SOLUTION EVALUATION

Blue Dog Café Rostering System

ABSTRACT

The following report assesses the effectiveness of the project plan and lays out a strategy for evaluating the effectiveness of the solution.

Jules Carboni

VCE Software Development

Contents Page

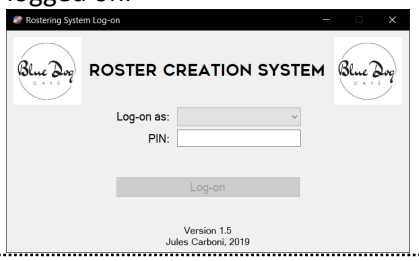
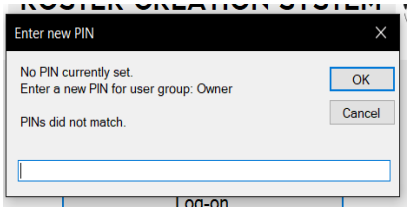
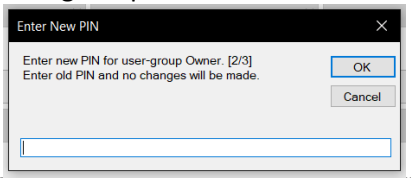
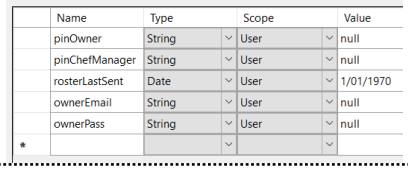
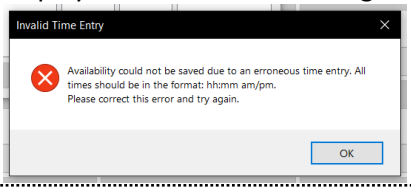
1. EVALUATION STRATEGY

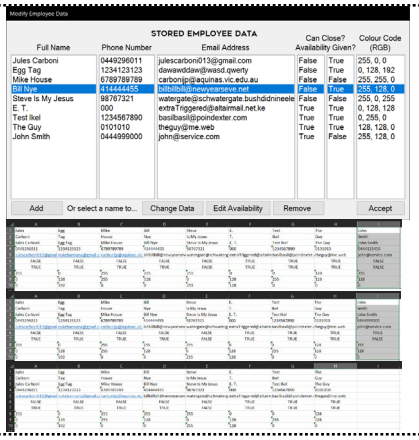
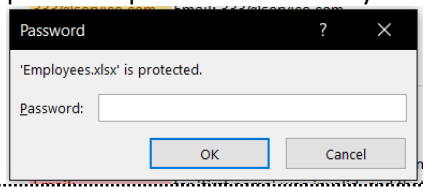
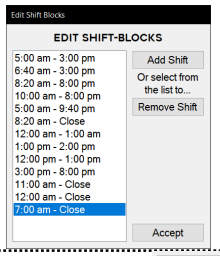
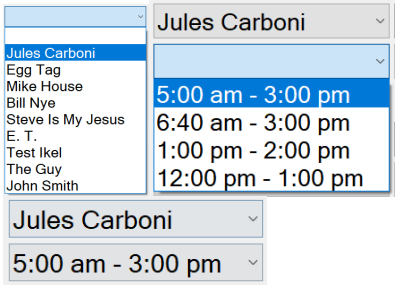
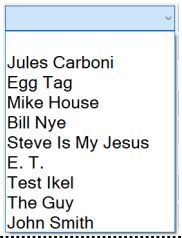
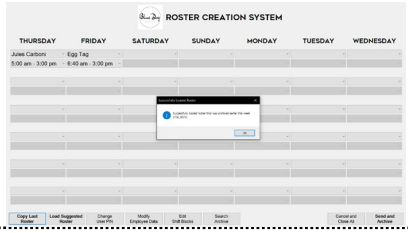
Time Frame	Description	Efficiency and Effectiveness Measures
Prior to signing off on the finished software.	Usability/Acceptance testing completed by client and some other testing users of the solution. The affected users include the client (as in the owner of The Blue Dog Café) and the testing users (who remain anonymous, of which there were two). The testing methods utilised are: - Observation (including thinking aloud). - Questionnaire. - Interview (as a follow-up).	Efficiency: - Can employee data be added/edited/removed? - Can the times of the shift-blocks be edited? - Can last/this week's roster be copied for use as a template? - Can rosters be generated/created and saved as PDF? - Can rosters be suggested to the owner user-group? - Are rosters sent to employees via email? - Can archived rosters be searched (by employee name)? Effectiveness: - Are the forms easy to navigate (buttons in logical places)? - Do the buttons accurately describe their functions? - Are useful error messages provided? - Is the UI basic (not graphics intensive, few visual elements)? - Does the UI follow the macOS design language (in terms of button positions and functionality)? - Are the UI elements big (with ample whitespace)? - Is it possible to send a roster within three clicks (not including generation)?
Immediately after the software solution has been installed.	Following installation of software on the client's device, ensure that the solution is operational (performs its functions correctly) and is ready to be used. The client (the owner of the business) will be present during installation and verification. Direct testing (an observation) of the solution being used for a brief period will be conducted.	Efficiency: - Can employee data be added/edited/removed? - Can the times of the shift-blocks be edited? - Can last/this week's roster be copied for use as a template? - Can rosters be generated/created and saved as PDF? - Can rosters be suggested to the owner user-group? - Are rosters sent to employees via email? - Can archived rosters be searched (by employee name)?

