# Evaluation

## Introduction:

To successfully evaluate my system I will evaluate each of my objectives comparing them to my system and saying what I believe I did well and what I did not do as well or what I would include in a different system going forward. Looking at each objective one by one helps to ensure that all aspects of my system have been evaluated and also by the use of heading I can evaluate different parts of one point including the design and implementation stages. I will use the objectives as the backbone of the evaluation to help form a plan for including all aspects of the system in the evaluation.

I will also compare my system with commercially available systems to see what I have done similar or what I have done differently for my system.

| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
|---|---|---|---|---|---|---|---|---|---|---|
| Allow the user to add a new employee to the system, entering key information including: Full name, Address, Telephone number and national insurance number. This will then be stored in a permanent file. | A permanent text-based file that stores the employee's Full name, Address, Telephone number and national insurance number. | When adding the information to the staff file I used a serial file I think that is good because it means it is easy to search for staff information and since there is only 2 members of staff it will be quick and efficient. | When adding the information to the serial staff file I used modular programming so that it was easy to write the information to the file – using a rewrite function. This means I could call the function as many times as I liked in all aspects of the staff section. Furthermore, it means it is much easier to add a new field in because you only need to add it to the readback and rewrite files rather than all the way through the program. | A good feature of the add function is that all the details are easy to input as they have prompts on any of the inputs which may be ambiguous in the format they should be entered. This is good because it makes the system easier to use and also makes the information quicker to enter for the user. | When entering the information for a new staff member as they are not familiar with the staff member, they could incorrectly enter the data and press enter and there is no way to go back and re-enter the data unless you go into change. | A strength in my design is the choice of inputs. I believe that this is a strength because without the correct inputs the staff information may not all be included and missing important details. However, due to my choice of inputs all an employee's relevant information is added. |  | When programming the add staff section I ensures that the basic additions worked and saved to the file before I implemented the validation. I think that this was a strength because it helped break down the problem and I wasn't confronted with lots of errors at once meaning it was more manageable. |  |  |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Permit the user to change information stored on an employee e.g., Address and telephone number. | Allows the user to change the employee's address and/or telephone number which is then saved to the permanent text file. | The use of the strcmpi in C++ allowed me to easily compare the staff reference entered and those in the staff file. I think that this was a good tool to use because it is an efficient way to compare all those in the file. |  | A good feature of the completed solution is that if information changes in the future such as they change their mobile number without the change function, they would have to delete the staff member and re-add them. | When the user enters the new data they can't go back and edit their input or check its correct before submission – it just happens. This is a short coming because it means that incorrect data | I think a strength in my design was the pseudocode I created for the change function. This helped me to easily code the change staff function speeding up the development process. Another strength in my design was |  |  | During programming, I struggled with the change not saving the new home address this caused me to spend a lot of time trying to find a small error as I mixed up which way the cin.get() function takes the input and it actually skipped over the input I required. | Ensure cin.get() is placed in the correct place and does not interfere with any input. Furthermore, I would ensure they are not just entered unnecessarily. When changing information, I could output the |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | The for loop was also a useful technique to use because when the reference is entered by the user it allows the code to be repeated for the amount of staff stored in the file. This is useful because it means that the reference is definitely checked against all those in the file before it is either accepted or an error message is produced.<br>The if statement technique allows reference to be compared against references in the file and depending on the outcome it can either be accepted or move on. This is a great technique because it means that something always happens such as it is accepted or an error message is produced meaning the user is never left in the dark about what is happening behind the scenes in the code. | | However, with the change function it is much easier since the user can now just change the one field which is much more efficient. Furthermore, a good feature of the completed solution is that I allow the user to confirm it is the correct staff member before they input any new information this means that they are least likely to make a mistake. | might be being entered. | the UI. I believe that this is a strength in my design because when the user selects the change function it gives them the choice of multiple different fields to changes in a clear format. This is a strength because it helps keep the system easy to use ensuring the user picks the correct field to change instead of getting confused. | | | | new information the user entered to check that it was all entered correctly before saving it to the staff file. This would improve the system as it would further reduce any accidental entries. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Allow the user to delete a member of staff that no longer works there. | Allows the user to remove a member of staff from the system and for the information to be deleted from the text file. | A c++ tool known as strcpy was very useful in the delete function because it allowed to copy all the values in the file and move their position up one to copy over the staff member to be removed. This was a good tool to use because it was an efficent way to copy data between variables. To allow me to move through the employees in the file and move the positions I used a for loop. I think that this was a good choice because it allowed me to only have to enter the code once but it also ensure all the staff members were included as it used the number of staff stored in the file to know how many times to loop through therefore I think it was a good tool to use because it means I was confident in the fact that the delete worked correctly. | The use of global variables to store the staff information helped me to create the delete function as it saved memory space since I did not to create local variables inside the function to delete the staff information they were already declared saving both memory space and programming time. | A good feature of the completed solution is that I allow the user to confirm it is the correct staff member before they are completely deleted from the file which helps to reduce any mistakes. Furthermore, the delete function is very important to the customer because if a staff member leaves then they can't store their private data so by having the option to delete a staff member they can clear that staff members private details. | When deleting staff the only way to delete them is by entering their staff reference however it may be easier for the user if the employee to be deleted is searched for using their last name rather than the staff reference. | | | During development I made sure that the delete function worked correctly before I included the checks that it is the right staff member I think that this helped me because it meant I could focus one piece at a time rather than an overwhelming amount to do in one function. | | I would produce another function which allows the user to delete a staff member by entering their last name. I would do this because the manager is more likely to know the last name of the staff member rather than their reference. This makes deleting a staff member much easier. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | the programming language | | | | | | | avoid the problem |
| Include validation on all data entered when adding a new employee. | National insurance number, telephone number, and the postcode will undergo format checks reducing the chance of incorrect information by producing error messages. | A tool I used in validation that was crucial is the isdigit() this helped to validation multiple functions including mobile number to check that all numbers had been entered by the user and no letters. This was a great tool to use because without it there would've been no way to check any of the digits were correct after 07. The tool 'strlen' was great in the validation of all the inputs because it allowed me to extract the number of characters stored in the input. This meant that I could check if data had been entered as if len was equal to 0 then I could output a message to the user telling them to ensure data had been entered. | The programming language was very effective for the validation functions as it allowed for parameter passing. This was very effective because it meant that I could take the users input from the add staff section and pass it to the separate validation function to allow it to be checked before it is committed to the staff file, allowing me to keep the adding and the validating separate. Furthermore, the modular programming was very effective since it allowed me to re-use functions rather than having to repeat the code helping to save time for me but also reducing processing times as the size of the overall program would be smaller. | When validating a good feature of my validation of the home address is that it does not require validation for address line 2. This is a good feature because address line 2 is not in all addresses so to require it would not make sense. Furthermore, I ensured that all my validation routines included contextual error messages I think that this is a good feature because it can help the user and reduce the same error being inputted however it was a generic message they could potentially keep entering the same thing. | | A strength in my design for the validation is my flowcharts. Creating these before starting any programming meant I had a basis to follow when I did begin programming rather than going in blind not having a clue where to start. This helped to improve my efficiency when programming and also to feel more confident when developing. | A weakness in my design was the validation table because not everything that needed validating was included on here as it is difficult to think of everything before you program it. For example, I said to validate address line 2 as I was thinking more validation rather than a real-life scenario. However, during my testing after the programming I realised my mistake and it was fixed. | | A weakness in my development is that I made a mistake through the validation on add staff where I did not set the variables inside the while loop equal to zero when I declared them. This meant that none of the validation routines worked however once I realised my mistake I corrected it and the validation routines worked correctly. I learnt from my mistake and did not do this again with a validation function. | |
| Allow the user to search for staff via reference. | Allow the user to search for a member of staff producing a list including their Name, Address, Telephone number and | An if statement was a great tool to use when finding the staff member to view because it allowed me to also use an else statement to | The use of the re-useable read back functions means that it is easy to call within the view function for searching the staff file. This helped speed up the | | Similar to delete staff, a staff members information can only be viewed by entering the staff reference rather than an option to view | A strength in my design is that I included sensible outputs for the view function which a user is likely to actually want to find out about an | | During development I tried to name the local variables so they were self-identifying I did this to help myself when programming so I | | I would produce another function which allows the user to view a staff member by entering their last name. I would do this because the manager is more |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | national insurance number. | output a suitable error message if the staff reference entered can't be found. Where as without it would be more difficult to compare the input with the staff file. | programming of the function as all I had to do was call it. | | them by last name. I think that this a negative of the view functions because Stuart is more likely to remember their name rather than a reference. | employee. I think that this is a good feature because it means only the necessary information is outputted saving processing time since only required variables are outputted. | | knew exactly what variable I was using but also when I had to go back to the function later on the variables all made sense making the code easier to remember and understand. | | likely to know the last name of the staff member rather than their reference. This makes viewing a staff member much easier. |
