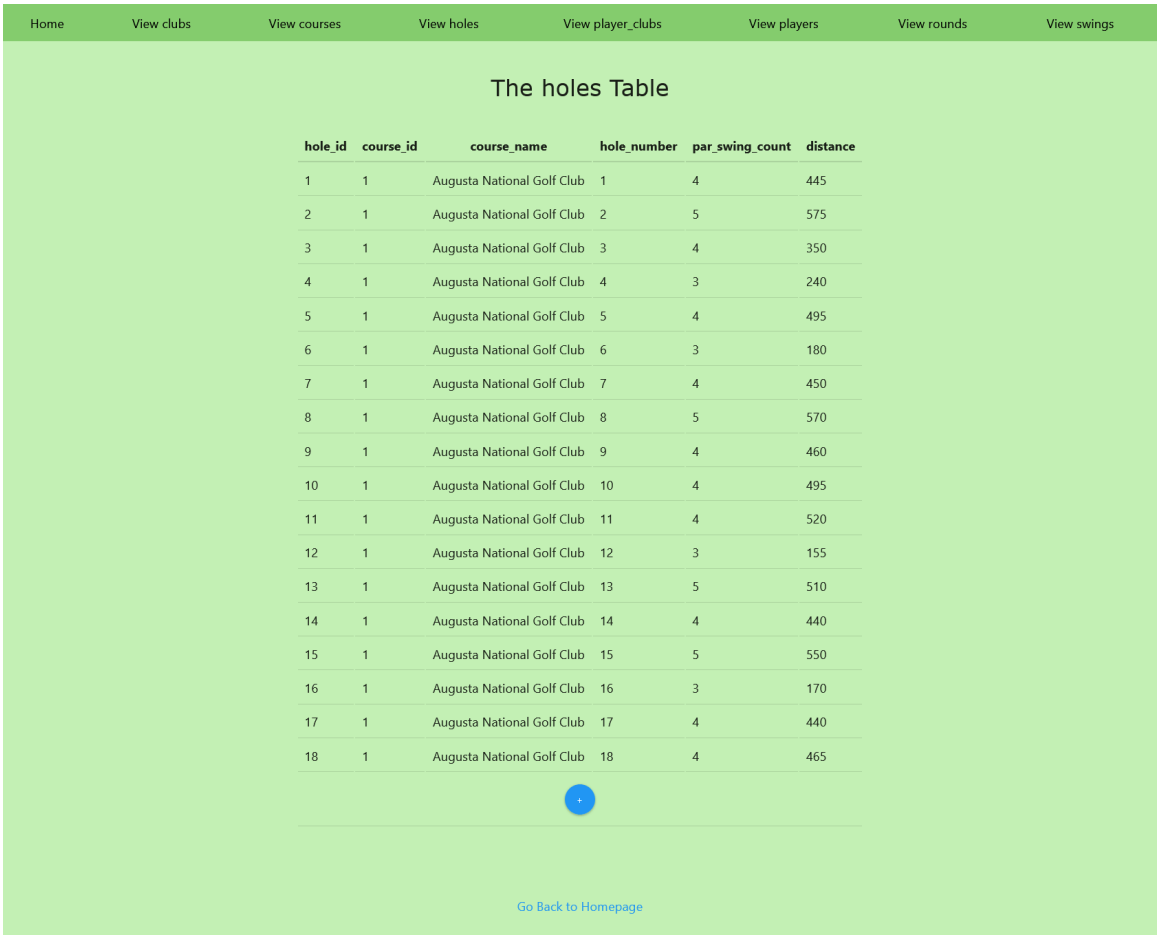
View/Read courses



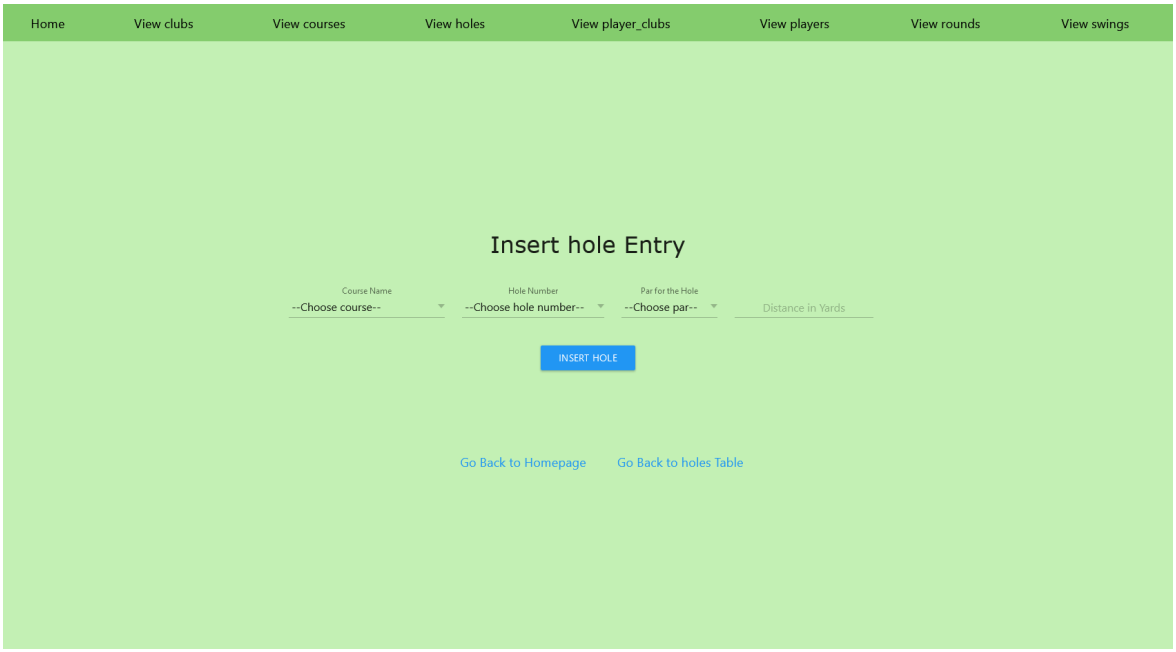
Insert/Create course



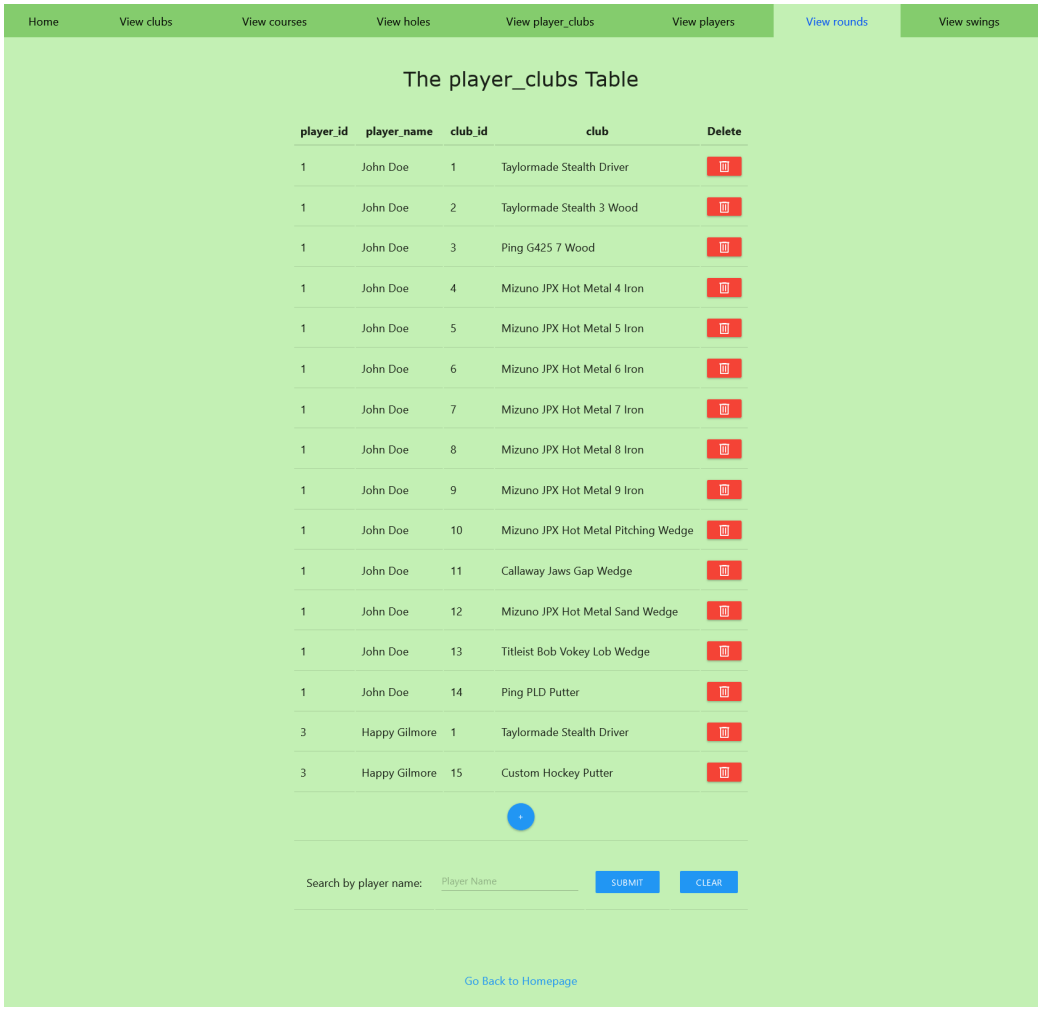
