Chapter 5: Conclusion and Future Work

Conclusion

As shown in both the system and user testing, all of the top priority functions from the first development cycle work on the backend, but several desirable features of the application are not fully functional on the front end. While slightly simplified from the initial vision of the program the forecasting and simulation functions work completely on the backend, as shown through the system testing in Chapter 4. This shows the partial completion of all of the Must Have features of the system, which with a small period of further development would be fully completed. Many of the other features relating to these are fully working through the full stack, such as being able to select the timescale, solar and wind settings etc. In addition to these the vast majority of secondary features of the program are running either across the full program or at the very least on the backend. There were a few key issues that led to not all functionalities being fully implemented.

One of the key issues with the undertaking of the project came from the multiple, lengthy database restructures. Over the five primary weeks of development there were two major database restructurings, in addition to several additional minor ones. Both major reworks, in part, had to be undertaken to resolve issues with handling the outputs of the simulation and forecast calculations in order to pass the data to the front end, and to store the data for future analysis. Initially too much data was allocated to the tables storing previously run forecasts and simulations, making the data handling unwieldy. In order to circumvent the difficulties with data handling the storage of data outputs and the lists of regions and counties used in calculation were moved to separate tables within the database, and these tables were linked with Many-to-Many and Many-to-One relationships to the primary forecast and simulation data tables. While there were other issues with the database, this issue was the one which cause the greatest disruption to the development of the actual application. In future projects, or if redoing this project, greater focus would be placed on the design of the database prior to undertaking development, the database is a core component of almost all software programs, and should be given the corresponding appropriate attention.

In addition to the restructuring of the database another major issue relating to the passing of data from the back end to the front end was the somewhat cumbersome data structures used to pass information to the front end. Maps were used in several instances to pass data from simulations and forecasts from the backend to the front end, maps were chosen as they simplified the database tables as less data needed to be stored in each table entry as well as simplifying the passing of the data to the front end, by using a few maps instead of dozens of arrays, it was easy to package all the data together into one simple entity to pass to the front end. However, on the front-end, some difficulty was encountered extracting this data from the maps and displaying the data to screen, so while the database and backend were simplified by this choice the front end was made more complex.

The time delays corresponding to database restructuring and issues with retrieving data from the backend severely impacted several aspects of the development, the primary of which being not being able to display the results of run forecasts and simulations graphically, and leading to the decision to not implement an interactive map in the initial development cycle.

Testing highlights that a some of the functionality of the back end is not completely applied to the front end. The delays arising from the prior mentioned issues meant that the predicted schedule had to be adapted, however this reduced time for development was somewhat incorrectly allocated, the focus was placed on getting all functionality working on the backend initially and then once this was almost complete, more attention was to be given to the development of the front end of the program. Due to the effective reduced time, not enough time was left to develop the front-end to the same standard as the back end. In future development periods, allocating more time to full implementation of primary features instead of distributing the time to the less essential features, and thus adhering more directly a MoSCoW centred Agile development structure. It however should be noted that several of the functions that were lower priority needed to be implemented on the backend in order to allow for high priority functions to work across the full stack and were therefore easy to implement on the front-end ahead of other, more essential, functions. For example, while displaying a list on the front end of previously run forecasts was less important than being able to rerun a previous forecasts, the backend logic of saving and retrieving forecasts had to be implemented, and thus it was an easy task to implement the display of the list prior to the rerun function on the front end.

The UI in this first iteration of the program is fairly simple, and if future development were to be undertaken it would be given a more polished finish, but for an initial release the UI meets an acceptable standard. The simplified design still followed all visual design principles outlined in Chapter 3, using sans serif fonts and neutral tones to increase readability, reduce eye strain and give a more professional look to the UI. With some of the components of the UI being quite simple, such as the forms and buttons, it could in the long term be unattractive to users as if the product looks visually simplistic it can give the impression that perhaps the underlying program is also simple, so before a full release of the program the UI would be given a more clean, professional look.

Ultimately, despite the fact that the application does not fully realise some of the important desired features, the database and backend are fully established meaning that in future development cycles only small modifications would be needed to bring to fruition the full functionality of the user interface, from the features that are fully developed in the backend, and create a product that meets all the user’s needs

Further Work

While the initial design period did achieve implementing many of the primary desired functions of the program, some were incomplete or were developed only to completion on the back end. The first step in future development cycles would be to add the functionality to the front end to complete all the actions that are presently working on the back end. The next key step would be to create a graphic interface element to graph the data produced. Following on from these two key points, further functionality and features could be added across the scope of the application, such as: finalising the registration and login in system to allow users to customise their experience and store data privately, partitioning the database for greater security storing user credentials, introducing more specification options for the forecasting and simulations, and widening the scope of regions accessible to users for analysis, for example widening the service to the entirety of Europe or worldwide.

Beyond adding additional functionality to the users, the other primary change that would be top priority in future development cycles would be to improve the methods with which energy output is calculated. At present, due to time constraints as well as the modelling constraints listed in Chapter 2, a simple methodology for calculating energy output is used, with weather data generalised to a county level and

An additional service that could be added in future iterations of the program that is outwith the initial intended use cases of the program could be to add a functionality to allow users to access historic weather data. This function would allow users to map previous weather trends, and thus allow them to make more informed decisions in their settings used in simulations, providing a function that these users would probably use another platform for. By drawing in the functionality of other programs that are likely to be used in tandem with this program user base can be expanded, as the convenience of using only one platform for all tasks is always desirable.