| I am very pleased with the staff section of my program as whole. All of my objectives were taken into account therefore creating a functioning aspect of the system perfect for accurately storing all staff information.<br>The use of a serial file was a great choice since there will only be a small number of employees so processing times will be very low for the company to help keep their system quick and efficient.<br>The menu driven interface and prompts on all functions were a great aspect of the system helping to make it easy to use for all, and quick to adapt too due to the helpfulness of the UI and also the simple numbering system to tell the system where in the staff section they want to access. | | | | | | | | | | |
| | | | | | | | | | | |
| Allow the user to enter a new customer to the system by storing: Full name, Address, and a basic description of the job in a permanent file. | A permanent text-based file that stores the customer's Full name, Address, Telephone number, and a basic description of the job. | For adding and storing customer data I used a random access file. I believe that this was the best choice because the company will stores lots of customers and lots of information so a serial file would've took longer to load a customer when searching however a random access file can jump straight to the correct record this is good because it saves processing time and speeds the system up for the company. | The programming language allowed me to clearly show to the user what inputs are required and where this helps makes the UI clearer and the data entered is less likely to be wrong if there are prompts to help them. This helps to improve accuracy and also save time as they won't have to keep re-entering things if the prompts clearly tell them what to enter. | A good feature of the addition and saving of customer information is that now it is stored to the customer file the system can use it in different aspects of the system such as linking it to quotes and invoices. This is a good feature because it means that the staff can relate a quote to a customer to find further contact information for their bookings. | | A strength in my design is the use of the problems and sub-problem table. I think that this was a strength in my design because it allowed me to break down the function in a more manageable way helping to keep the development of the add customer function later on a much easier task. | A weakness in my design was that I included a basic job description as a required input however this is entered under the quote seen as it was quote information rather than customer information. This is a weakness because it is wasting storage space and also the users time because they are having to enter the same information twice. Also, a customer may have more than one job so the job description would not be the same every time so this | | A weakness in my development is that when adding a customer at first I forgot to check the value of the flag for the reference entered. This meant even the reference entered was already in use it would right over it because I was not checking the value of the flag. However, I have now sorted this so the customer reference is checked before addition in the add function. | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | would not be an effective input. | | | |
| Allow the user to delete a customer from the system when they are no longer using the company. | Allows the user to remove a customer from the system and for the information to be deleted from the text file. | When deleting a customer I used the strlen command which extracts the number of characters in the input. I think that this was a good command to use because it allows the software to check data has actually been entered and alert the user if this is the case. This is important because it means that the delete function will run much smoother compared to if I didn't use strlen and there was no reaction from the delete function if no reference was entered. | | A good feature of my delete function is the schedule check. I think that is a strong feature of my program because it checks that any quote reference linked to the customer being deleted is checked across the schedule and if the quote reference is found on the schedule an alert message is outputted to the user to say they have been found on the schedule. I think that this is important because it helps to reduce any chance of the wrong customer being deleted or even the fact, they may have forgot this customer is booked in if it was booked a while ago. | When the user deletes a customer and confirms it is the customer to be deleted nothing alerts the user to let them know if it was successful or not. I think that this is a negative of the function because it makes it unclear to the suer whether or not the customer has actually been deleted so the user has to trust the system has deleted them since not even a small message is outputted to the screen, it just returns to the customer menu. | A strength in my design is the choice of file. The random access file allows the system to jump straight to the record and delete only that customer's information unlike the serial file which has to move all the positions and if there is a lot of customers this could've took a long time therefore I think that the choice of file in my design was a strong point. | | | During development I used the strcpy tool to convert the value of the flag once the customer has been deleted. However during development I did this incorrectly as I saved