View/Read holes



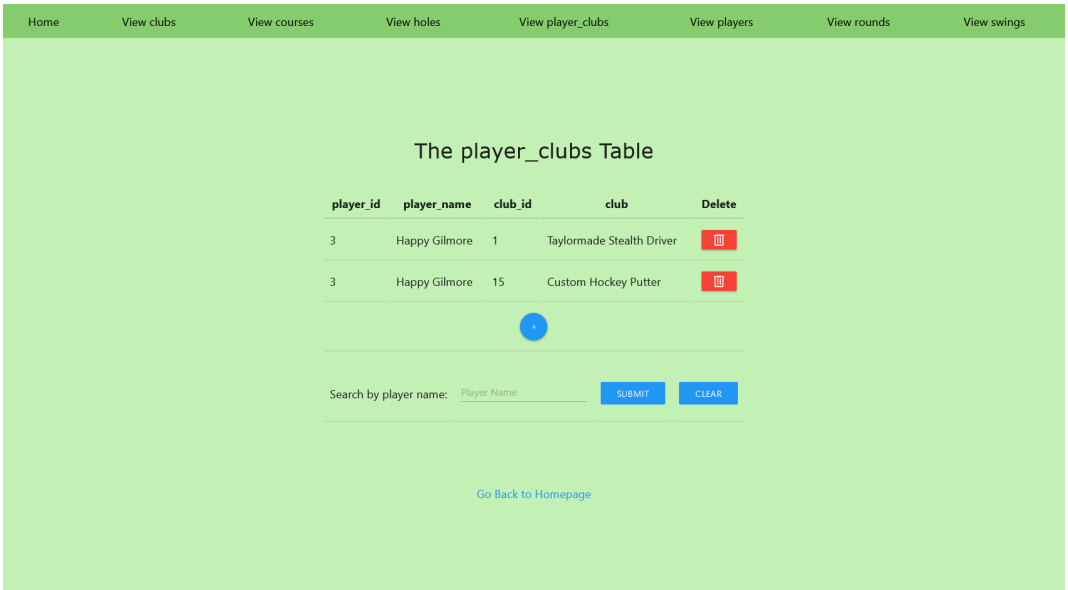
Insert/Create hole



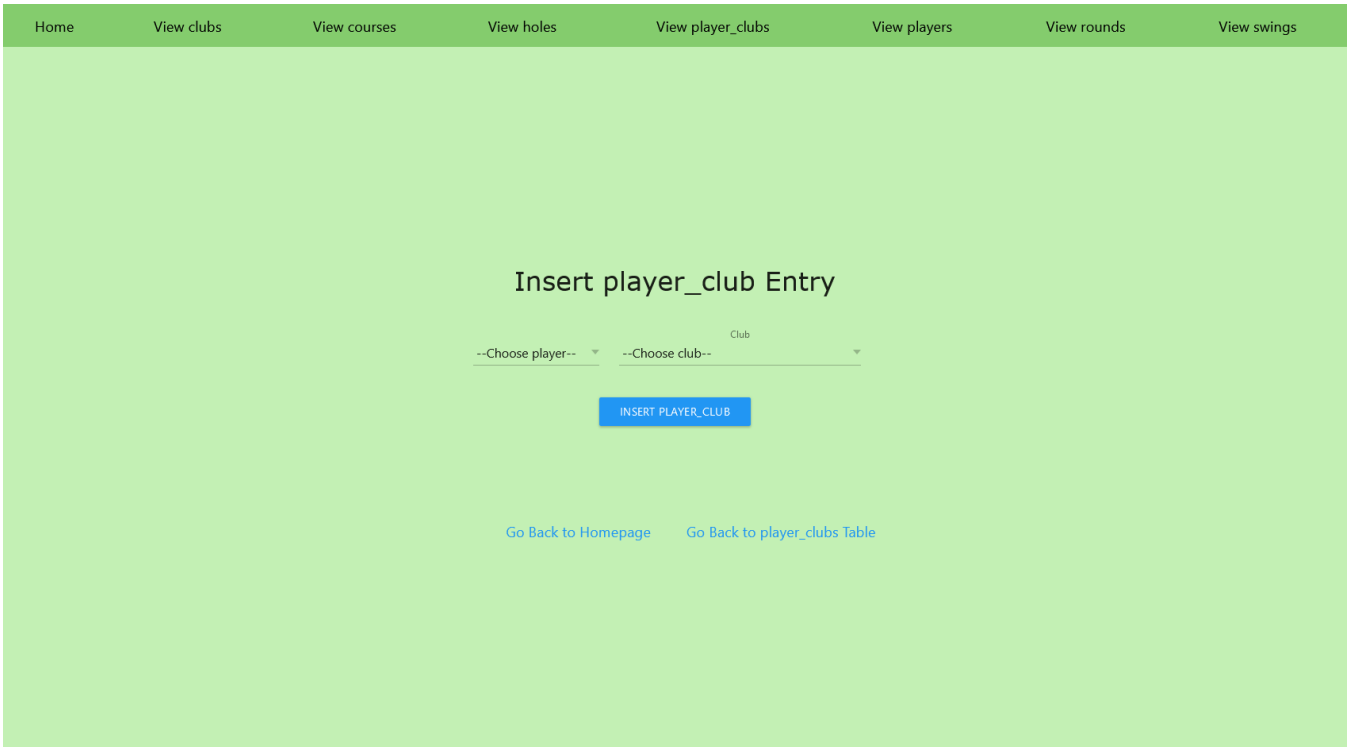
View/Read and Search player_clubs (M:M relationship)



