# Programming Documentation

A large part of the back end of the web application is the database. To implement the database, we first had to design the tables. We examined the brief, the Figma prototype and the existing Thornton web app to produce a list of tables that would be needed along with the fields for each table. Once we had identified the tables that would be needed, we laid out each table with all fields and their datatypes. After completing that, we discussed the tables, identifying if we could separate certain fields into new tables to normalise them into third normal form.

Once the tables had been normalised, we began to think about the relationships between each table and how they would be linked together. We used a whiteboard to map this out, based on this whiteboard we then produced the first draft of the ERD (Entity Relationship Diagram). This ERD gave our back-end developers a greater sense of how the tables were laid out and how they connected to each other. After production of the first draft, we discussed it and drew up an altered version with any changes made highlighted. Through the discussion we also identified any areas that clarification from the client would be needed and made a note of this. This version of the ERD was then digitised and made into the second draft.

We also made a note of tables that were finalised and could be implemented. We considered the clarification we needed and came up with questions to ask the client to gain clarification. After gaining clarification from the client, we thought about necessary table alterations, including adding columns, separating tables, and removing some tables entirely. Then, we redrew the ERD as a team using a whiteboard. Once we were satisfied with the final design, we digitised it. This final design would be the basis for development of the back end.

For implementing the database, we decided to use the repository design pattern to minimise code duplication and to make it easier to edit if we need to. This involves using a repository which holds all possible operations for the database, the repository serves to provide a separation between the database and the rest of the application. This design is more ideal for larger scale projects and allows for greater control over code as opposed to using autogenerated code produced by using scaffolding.

The web application uses two types of users each with different levels of access to the web application, for example, Admin users can access the admin pages where they can edit directly to and from the database through crud functionality. This prevents unregistered users for maliciously manipulating sensitive data that the university chooses to publish.

Admins can access the building page which allows them to add new buildings to the web app, this information will be updated and displayed on the homepage which means users can see the latest changes while also having immediate access to the laboratories (such as the energy centre.

If users wish to get into contact with staff members, then they can access their credentials from the contact page, this means that if users have any queries then they can send staff members an email.

# Login Credentials

Admin Username: admin@labmanagementsystem.onmicrosoft.com

Regular Username: regular@labmanagementsystem.onmicrosoft.com

Password: PasswordUOC123456

Users are contained within Active Directory. As the client desires for university accounts to be used to login, the Active Directory credentials within the application’s “appsettings.json” file will need to be replaced in a production deployment.

# Form Emails

The “Report Adverse Event” and “Request Existing Resource” pages use an external service named “Getform” to send emails. This resolves SMTP issues such as mail going to spam or being rejected due to mail coming from an unrecognised source.

The details to access this service are as follows:

Username: experimentalteama@yahoo.com

Password: groupA123

# References

Dot Net Tutorials. (n.d.). *Repository Design Pattern in C#*. Dot Net Tutorials. Retrieved May 24, 2023, from https://dotnettutorials.net/lesson/repository-design-pattern-csharp/