the flag as 1 instead 0. This was a weakness in my development of the delete function because it means instead of deleting the customer it saved there details to the file. By changing the value to copy to the flag to 0 I overcome this and the delete function worked. | A suggestion for improvement would be to include the message discussed in the shortcomings of the program which checks the customer was actually deleted because it helps makes the system clearer for the user and less stressful as if the customer details were kept rather than deleted this could cause gdpr issues. A suggestion for improvement would be, similar to on the view function, allow the user to delete a customer by entering their last name. I think that this would be an improvement because it may be easier for the user to remember a customer's last name compared to a reference. Furthermore, by having both it gives the user the option since different employees may prefer different ways of doing things. |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Allow the user to change the address of a customer stored on the system. | Allows the user to change the customer's address which is then saved to the permanent text file. | The use of the tool 'strcmpi' allowed me to compare the last name entered by the user and those stored in the file. I think that this is a good tool to use because if there is a match it must be definite as the comparison are coming from within the customer file. Furthermore, from the use of strcmpi I can output further information on the customer to allow the user to check it is the correct one because the strcmpi tool has already found the customer in the customer file so it was easy to output further information. | | A good feature of the change functions is that instead of asking the user to enter the reference they enter the last name of the customer. I think that this is a good feature because an employee is more likely to remember the name of the customer compared to a reference. Similar to staff change another great feature of the completed solution is that without it a customer would have to be deleted then re-added therefore this feature is great in saving the users time. | An imperfection of the function is that when the user types in a customer that's information does not need changing and types N to confirm it is the wrong customer it does not give them the option to re-enter a last name but instead takes them back to the change menu. This is an imperfection because if the user was in the change function it is likely they needed to change something they just entered the wrong customer. | I think a strength in my design is the usability of the change function for the user. The UI is clear and easy to understand with suitable prompts to guide the user through the function this is a strength because Stuart and his employees aren't used to a computer system so the easier it is to follow and understand the better for his company personally. | | The choice of modular programming during development was a great choice because it means that in the change function instead of requiring lots of repeated code for validation, I can call the validation function which can then be used to test in multiple aspects of the system. This is also good because it helps to reduce errors since rather than having the same error repeated everywhere, it is only once and can be quickly modified. | | When a user enters a customers last name if they enter a single character off the system will produce an error message to say that it is wrong therefore a suggestion for improvement would be to extract the first few characters of the one entered and those in the file using sscanf and compare them using strcmpi to see if there was any similar and id there was produce an output to ask if the user meant to enter that customer. |
| Enable the user to search for customer using name or customer reference. | Allow the user to search for a customer producing a list including their Name, Address, Telephone number, and a basic description of the job. | To help improve the quality of the search function I used the 'strlen' tool this was useful because it meant I could check that the user had actually entered data this improves the function because it means I can use more contextual error messages rather than one overall generic | | The layout of the view function is good because it allows the staff members to easily see the information that they need due to the clear headings implemented in the UI. | A shortcoming of my solution is that I do not link any of the customers to their quotes. For example when an employee views a customer they only see customer information however due to links already made in the file I could link in quote | I think a strength in my design is my choice of outputs for the view. I think that this is a strength because my choice of outputs allows the user to see all the relevant information for a customer since the view functions job is to show all important | | | During development I struggled with the use of cin.get(). I struggled because of where I placed it meant that the name entered by the user was skipped over and instead searched for nothing. To overcome this I used dummy prints to locate why no information was outputted which then showed me that my input was coming up as blank this lead me to the cin.get() which I then moved so that the input | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | message at the end. | | | information such as the job description to help the employee match the customer to the jobs. | information about a customer. | | | was stored in the local variable 'name'. | |