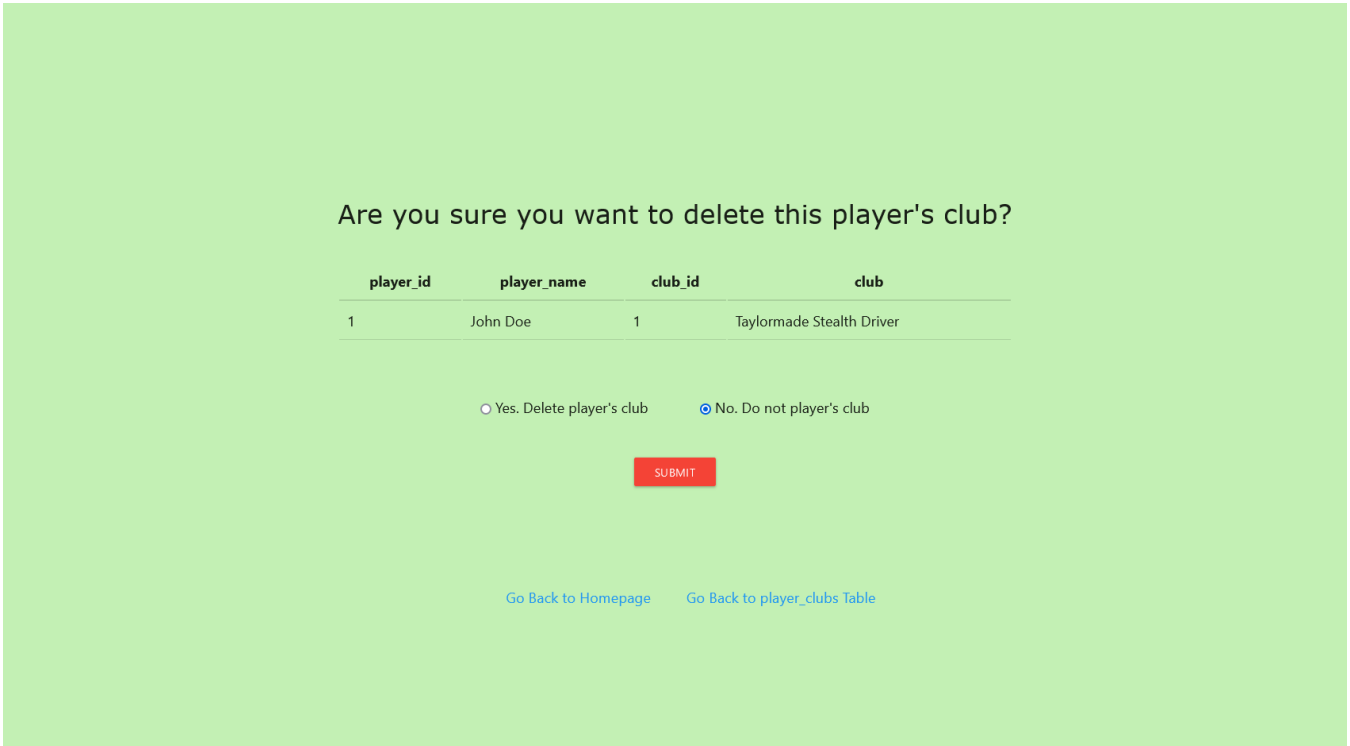
View/Read and Search player_clubs (Search for “Happy”) (M:M relationship)



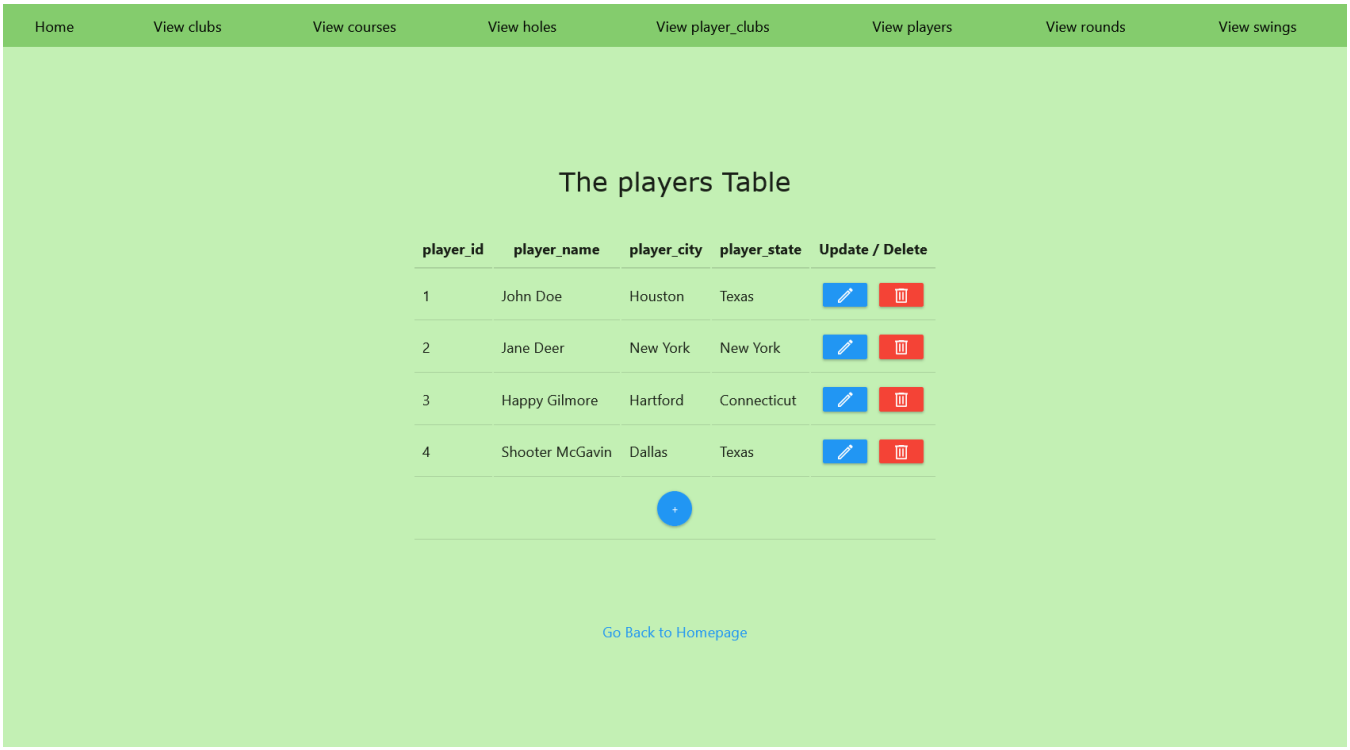
Insert/Create player_club (M:M relationship)