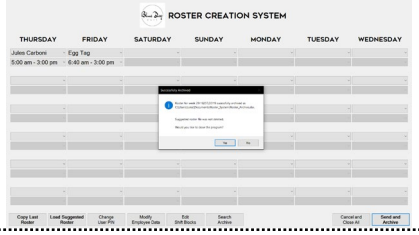
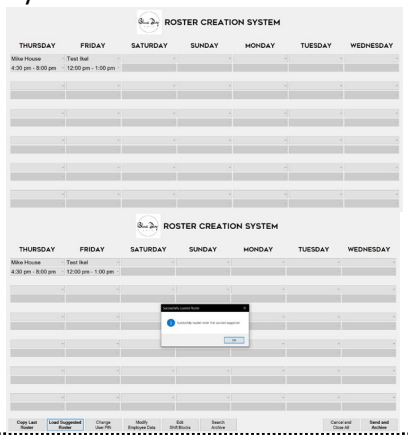
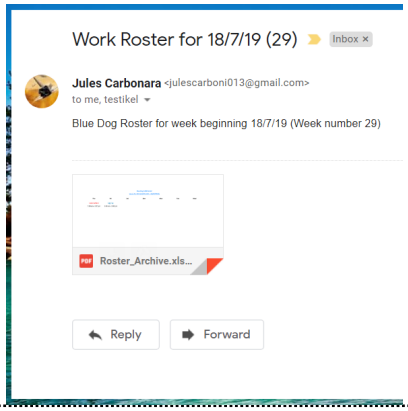
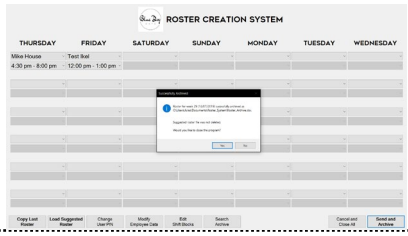
Three to five weeks after the software has been introduced.	Feedback regarding the solution will be gathered from the owner and any chefs/managers using the system. They will be questioned on how well the solution performs in the real-world environment and any shortcomings will be addressed. The owner (the client) and the chefs/managers will be involved, and the following methods will be employed: - Interviews. - Surveys. - Examination of software files and outputs.	Effectiveness: - Are the forms easy to navigate (buttons in logical places)? - Do the buttons accurately describe their functions? - Are useful error messages provided? - Is the UI basic (not graphics intensive, few visual elements)? - Does the UI follow the macOS design language (in terms of button positions and functionality)? - Are the UI elements big (with ample whitespace)? - Is it possible to send a roster within three clicks (not including generation)? Efficiency: - Can employee data be added/edited/removed? - Can the times of the shift-blocks be edited? - Can last/this week's roster be copied for use as a template? - Can rosters be generated/created and saved as PDF? - Can rosters be suggested to the owner user-group? - Are rosters sent to employees via email? - Can archived rosters be searched (by employee name)?

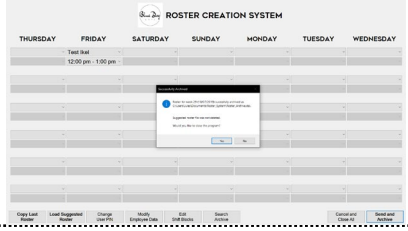
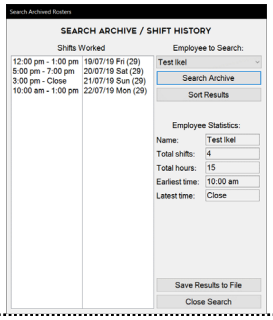
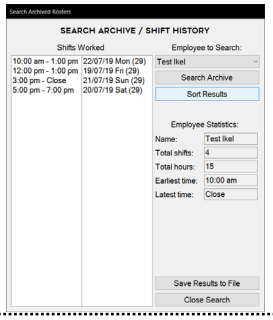
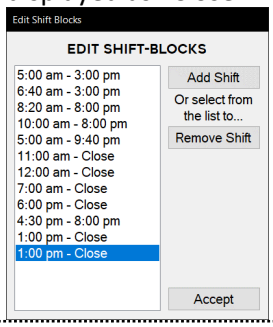
2. REQUIREMENTS EVALUATION

