**Project Brief**

Gym

A local gym has asked you to build a piece of software to help them to manage memberships, and register members for classes.

MVP

* The app should allow the gym to create and edit Members ✓
* The app should allow the gym to create and edit Classes ✓
* The app should allow the gym to book members on specific classes ✓
* The app should show a list of all upcoming classes ✓
* The app should show all members that are booked in for a particular class✓

Inspired By

* Glofox, Pike13

Possible Extensions

* Classes could have a maximum capacity, and users can only be added while there is space remaining. ✓
* The gym could be able to give its members Premium or Standard membership. Standard members can only be signed up for classes during off-peak hours.
  + Can either do this and:
    - if session.start\_time between 16:00 and 19:00..
  + Or can do a count of total sessions a customer has had in a given week
    - Silver = 2/week
    - Gold = 4/week
    - Platinum = Unlimited
    - Or by month
  + STEP 1: Change the entries in the membership, to a dictionary?
* The Gym could mark members and classes as active/deactivated. Deactivated members/classes will not appear when creating bookings. ✓

**Still To Do:**

1. All CSS
2. Modify home page for instructions on what this website can do, and how to use it? e.g. All customers must have a unique alias
3. Update and complete diagrams
4. Evidence on testing
5. Tidy up if statements to modernise them
   1. Tidy up my nested if statements in booking repo
6. Route pages for the errors, and suggest hyperlinks to change the problem
   1. e.g. if a customer membership is inactive, hyperlink to a the customer page where they can edit it

**Extensions:**

1. Customer Class
   1. Number of sessions had this month
2. Session Class
   1. Duration calculation
   2. Disallow sessions to be booked at the same time (this assumes the gym has one room only)
   3. Cannot book sessions that are done – We’d need a way of grabbing todays date, and calculating dates/time etc. And maybe even auto move into a “past events” table?
3. Rooms
   1. Can add another DB of rooms
   2. Each session is assigned to a room
   3. Each room and each session has a capacity
4. Other
   1. Capacity can be negative
   2. Dates can be any date at all, and in the past
   3. Times likewise

**Front end functionality for the user:**

* Customers:
  + See all in one page
  + See individual
  + Edit
  + Delete
  + Add new
  + View all bookings a customer has
* Sessions:
  + See all in one page
  + See individual
  + Edit
  + Delete
  + Add new
  + View all customers booked into a session
  + See current and maximum capacity (incase a bunch of people want to attend together)
* Bookings:
  + See all bookings on one page
  + Delete
  + Add new
* Date and time inputs are date/times, not text

**Behind the scenes functionality:**

* Adding a new booking will check for (and error if there’s a problem):
  + Duplicate bookings
  + Active membership status
  + If the session is at maximum capacity
* Adding a new customer will check for duplicates, based on alias
* Editing a customer will bring up their current membership status and level as default (instead of the options list always defaulting to the same value for every customer)
* Date/time inputs are by default taken in, in reverse. These are reversed for then displaying back to HTML in standard order
* Adding a new session will check for availability, as we only have one room. We can’t have two sessions happening on the same day and time

**Logbook of my process:**

1. Went through each brief, what’s most interesting, what would I do for each one, how could I grow each one, and chose my brief
2. Made an Excel file for all my draft drawings, made my DB drawings, classes. Thought what do I want the user to be able to do
3. Set my rough plan:
   1. Set up a git repo, make all my files/folders, get my standard code in, make my classes and some instances, test
   2. Set up my repositories and controllers, with plain HTML for testing
   3. Test all my basic functionality of initial desires
   4. Work through the pages with CSS and images, make it look good
   5. Start on further improvements
4. Made all my files/folders
5. Got my standard code in (e.g. in app.py)
6. Made my classes
7. Made some class instances/objects in console.py
   1. Then found the first thing I need, is to save them
8. Started in customer repository, made save/delete
9. Then same for session, then booking, which uses customer and session
10. Think about what I want to be able to do first on the website – I want to see my customers, see my sessions, and see my bookings in lists, and individually (not necessarily bookings individually at this point)
11. Then made the select all and select by id functions in customers, session, and select all for bookings
12. Test my DB connections work in the terminal by viewing the tables
13. Stopped, recheck over plans, what else do I want to do? I want to edit, delete, and add new
14. Started with delete – Had a problem, session was tied to a customer through a booking. Simple fix, then had to redo terminal psql etc, and tested. Then tested in website, works
15. Edit function likewise had a problem, it would redirect but it wouldn’t change anything. Problem was related to the ID
16. Then made the edit function for customer. And repeated both for session
17. Then added new function for customer, session, and booking
18. We can now:
    1. View (all and individual), Add, Edit and Remove Customers
    2. View (all and individual), Add, Edit and Remove Sessions
    3. View (all), Add, and Delete bookings
19. Added a feature to check for duplicates to bookings
20. Repeated for customers
21. Repeated for sessions with date/time checks since we currently only have 1 room
22. Ability to delete bookings
23. Date inputs changed to actual dates
24. Time inputs changed to actual times
25. Added a whole bunch of other checks
26. Update all diagrams
27. Format the website with CSS, including Images