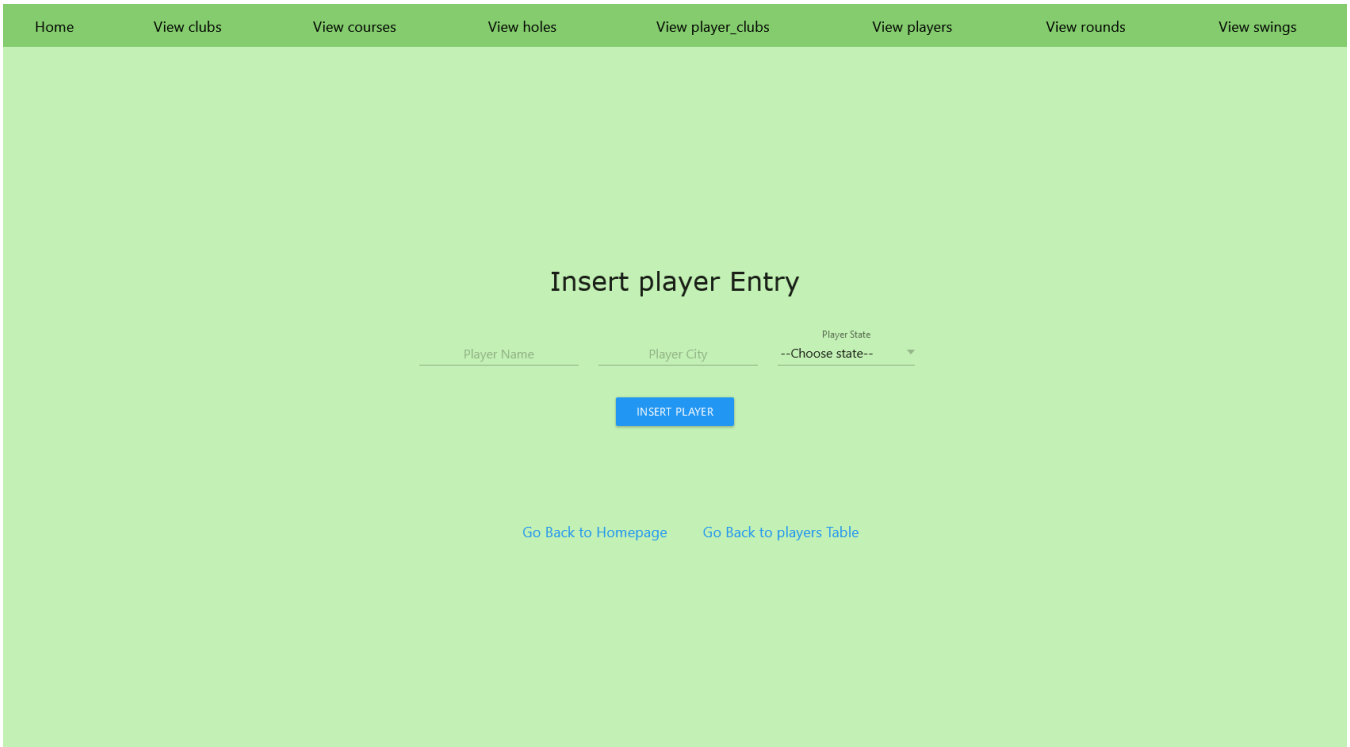
Delete player_club (M:M relationship)