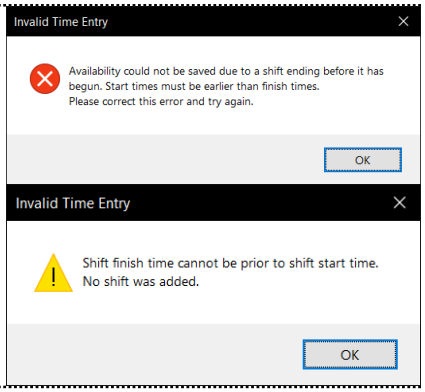
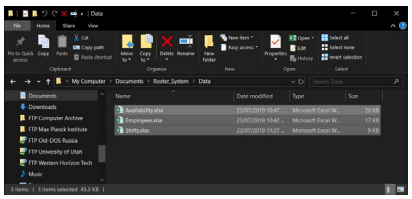
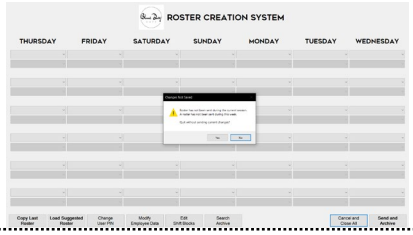
2.1 FUNCTIONAL REQUIREMENTS

No.	Requirement	Notes	How Requirement Has Been Met
FR01	Are users required to log-in with a PIN?	When the software is opened, users must log in as either an "owner" or a "chef/manager" before they are able to use the functions of the system. To do this, the user is required to enter a predefined PIN. This is to prevent the employee data being decrypted by an unauthorised user (therefore violating privacy regulations).	Program does not fully open until logged on. 
FR02	Are users prompted to create a new PIN upon first log-on?	If the file that stores the user-group's PINs is missing/deleted or if no PIN is stored within the file, prompt the user to create a new PIN upon first log-on. Though this is insecure, it allows the client to regain access to the system if their PIN is forgotten. *This should occur when the software is executed for the first time.*	PIN creation prompt shown when no PIN is stored. 
FR03	Can PINs be changed (of that user) once logged on?	Once a user is logged-on, they should be able to change the PIN for their user group (and only their group). This is in case the PIN becomes insecure.	PIN entry box appears on PIN change request. 
FR04	Are PINs stored in an encrypted file/secure state?	The PINs must be stored for next log-on and must be encrypted so that they cannot be read by anyone with access to the computer.	PINs are stored in the application settings, not in an external file. 
FR05	Is all input data validated?	When data is being entered into the software, it should be validated against a regular expression to prevent incorrect/corrupt data entering the system as it can result in the software behaving unexpectedly. This will require the use of the *Regular Expressions class*.	Incorrect data is rejected, requiring the user to retry before continuing. Error messages are displayed such as the following. 
FR06	Can employee data be added/edited/removed?	Take in personal data submitted to the owner by the employees and store it in a data store (spreadsheet) for use in the system. The data store must also contain fields specifying whether an employee is able to close the shop and specifying a "colour code" (see NFR04).	Changes are made through the employee data form. When data is modified, these changes are saved to file.

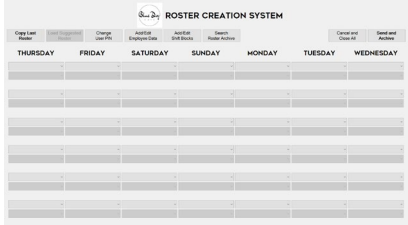
		Modify or remove employee data within the system. <u>Only the "owner" user-group can use this function.</u> *The term "employee" includes chefs/managers.*	
FR07	Is employee data stored in an encrypted file/secure state?	The spreadsheet containing personal employee information should be encrypted with a password to adhere to privacy regulations.	The employee data file is password protected with a key. 
FR08	Can the times of the shift-blocks be edited?	The owner user-group (only) can edit the times of the shift-blocks available for selection during roster generation. This includes a 'close' time which is used when a shift ends when the café is closing (a variable time).	Shifts can be added via the shift block form. Changes are saved to file. 
FR09	Can names and shift-blocks be selected from a dropdown list?	When generating a roster, employee names are selected from a dropdown list sourced from the employee data spreadsheet. Shift-blocks are selectable from a predefined list also.	
FR10	Is manual entry in fields prohibited during roster generation?	When generating a roster, employee names and shift-blocks are selectable from a noneditable dropdown (as above) to prevent invalid data entry (user cannot type).	The combination boxes disallow custom items to be entered. 
FR11	Can last/this week's roster be copied for use as a template?	When generating a new roster, it is required that the client can copy last week's roster to be used as a starting point so that they do not have to start from scratch each time.	By selecting the appropriately labelled button, the most recent roster data can be loaded. 