| Ensure all data entered is validated. | Validate the address and telephone number using a format check, producing an error message if incorrect. | A tool I used on the validation functions that was important was a while loop. This was a good tool because it meant that the validation routine would run until data of the correct format was entered meaning no invalid data could be saved to the file.<br>Furthermore, tools such as atoi and itoa allowed me to smoothly transfer between the two data types, character and integer, since may references were stored as integers however to be able to validate them completely they needs to be in a character format therefore the itoa and atoi tools became very useful throughout all the validation functions. | The modular programming used in c++ was very effective. This is because it allowed me to program one validation function but then use it throughout the adding and changing of the customer information. This will also be useful if Stuart wishes to make any changes to the validation on something as he will only have to make the change once rather than a repeated number of times. | A good feature of my validation is that all the error messages are contextual. I think that this is important because for my system the employees aren't used to the computer so if they enter something incorrect the more help they can retrieve the better. This will help both them and the system to run smoother since over time they will become more used to what they need to enter and what they can't enter. | | A strength in my design is that I ensure that every input within the add customer function was validated in some way. I think that this is a strength in my design because it increases the chance that the data has been entered accurately. Furthermore, it ensures data is entered for all inputs other than address line 2. I think that this is important because it means that the system stores all information about a customer and there are no blank spaces of information that could've been crucial to a staff member. | | A strength in my development was that I worked through the validation step by step doing one input at a time. This was a great way to do it because it meant that I could focus my time into each function making sure it all worked correctly before moving on. Furthermore, by doing it input by input it made it more obvious what the error message were for since I was only working one input at a time this was good because it meant I only had to try and find the bug in a small area of code rather than a whole function. | A weakness in my development was the postcode validation. When programming I became very confused with the many different formats to try to help this I went through each postcode format logically ensuring I had included it and that the correct error message was being outputted.<br>Furthermore, I had issues with the declaration of the variable size. For example, the mobile number was declared to store up to 12 characters so if a user entered anything over the character limit the program would crash. I fixed this by included a buffer solution which allows a much larger amount of characters so the chance of the program crashing is significantly reduced. | |
| Overall, I am very happy with the customer section, all objectives were met, and the system includes all the crucial features of where the customers may be needed meaning Stuart's company can easily transfer from paper based without worrying aspects of their company are missing from the system in relation to the customers.<br>I believe that the random file was the best fit because the company is likely to store many customers so a serial file may have taken a long time to load whereas the random file can swiftly located the record without having to search through every customer – this is a great way of improving processing times to help make sure the system runs efficiently. | | | | | | | | | | |
| | | | | | | | | | | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Allow the user to add a new quote to the system by storing it in a permanent file. The information to be stored on a quote will be: Date the quote was produced, customer reference, a job description, the number of days required to complete the job, a basic idea of what stock will be needed and an estimated price of the job. | A permanent text-based file that stores the following information on a quote: Date the quote was produced, customer reference, a job description, the number of days required to complete the job, a basic idea of what stock will be needed and an estimated price of the job. | A tool I used throughout the add quote function is itoa. This was a great tool to use because it allowed me to convert the variables between characters and integers to use in the calculation to find the updated price but then be able to convert them back to characters for storing in the text file.<br>Strcpy allowed me to use temporary variables to check that the data was within the buffering limits and if the validation came back correct it meant I could copy the value to the variable in the file. | The programming language allows me to keep the calculation separate to the add function. This was effective because it meant that I could concentrate on checking that the variables added to the file correctly without having to worry about the calculation as this was in a different function. | | An imperfection of the add quote function is that when you enter a customer reference there's no way for the user to check if it's the correct customer to link they just enter the reference and that is it. This is a shortcoming of the solution because an employee may get confused and link the wrong customer to the system which can then not be changed once entered | A strength in my design was the choice of inputs for add quote. During design, I included the customer reference as an input I think that this was a good point because it means that the customer is automatically linked to the quote so that when a quote is viewed the staff can see who the quote is for rather than a quote with no attachment to any customer. This helps to keep the system organised. | | A strength in my development is that I ensure the VAT was only included in one line of calculation. This is a strength because if the VAT needs changing due to the way I programmed it they would only have to change one value on one line of code meaning it would not be a difficult job and would not cause great difficulties. | A weakness in my development was that I chose to include the validation after the addition this was a good thing to do because it meant it was much easier to breakdown problems. However, it meant that when I accidentally entered a customer reference which didn't exist it was accepted since there was no validation. So I spent time trying to find a reason why no customer information was being linked. This then became a weakness on add quote since I could have spent my time elsewhere in the program. Once I realised what I had done I entered a customer reference definitely stored in the file and found that the add quote was working correctly. | A suggestion for improvement would be to include a check that the customer reference entered was the customer to definitely to be linked to the quote. I would do this by outputting the customers name and mobile number associated with the reference and similar to change and delete ask the user to input Y or N depending on whether or not the customer was correct. If Y then I would allow the input to be saved to the file and if N I would ask the user to enter a new customer reference. |