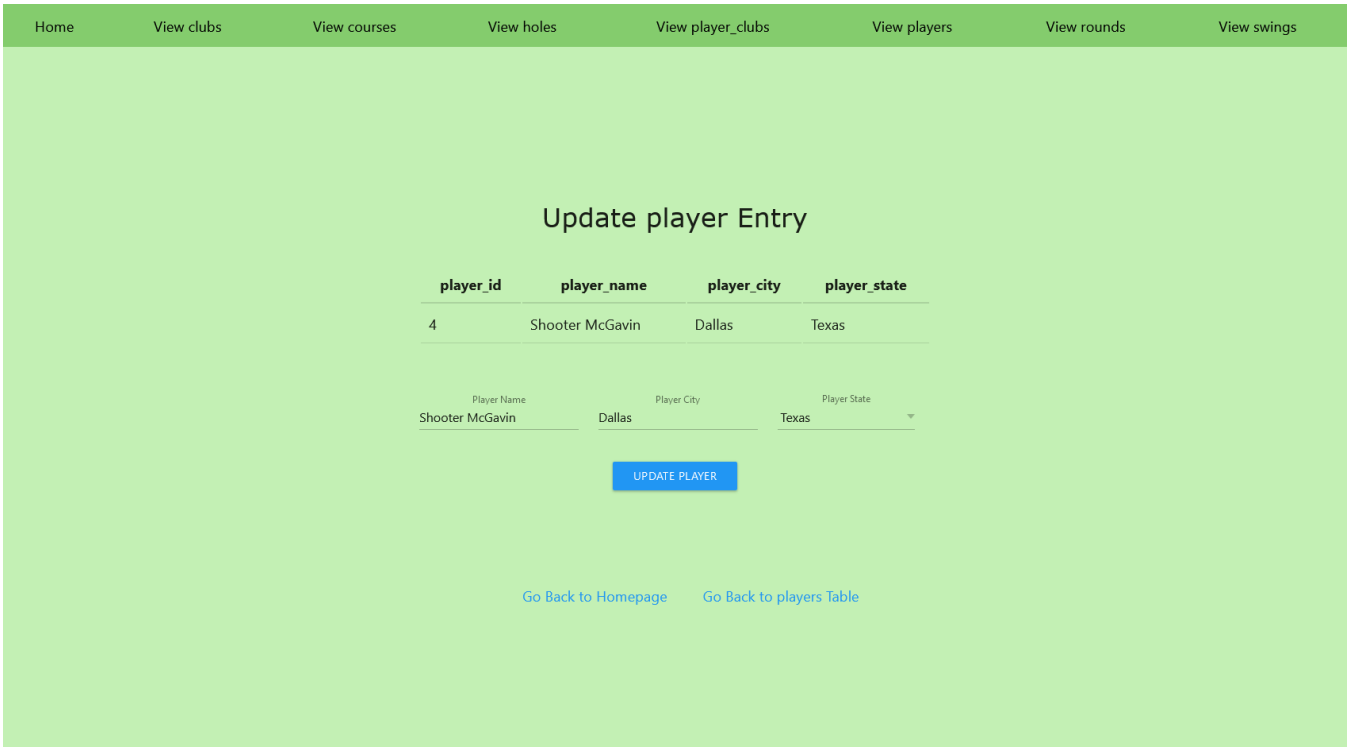
View/Read players



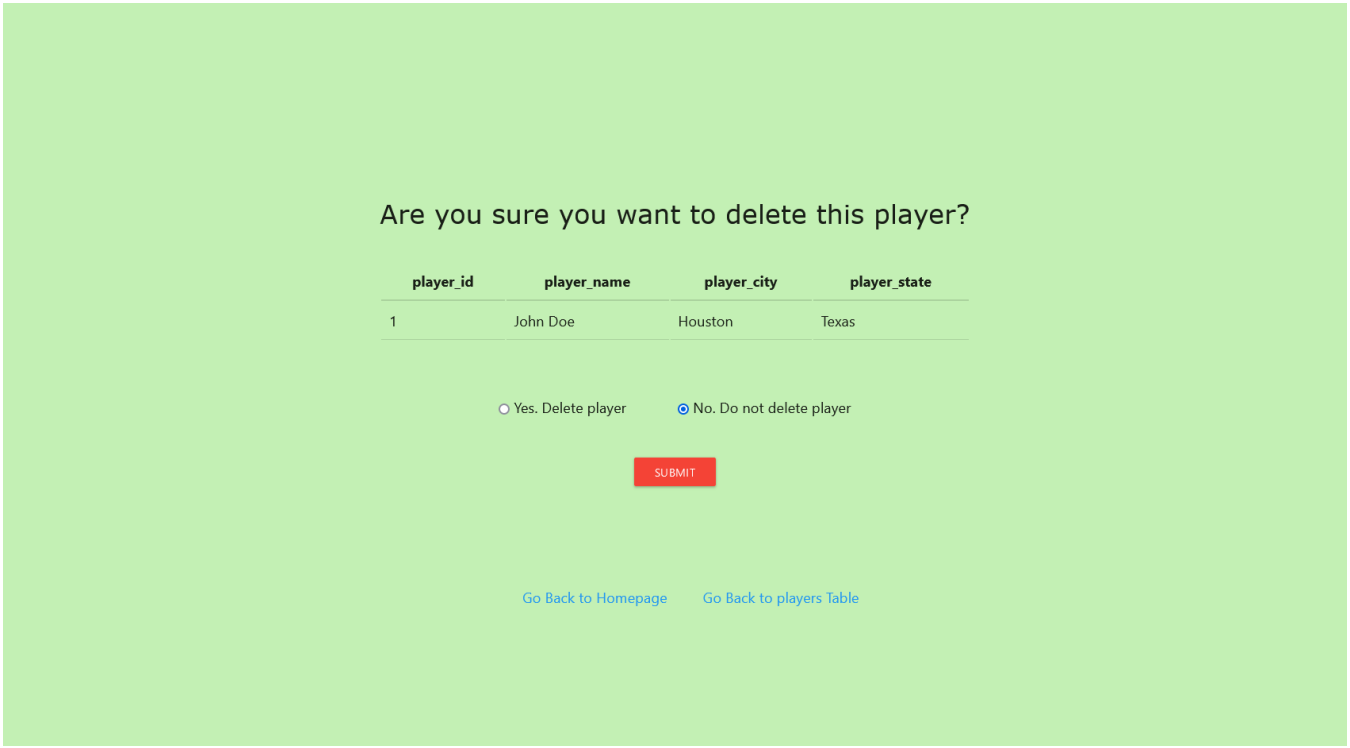
Insert/Create player



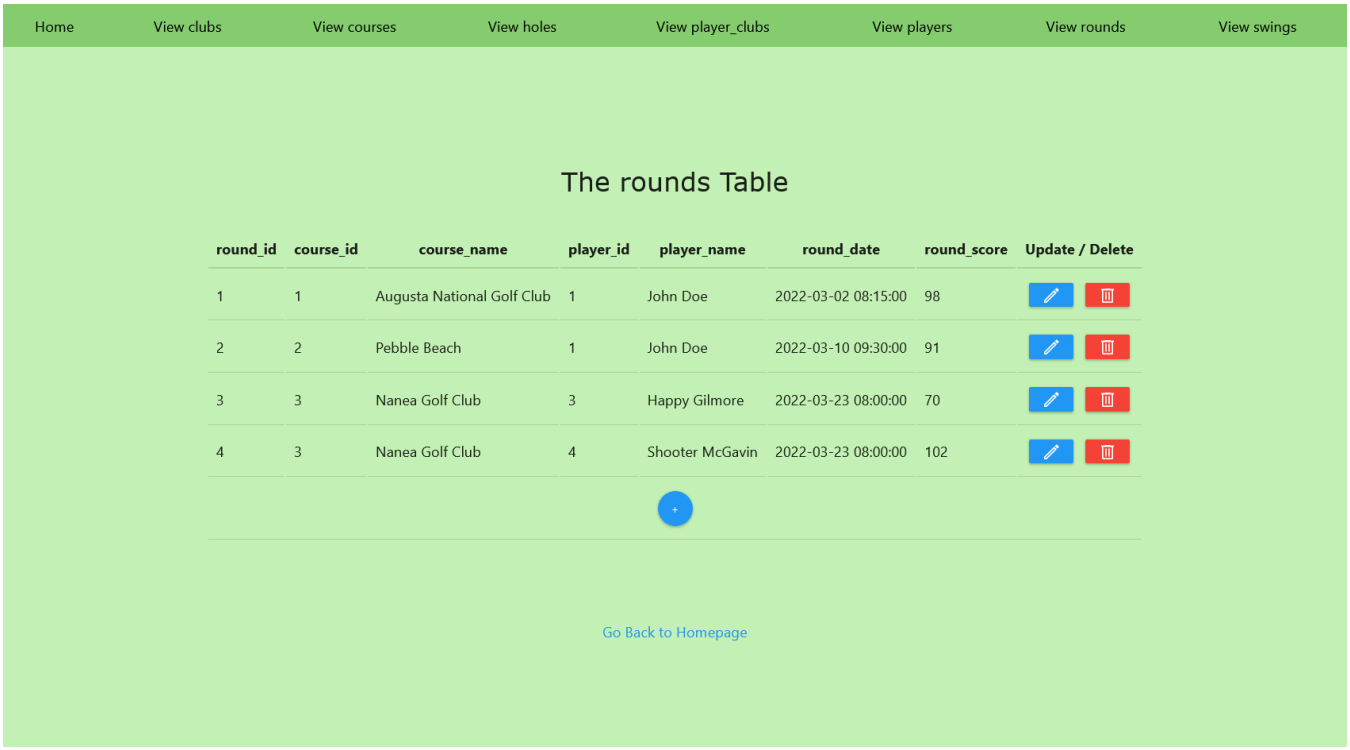