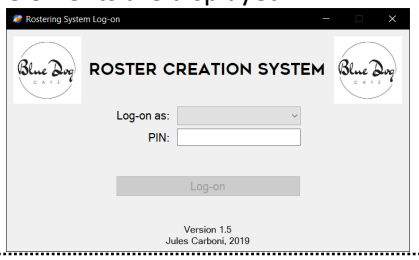
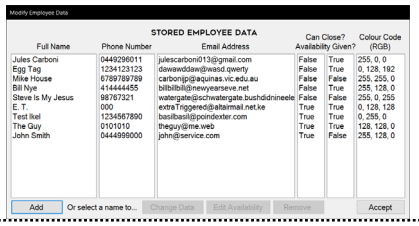
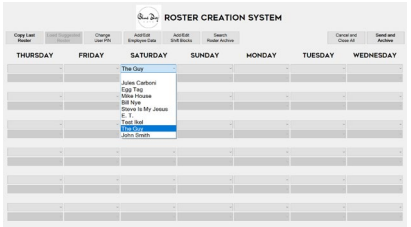
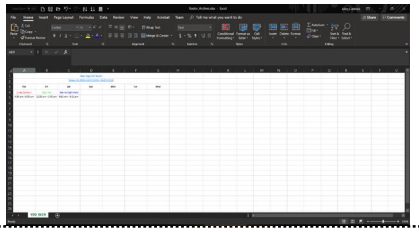
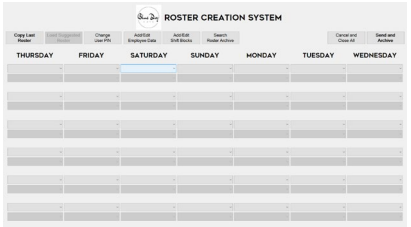
FR12	Can rosters be generated/created and saved as PDF?	The solution must be able to generate a roster *(see above)* in the form of an excel spreadsheet.	By selecting the appropriately labelled button, the current roster can be sent. 
FR13	Can rosters be suggested to the owner user-group?	When logged-on as a chef/manager, instead of being able to send a roster, users can suggest a roster. This means their generated roster is saved and shown to the owner on next log-on.	By selecting the appropriately labelled button, the current roster can be saved as a suggestion which can be loaded by the owner in a later session. 
FR14	Are rosters sent to employees via email?	At the same time as a roster is converted to PDF, a copy must be sent to every employee email listed in the employee data spreadsheet. The "chef/manager" user-group cannot access the functions for sending a new roster (the initial version/first for that week) but may modify, resend and re-archive a roster that has already been sent within the current week.	After generating/exporting a roster, it can be emailed to all the emails listed in the employee data file. 
FR15	Are rosters archived upon sending?	At the same time as a roster is sent out it must be archived as an excel spreadsheet in a folder so that a record is kept for the business. (Rosters are archived with no expiry/deletion date. No requirement for deletion functionality.)	After generating/exporting a roster, it will be added to/overwrite the latest roster in the archive. 

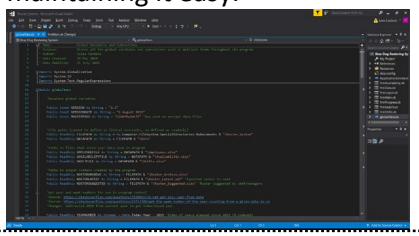
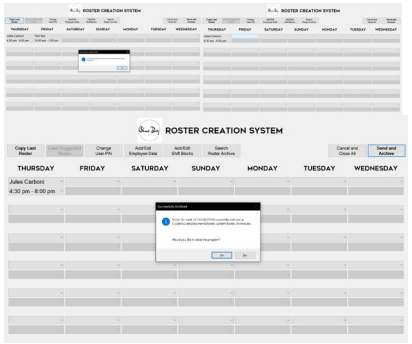
FR16	Are archived rosters overwritten if a new roster is archived in the same week?	If a roster is regenerated during the same week that the previous one was generated, upon archival the old roster must be overwritten.	Rosters archived within the same week will be overwritten. 
FR17	Can archived rosters be searched (by employee name)?	Past rosters can be searched with an employee name as the search string. It should return a list of all the shifts allocated to the employee name, sourced from the entire archive of past rosters.	The archive can be searched by name for shifts worked via the search form. 
FR18	Can the search results be sorted (by shift-start-time)?	Once the result of a search is returned, it must be able to be sorted by date—chronological order—or sorted by shift block starting time—from earliest in the morning to latest in the night. This allows the client to analyse the data in a meaningful way to increase the efficiency pertaining to the allocation of shifts on the rosters.	Clicking the sort button will arrange the results in order of earliest shift to latest. 
FR19	Are closing times past midnight interpreted as the “close” time?	When defining shift-blocks, if the user specifies a closing time later than “12 am” (midnight), the software should autocorrect the entry to the variable time labelled “close”.	When entering times (for availability or for a new shift) any time later than midnight will be displayed as “Close”. 
FR20	Is the user alerted and prevented from saving if they enter a finishing time earlier than a starting time?	When defining shift-blocks, if the user specifies a closing time prior to the opening time, show the user an error message containing details of the error and require that the user select another closing/opening time. *Do not save the ‘incorrect’ input.*	Error message informing user that finish time must be later than start time is shown. Availability/shift block is not accepted/saved.