| Enable the user to delete a quote once the booking is finalised. | Allows the user to remove a quote from the system and for the information to be deleted from the text file. | The use of a for loop is used multiple times in the delete quote function as a tool. I think that it was a great tool to use in both cases. Firstly, because it meant that the quote reference entered could be compared with every quote reference stored in the file | The programming language allowed me to call the read back and re write functions. This was effective beccause it allowed me to easily retrieve the information required for the file but also to re-write the new format to the file with the quote entered deleted. | A strength in my program is that I allow the user to confirm it is the correct quote. I think that is a good feature because it reduces the chance of the wrong information being deleted as the user can see the quote information and read it | | A strength in my design is the way I broke down the delete quote function into problems and sub-problems. I believe this was a strength because a sub-problem was to ensure the delete function was clear to understand. By breaking it down | | A strength in my development was my choice to break down the function into different sections. I think that this is a strength because it allowed me to work through the function checking each part was working step by step rather than finding lots of | | A suggestion for improvement in the delete function is that when an employee goes to delete a quote they can see the customer linked. However, what if the employee wanted to delete the customer too? Therefore, a suggestion for improvement |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | however I only had to write on piece of code this is good because it saves time and is much more efficient. Similarly, a for loop is used to delete the information stored for that quote by looping through all the positions in the file and moving their positions so the quote would be overwritten. This was a good tool because it meant I could easily control all quotes in the file as the for loop looped through the required quotes. | This also helped to save me time during development as I was not continually writing the code to read and write to the files. | themselves before any information is deleted. | | in this way it ensured I remembered through development to make it easy to access the function through the menus and, the user prompts as clear and easy to understand as possible so that the chance of mistakes being made was low since once the quote was deleted you would not be able to get it back unless it has been backed up. | | different error which I had to then all solve all at once. | | would be to produce a message to the user asking if they wanted to also delete the associated customer. This would help to improve my program because it would be another way to help Stuart and his staff save time. |
| Allow the user to change information on the quote including the estimated price and the number of days required. | Allows the user to change the number of days and the price quoted which is then saved to the permanent text file. | The atoi tool was very useful in both change functions because the inputs by the user are stored as characters however, for the calculations in number of days and the range check for the price they need to be integers for it to work correctly. Therefore, the atoi tool was a great tool to use because it helped me to improve the quality of my function by allowing me to | The programming language requires all variables in the calculation to be the same data type. This meant that I had to ensure in the change that I was always using integers within the calculation and nothing else because if not the calculation would not work correctly. | A good feature in my change function for the number of days is not only does the function change the variable which stores the number of days but it also updates the quote of the price accordingly. I think that this is good feature of my function because the employee now doesn't have to change any of the quote prices as it is already done for them. This is good because it | A shortcoming of my change number of days feature is that when the function changes the labour costs it does not take into account the extra materials used. I think that this is important to my program because stock is expensive and if the extra cost is discarded the company will lose money. | | A weakness in my design is that on the change when a price was entered it has to be just an integer. However I think I should have designed the input so it would be NN.NN as this is a much more likely price compared to a whole number. | | | A suggestion for improvement would be to include the price of materials in the updated price because if the number of days is changing this means that the stock required will either increase or decrease so the material cost will therefore need adjusting. |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | validate the inputs and perform a calculation. | | allows the employee to do other more productive things such as answering customer calls or booking in jobs. | | | | | | |
| Enable the user to search for a specific quote using quote reference or customer reference. | Allow the user to search for a quote producing a list including the customer information, the basic stock required and the estimated price of the job. | The for loop is very important for the search as it allows the function to lop through all the quotes in the file and compare it with the one entered to output the correct quote. I believe that it is a good tool to use for my function because without it, it would be more difficult to find the correct reference. | | A good feature in my system is that when a user views a quote not only does it show the quote information but also the customer information linked to it. I think that this is a good feature of my system because when a staff member views a quote the more information they can see the better as it will help to remind them of the booking when the customer rang originally. | A shortcoming of my system is that similar to view customer my system only allows a user to search by a reference