Update player



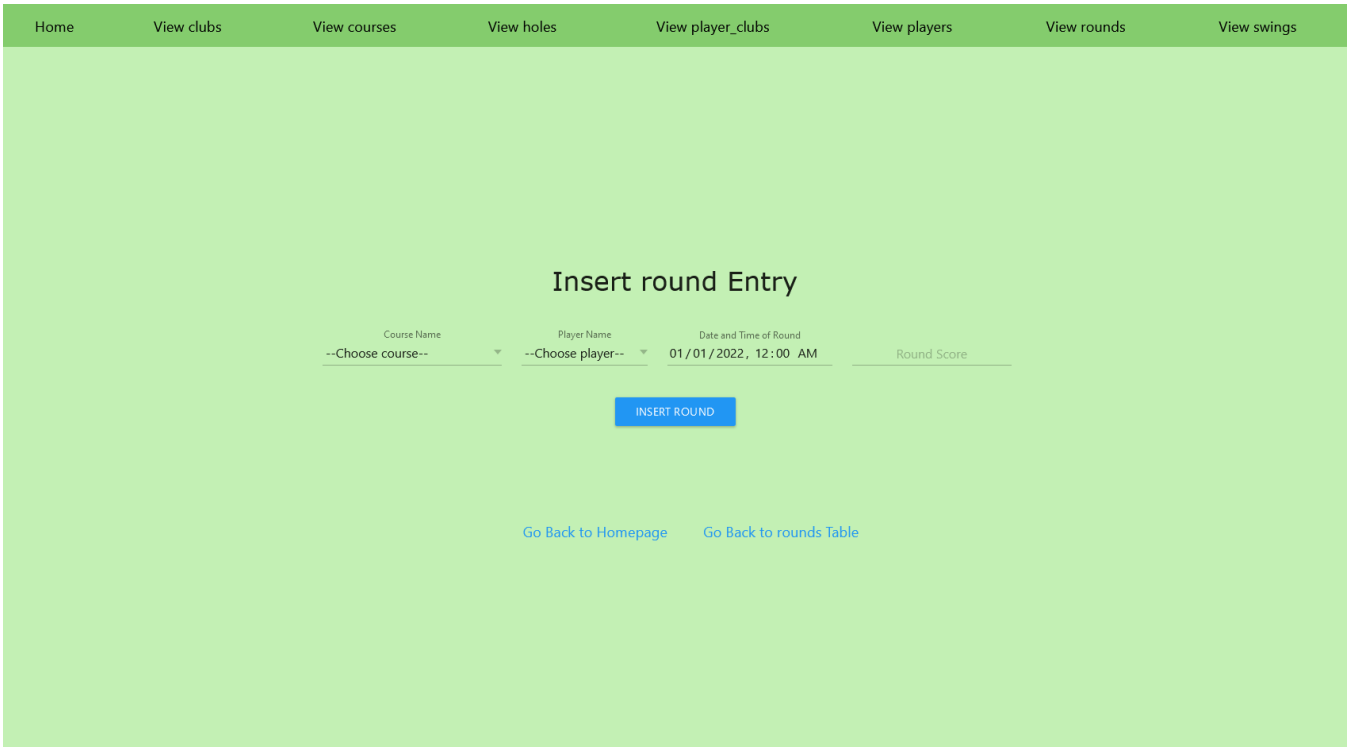
Delete player



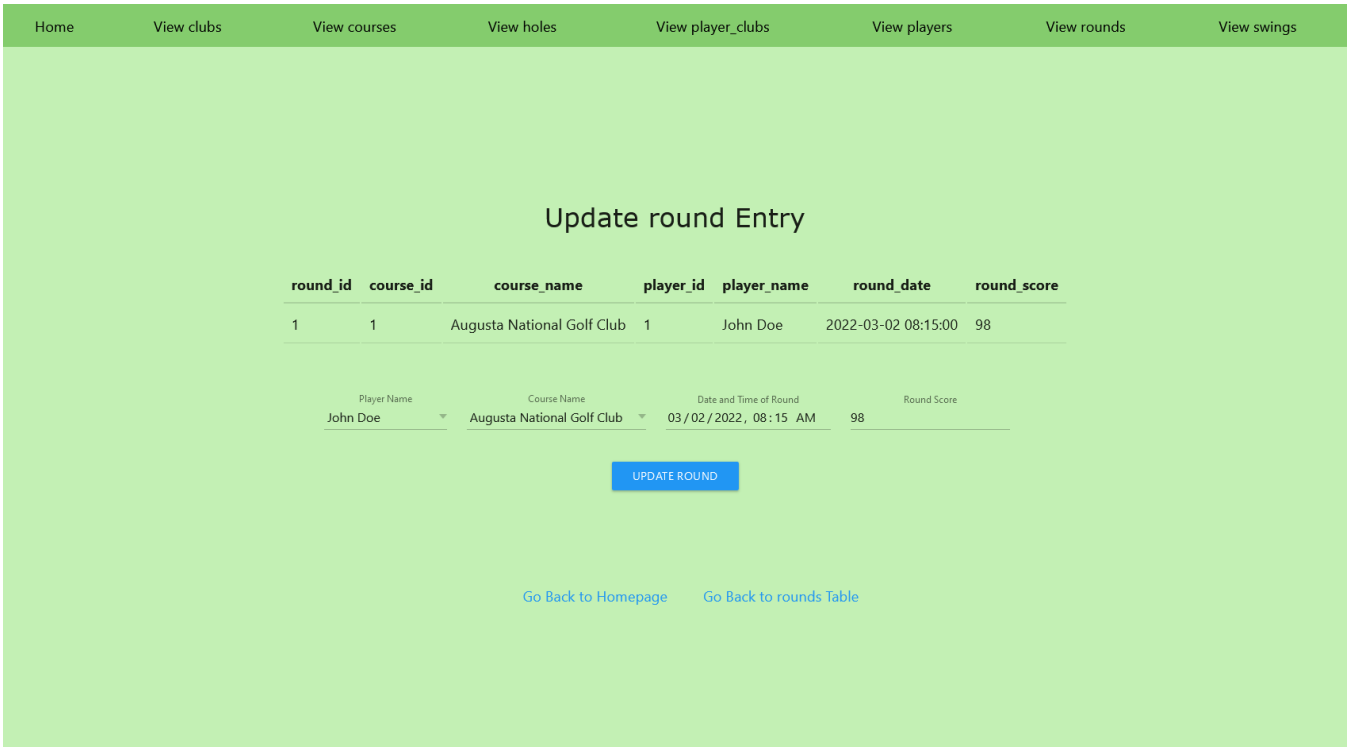
View/Read rounds