			
FR21	Are missing files replaced with a blank file (or if first time run)?	If a file the software depends on (see above) is missing, create a blank "template" file consisting of the same structure to replace it. This file can then be filled out by the user as if the software was being run for the first time. *This should occur when the software is executed for the first time.*	Files are created as they are needed. 
FR22	Is the user prompted to send/save their roster before closing?	If the user attempts to quit before saving their work, display a confirmation dialogue box.	Warning shown informing user that they have not sent/saved/suggested their roster during the current session. 
FR23	Are useful error messages provided?	If the software catches an error, an understandable and useful error message should be shown to the user.	A wide range of messages are supplied to inform the user of the actions of the program. *Please see above examples.*

2.2 NON-FUNCTIONAL REQUIREMENTS

No.	Requirement	Notes	How Requirement Has Been Met
NFR01	Is the application always in full-screen mode?	The application should run in full-screen mode and should not be resizable. It should be minimizable.	The control box is disabled—the program cannot be minimised or resized. 
NFR02	Is the UI basic (not graphics intensive, few visual elements)?	Must not be graphics intensive as it may be running on old hardware.	The UI is clean with a clear focus on functionality of elements. There are no intense graphics or images, and only the necessary

			elements are displayed. 
NFR03	Does the UI follow the macOS design language (in terms of button positions and functionality)?	Should have similar layout, design, feel and behaviour of a basic macOS application. The client does not have an existing design language to adhere to but prefers familiarity.	Each window consists of its main group of elements which provide the main functions of the form, and all buttons that lead to separate forms with further functionality are in a strip in the upper or lower letterbox region of the form. 
NFR04	Are the UI elements big (with ample whitespace)?	The buttons, text-boxes, images, etc should be large and obvious to the user. It should not be hard to read screen content, nor should it be hard to find anything. Ample whitespace should be used between objects to aid the clean design.	Every UI element has a large font size, a large visual footprint on the form and is spaced evenly and with enough distance to be clearly distinguishable. 
NFR05	Are the employee names colour coded when they appear in the roster?	In the exported roster, employee names should be coloured according to the colour specified alongside their name in the employee data store. *See FR06.*	In the archive and in the PDF export the names are colour coded. 
NFR06	Is the company logo displayed in the top left of the main form?	The application logo should be the client's business' logo. The logo should also be displayed at the top of each form.	Yes. 
NFR07	Will the software reliably function (for five years) after deployment?	The software must be able to perform its required functions for at least five years from the date of deployment.	TBC.

NFR08	Are all errors caught and handled so that it does not crash?	The software should not crash at any time. All errors should be caught and dealt with.	Using Try Catch blocks allows the solution to not break and catch any logic errors. 
NFR09	Is the software easily updated/maintained?	It is important that the code is easily serviceable and updatable if, for example, the operating environment changes. This will reduce time and therefore the cost required to make changes to the system.	The code contains in-line comments and makes use of global constants to make maintaining it easy. 
NFR10	Is it possible to send a roster within three clicks (not including generation)?	It should take a maximum of three clicks (not including roster modifications) to copy last week's roster and send and archive the new roster.	Yes. The form is loaded, upon which the user can load the prior week's roster, make the relevant changes if necessary, and send in as little as two clicks. 

3. Evaluation of Project Plan

3.1 Introduction

A **final updated project plan** has been attached as a supplementary document in Appendix A. This plan shows the **actual durations of tasks** (annotated) versus the expected durations of tasks (printed) and the achievement of milestones. It has been **annotated** to show the adjustments made to the project timeline and other important dates (Easter!).

A **digital journal** was kept alongside the project plan which goes into more depth regarding the **progress that was made** during a given time period and what factors influenced its advancement/delay. This allowed me to log in more depth the causes and effects of changes made to the project plan.

3.2 Factors That Influenced Effectiveness

This project had a **clear scope** defined in the SRS; the features to be included and those to be left out were written unambiguously. The original scope was correctly translated into goals and specifications for the solution. All **tasks and resources** were identified in full.

Specification creep was not a factor that presented itself during development. There were no changes internal to the development team (no changes in staff—it's just me) and no changes to the external marketplace or environment. However, the client did request that a new feature (the ability to create two distinct rosters per week) be added to the solution. This request was declined as it would have added too many days to the development (which would have been unacceptable).

As there was only one developer, no time had to be allocated for handovers as a result of development **staff changes**.

Enough meetings with the client were scheduled to ensure good **communication** and that the development was on target to meeting its deadline. While no time was spent idle and unproductive, many hours were spent researching and exploring things that were not used in the final software solution; this wasted time.

Inadequate time for testing presented itself as an issue during development. Not enough time to test each individual module at various stages of development resulted in problems that amplified in later stages of development. Additionally, unnecessarily fixing modules that "ain't broke" in search of perfection used up time that was originally allocated to testing. This added the time required to complete the full solution testing later.

