**Project V2 - Project Report**

**2.1** *(Conceptual Design)*

An image of the ER-Diagram can be found in the Documentation folder.

Link to ER-Diagram in repo:

**2.2** *(Reduction to Relation Schemas)*

An image of the Relational Database Schema can be found in the Documentation folder.

**2.3** *(DDL File)*

The DDL File can be found in the SQL folder.

Link to file in repo:

**2.4** *(DML File)*

The DML File can be found in the SQL folder.

Link to file in repo:

**2.5** *(Implementation)*

**Application Architecture**

The application is designed to provide functionality of a fitness club web application. The architecture of the application follows a client-server model with a relational database management system (RDBMS) serving as the backend storage. The main idea is for there to be three separate sections of the website, which will support actions for members, trainers, and admin staff respectively. The main source code is stored inside of the ‘/code’ folder.

**Components in /code:**

1. **/public**: This folder holds the bulk of the code for the client-side. This includes the scripts, CSS styling, and even some images for the design of the website:
   1. **/images**: These images are used for adding style to some of the webpages. It also includes an image used for the shortcut icon found on the browser tab.
   2. **/js**: The folder holds the scripts which are used for the majority of the backend for this project, and it is split into three separate folders (members, trainers, admin) for ease for transversality. It also includes logregscripts.js which handles the scripting for the login/register page, and the navscripts.js file which is used for traversing between webpages (HTML files) while the program is running.
   3. **/styles**: This folder holds the CSS files responsible for styling the website as needed. It includes separate files for member web pages, trainer web pages, admin web pages, and the login/register web page.
2. **/views**: This folder holds the .html files which will be used for loading up the webpages that will be needed for the remainder of this project. They are separated into three separate folders (members, trainers, admin) to distinguish the web pages for each website.
3. **server.js**: The server is responsible for establishing a connection to the PostgreSQL database, and handling the middleware. Once a connection is formed, the server will be started on PORT 3000, and will provide links in the terminal for testing purposes.
4. **routes.js**: All of the routes for sending queries to the PostgreSQL database are stored in this file. This file is also responsible for allowing navigation between webpages.
5. **Relational Database**: This relational database serves as the persistent storage for the application's data. The chosen system for testing purposes is pg Admin 4, and it utilises PostgreSQL.

**Technologies Used:**

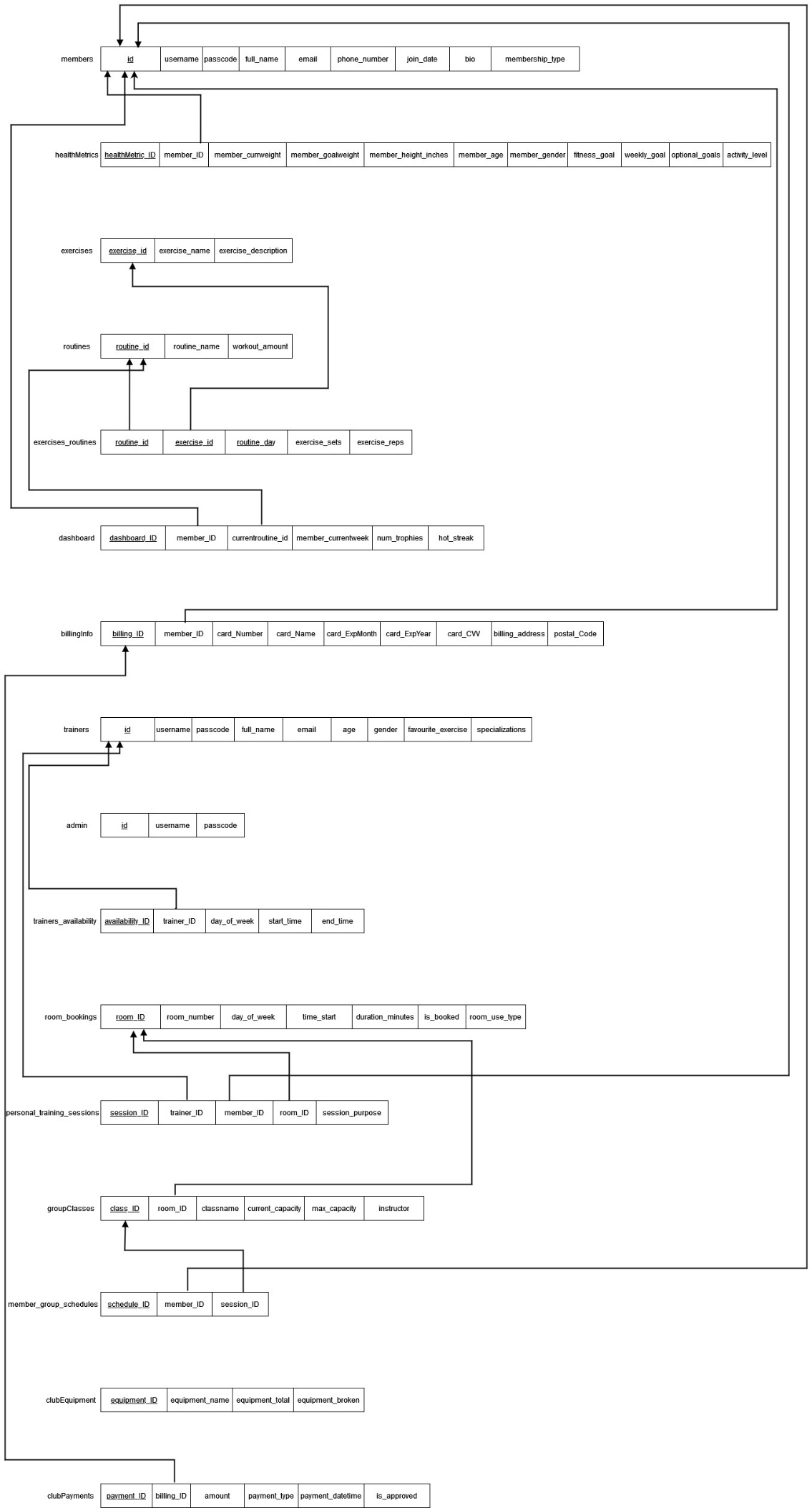
* **Programming Languages**: JavaScript, HTML, CSS
* **Frameworks/Libraries/Environments**: Express JS, Node.js, node-postgres(pg)
* **Relational Database Management System (RDBMS)**: PostgreSQL

**2.6** *(Bonus Features)*

Bonus Features:

* Included a login page
* Included a password system
* Included a pop-up side bar navigation bar for members
* Implemented the profile web page in a way where there are different sections for the profile

**2.7** *(GitHub Repository)*

*Relational Schema:*