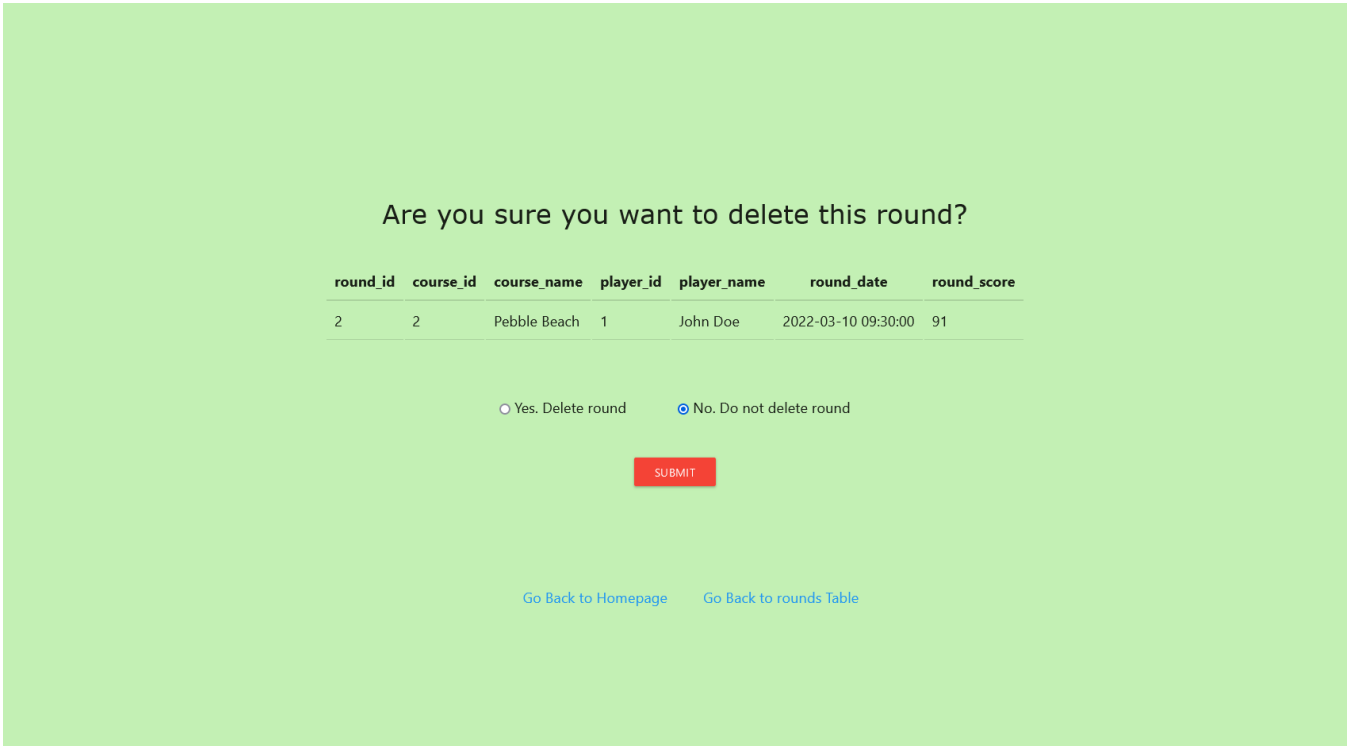
Insert/Create round



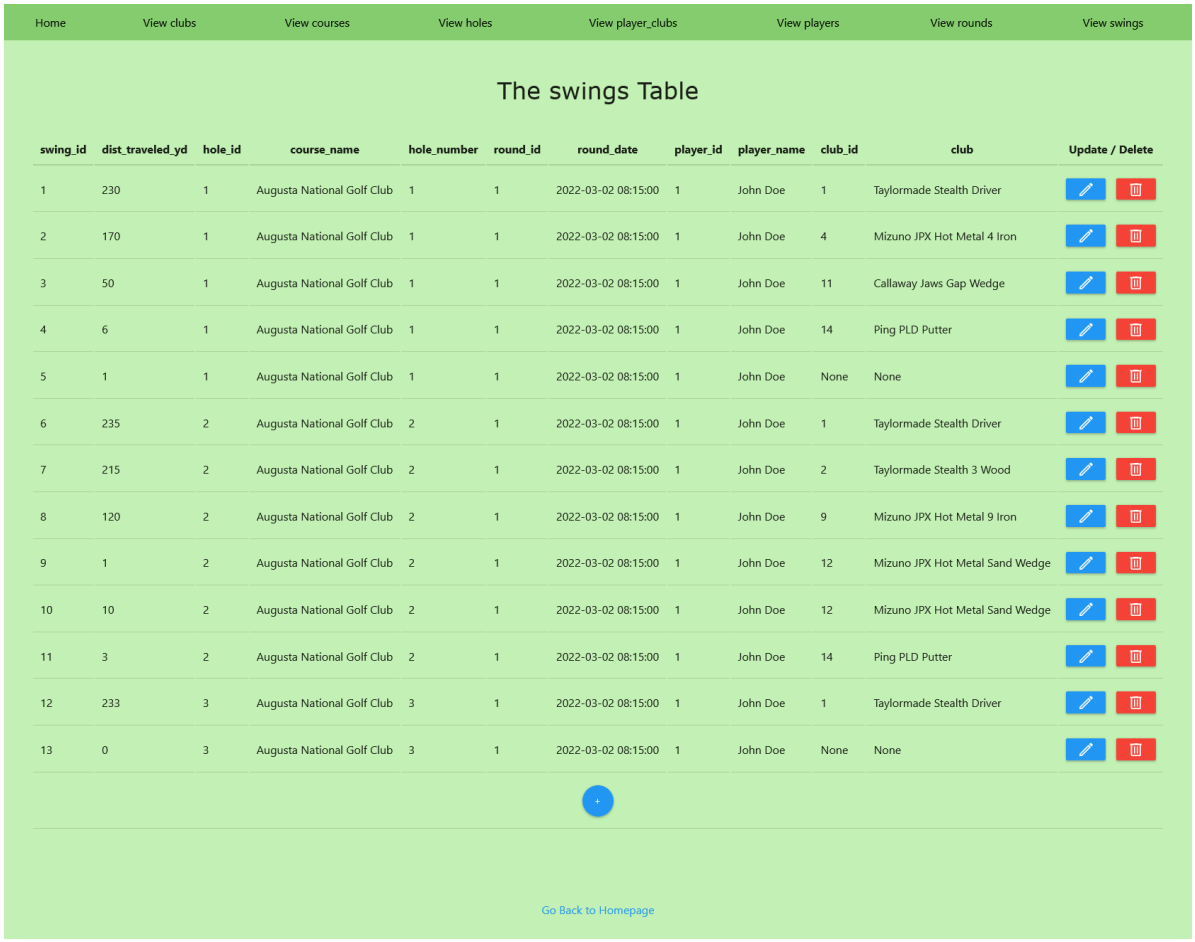
Update round