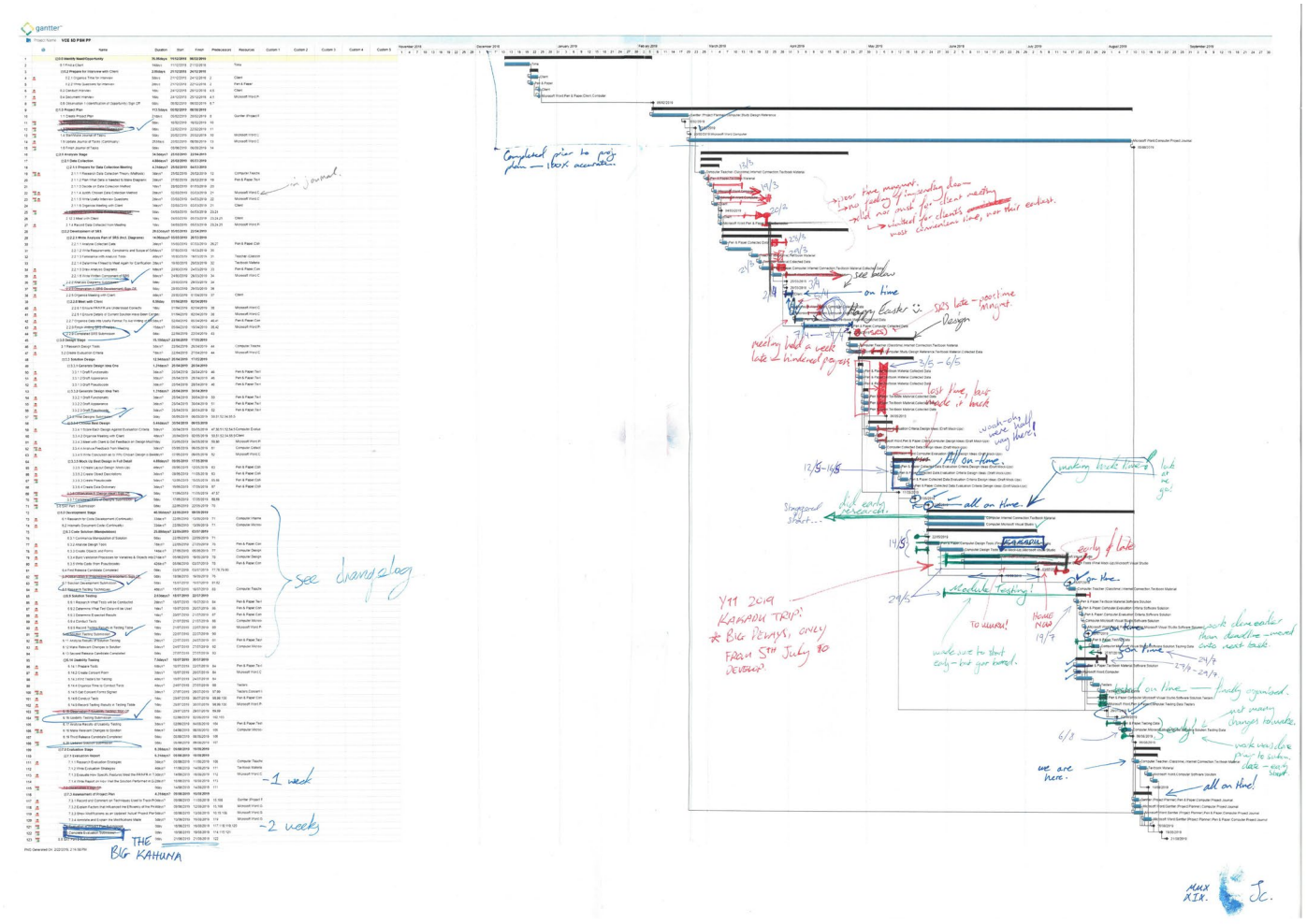
The Blue Dog Café Rostering System is not **dependent** on any other software packages, thus there were no problems in quantifying the slack time and effort allowed for debugging and no problems in communicating and collaborating with other developers.