however it may be easier for a user to be able to search by a customers last name because this could be much easier for a member of staff to remember compared to a number. | I think that a strength in my design is the choice of outputs for the search. I think that this is a strength because if I had missed any important outputs then the staff would struggle to make sense of the quote however by including all the outputs I chose the view quote will be a useful function for an employee. Another strength in my design was the entity relationship diagram this helped to remind me what I would need to link from other files. | | A strength in my development is that when I link the customer information to the view I chose to do this in a separate function. This is a strength for more than one reason. Firstly, it means that if I want to call the function in other aspects of my program I can do. This is good because it saves me time when I am programming as I would not be unnecessarily repeating code. Also, this means that if I want to change the information outputted by the user I can do this easily in one function rather than having to change the variables outputted in multiple different places in the program. | | A suggested improvement would be to allow the user to search for a quote by entering the customers last name. I think that this would be an improvement to my system because if a customer was to contact about a quote that was made the user would only have to ask for their last name and they would be able to view the quote straight way. I think this shows that it would be an improvement because it helps make the company look more professional but also for the system to be more effective for the needs of the company. |
| Produce a calculation for the quote taking into account current stock prices, | Allow the system to calculate the cost of the quote and then show the cost in an easy-to- | | During the creation of the calculation the programming language caused | A good feature of the calculation is that when the stock is added and the quantity is | A weakness in my system is that I use integers throughout. I think that this is a | A strength in my design was the pseudocode. I believe that this was a strength | | | A weakness in my development is that when I included the VAT costs for the price I was adding the incorrect VAT as I had | A suggestion for improvement would be to convert all the integers to floats |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| labour costs, number of days worked and mileage. | read table breaking down the cost and including the VAT. | | me some issues. For a function in c++ only one variable can be returned. This became an issue because I wanted to return multiple variables during the function. To overcome this I had to split the function up into multiple smaller functions that are all interlinked. | entered the stock levels automatically decrease. I think that this is a great feature because it means the staff don't have to worry about the stock levels as the system takes care of it for them. | weakness because to get a more accurate price I should have used floats for the variables because this would then allowed me to have the price as NN.NN rather than just NN. | because by having the pseudocode it allowed me to feel more at ease when starting the actual programming because I had step by step lines to follow which made the whole process a lot less overwhelming. | | | not properly researched what VAT to add so the overall price of the quotes were incorrect. However, once I changed this the calculation worked correctly. | to allow for a more accurate price overall. Furthermore, when a user enters the travel costs there is no guidance on what to enter it is all down to the enter. So instead, I would work in further detail to work out the average price of petrol per mile and ask the user to enter the number of miles to the destination instead, further improving the accuracy of the price. |
| Validate all quote information upon entry. | Validate the date entered and the price using a format check, producing an error message if incorrect. | An important tool I used throughout the validation functions was 'strcmpi'. I believe that this was a good tool to use because it allowed me to check that any references entered were correct in the sense of they were unique or that the customer did exist. Furthermore, it allowed me to improve my validation on the type of paint because I could compare the input to specific key words I had chosen which will | | A good feature of the validation in my quote function is that when all the inputs are validated if the user enters something incorrect all the error messages that are outputted are suitable for the input. For example, if it is a range check it would tell the user the accepted range. I think that this is a good feature because it helps to keep the system easy to use for the member of staff as they get used to the system. | An imperfection in my validation in my system is that when a user enters the stock to be included in the quote it allows the quantity of the stock to fall below zero. However, this would not be very useful for the employees as they would presume they had all the relevant stock therefore I believe that this is a shortcoming of my solution. | | A weakness in my design was the validation table because not everything that needed validating was included on here as it is difficult to think of everything before you program it. For example, I did not say that the stock references on add quote would need validating to check they existed however this is crucial for the quote calculation since materials | A strength in my implementation of the validation functions is that instead of a simple presence check fo validation I included validation so only certain types of paint could be accepted. This was a strength in my development because it meant that the validation is even more accurate. | | A suggestion for improvement is that when the stock reference entered in add quote is validated the system should check that the quantity will not drop below zero when added to the quote or it could output a message to the user telling them the quantity of that item of stock before they add to the quote so the user can check that they have all the stock required. |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | be ideal for my system helping to keep the system unique to this company rather than a generic piece of software. Also, you can see further tools and techniques used above in the staff section of validation. | | | | | that don't exist within the system cannot be added to the quote price. | | | |