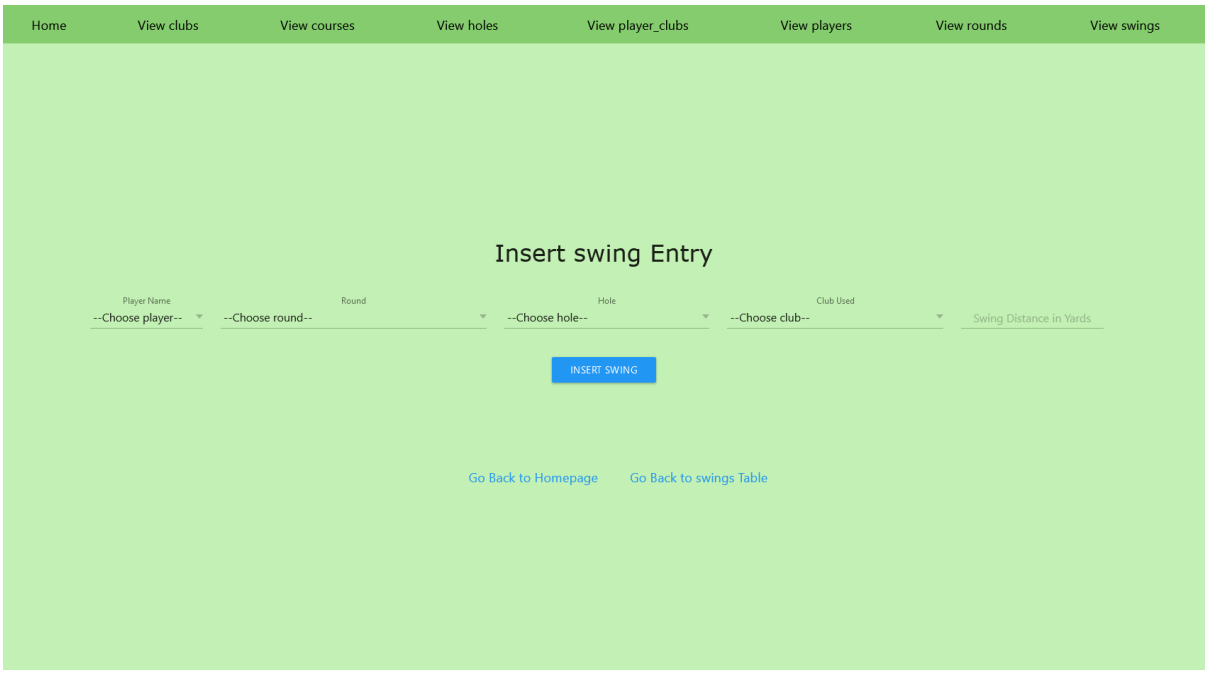
Delete round



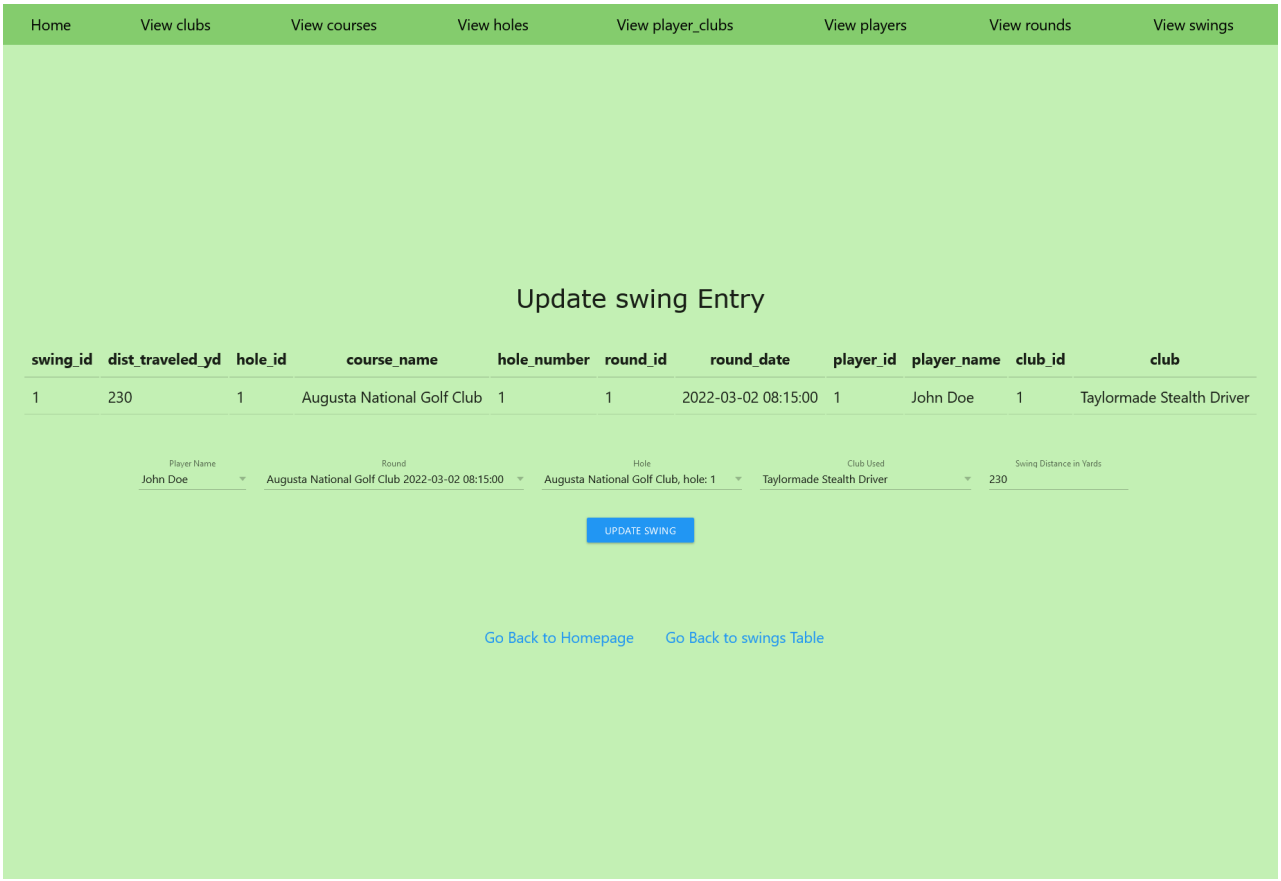
View/Read swings



Insert/Create swing



Update swing



Delete swing