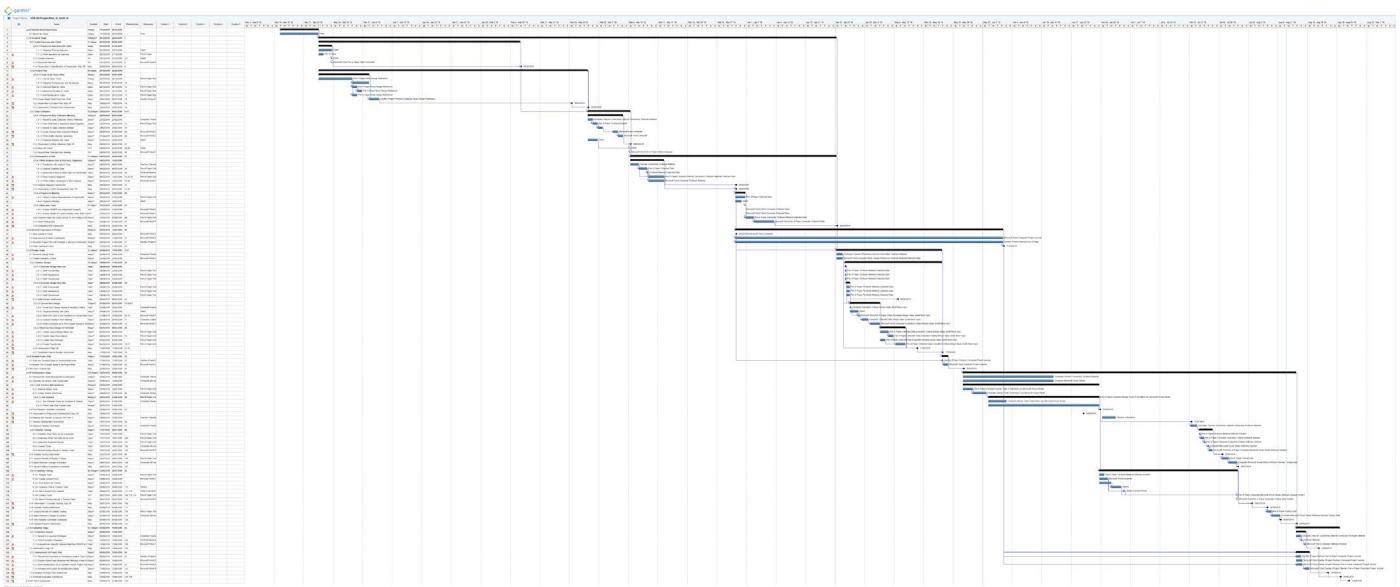
During development the client did not upgrade or **change their technology** (not hardware nor software), thus there were no compatibility issues and no time was wasted determining software changes.

3.3 Usefulness of the Initial Plan

The project plan was very useful in keeping track of the project; things would have gone a lot worse had it not been there to remind me of due dates. It allowed me to see clearly which tasks had to be completed and in which order—this made it easy to start the next stage of the project and not overthink it (even though sometimes I didn't start on time). The one thing the Gantt chart couldn't supply (which the criteria did) was reassurance that the tasks, once completed, were up to standard. By adjusting the initial plan, I could visualise the recurring reasons for delay during the project. This allowed me to make changes to avoid the same shortcomings in future stages of the project.

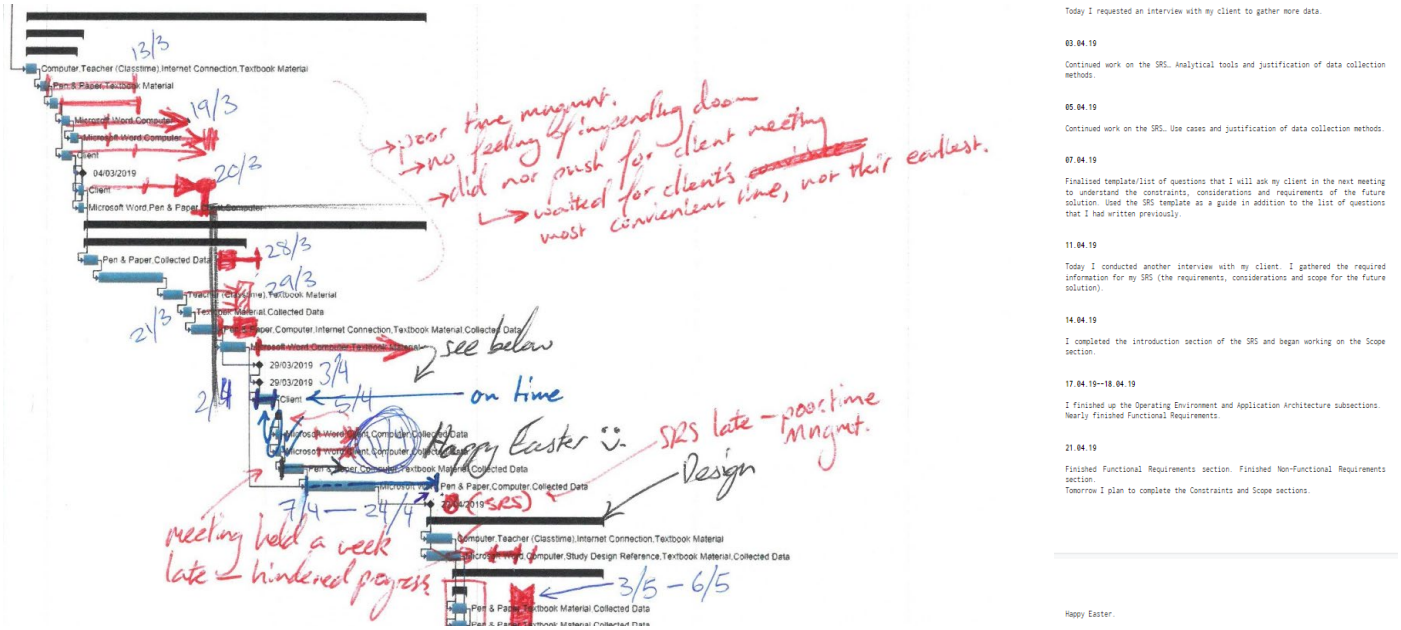
Below is the initial project plan (top) and the updated project plan including adjustments and annotations (bottom). A progress journal was kept during the entire project and a change-log was kept during development. Refer to these documents (attached in Appendix A) to view the changes made against the updated project plan. This journal includes comments and reflections on the causes and effects of delays or advancements.



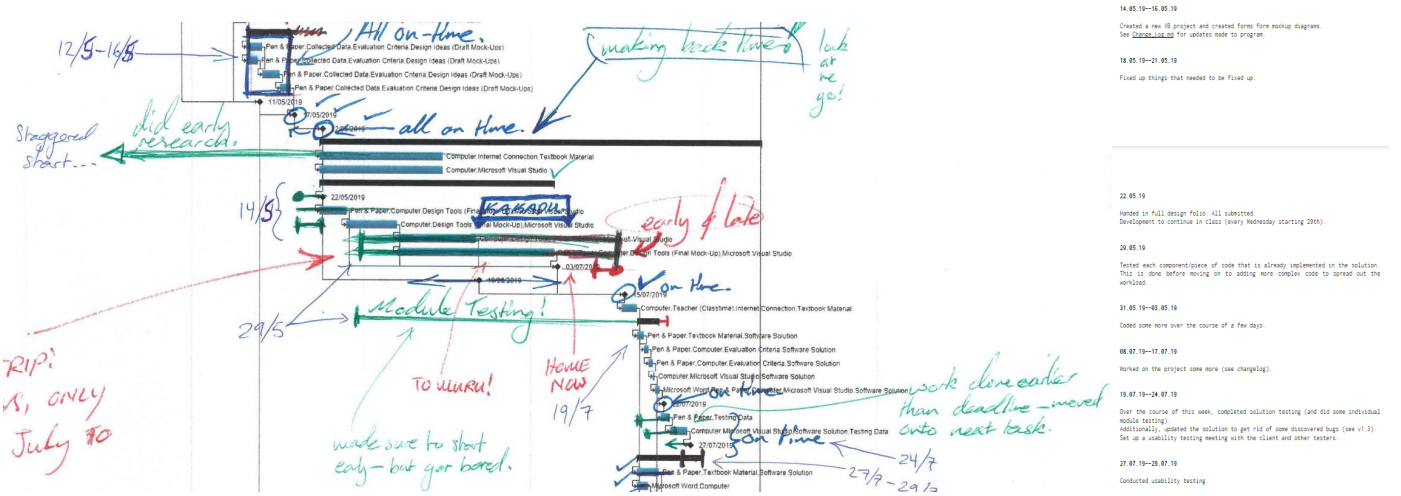


Below is an excerpt from the updated project plan showing and explaining delays via annotations. The corresponding journal date entries have been included alongside. The cause of the delays in this section of the project were chiefly because of my client's limited availability—her café was busy, and it was most convenient for her to postpone our

meeting by a week. In addition, it was noted that poor time management allowed the delay to snowball and have a slightly larger effect on the project—enough to ripple through to the next stage.



Below is another excerpt from the updated project plan showing and commenting on changes. This time however, the changes made were mostly advancements. Due to my limited knowledge of the VB.NET programming language, I made the decision to begin programmatic research early in the project, before development, so that when the time came, I would be well equipped to code the software solution. This was doubly important because of the planned school trip to central Australia which would disable development for two weeks. A similar decision was made to begin module testing during development with the goal of reducing the load of testing post-development. (This decision did not last though, and module testing was not consistent.) The delayed sections of this excerpt were caused by stagnation and the school trip. The period of absence is labelled “Kakadu” and the few days following this period were also unproductive.



3.4 Effectiveness in Monitoring Progress

- The project **finished on time** according to the project plan.
- Setting up meetings with the client and debugging **delayed the project** the most.
    - The longer times required to set up meetings was not anticipated as the client’s schedule changes sporadically depending on how busy their work is (and thus how much free time they have).
    - The delays caused by debugging was anticipated, though due to the untimely scheduling on the central Australia school trip, it was not possible to allocate any more slack-time to the development stage.

- **In future projects** I will apply a more rigorous planning and time management scheme and stick to it more closely. I will ensure that every interruption is foreseen (within reason) and is planned for, allocating slack-time where necessary.
- **No new requirements** were added, and no ambiguous requirements were left not catered for. Each requirement was described in depth and those that were too ambitious were designated out-of-scope.
- **My limited knowledge** of the VB.NET programming language was a limiting factor, however it was managed and removed by starting research regarding the features and techniques used in the language earlier in the plan than usual. This meant that by the time development begun, I had enough knowledge to successfully and efficiently complete the task.

A. SUPPLEMENTARY DOCUMENTS