| As a whole, I am very happy with the quote function I believe that it does every objective I set out to do. However, I think that are always ways to improve a function such as the use of the float data type in the calculation.<br>I think that the quotes aspect of the system is ideal for the company I have designed it for because all the menu options are clear as to what they do and when a user is asked to input any information it is clear what they need to enter helping to keep the system easy to use as they will not be used to the computerised style. | | | | | | | | | | |
| | | | | | | | | | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to add additional stock to a permanent file including information such as: colour of paint, type of paint, volume of can, price and quantity. | A permanent text-based file that stores the following information on the stock: colour of paint, type of paint, volume of can, price and quantity. | Strcpy was a great tool to use in the addition of stock because it allowed me to pass a temporary variable for validation then if the input was valid copy the value to the variable associated with the stock file since the global variable had a limited character size so when it automatically saves to this program can crash. | Similar to above in add staff and quote. When adding the information to the serial stock file I used modular programming so that it was easy to write the information to the file – using a rewrite function. This means I could call the function as many times as I liked in all aspects of the stock section. | The addition of stock is a great feature because without it when a staff member prices a quote the materials would not be included however this is a key part of a quote price meaning the addition of stock is a great feature within the system. | | As there are many inputs which could be included in the addition of stock the variable table I included in the design stage is definitely a strength because it ensured I included all the relevant information I needed about an item of stock and that nothing was missing like if I had done it off the top of my head when programming. | | Similar to the other add functions I broke down the addition function by ensuring the basics of writing to the file worked first before adding in the validation. I belie that this is a strength because it allowed the development to run a lot smoother with a lot less errors. | | A suggested change would be on the outputs for fields such as stock price, the the input must be a whole number since a decimal would cause issues later on when being used in a calculation. Since this is not made clear to the user in the current program, I would change this and add a small output message using cout. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to delete stock from | Allows the user to remove stock | Similar to delete staff and quote I | A great feature of the programming | The delete function is a great | | A strength in my design is the | A weakness in my design was | Furthermore, similar to the | | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| the system if it is no longer required or is no longer sold. | from the system and for all the information related to be deleted from the text file. | think that the for loop was a great tool to use in both cases. Firstly, because it meant that the stock reference entered could be compared with every stock reference stored in the file however I only had to write on piece of code this is good because it saves time and is much more efficient. Similarly, a for loop is used to delete the information stored for that quote by looping through all the positions in the file and moving their positions so the stock would be overwritten. This was a good tool because it meant I could easily control all quotes in the file as the for loop looped through the required quotes. | language is the ability to use global variables. Also seen above in the staff section, the use of global variables to store the stock information helped me to create the delete function as it saved memory space since I did not to create local variables inside the function to delete the stock information they were already declared saving both memory space and programming time | feature because if an item of stock became discontinued and the staff could no longer buy it then they would not need it in the stock file. So by having the option to delete it, it saves space within the file and also prevents them from entering it by mistake when they can no longer buy it for their bookings. | | pseudocode. This is a strength because it helped me to understand the delete function and know the step by step process before starting the programming. Another strength in my design was the design of the UI for the stock function. I believe this was a strength because it is very clear to the user how to access the delete stock feature as each menu option is clearly labelled in a way that would make sense to Stuart and how his company works. | the choice of global variable name for the number of stock stored in the file. This is a weakness because when changing the number of stock stored in the delete function the name of the variable, nsti, was very similar to that in staff, nsi, so I mixed them up when programming causing errors in the delete function. | other delete functions during development I made sure that the delete function worked correctly before I included the checks that it is the right item of stock. I think that this helped me because it meant I could focus one piece at a time rather than an overwhelming amount to do in one function. | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to change the price and quantity of the stock. | Allows the user to change the price and quantity of the stock which is then saved to the | The use of the tool 'strcmpi' allowed me to compare the reference entered by the user and | | As seen above in the staff section, a good feature of the completed solution is that if information | As can be seen above , when the user enters the new data they can't go back and edit their input or | A strength in my design similar to what can be seen above is the design of the user interface. | The UI is clear and easy to understand with suitable prompts to guide the user | | A weakness in development was the use of operators. This was a weakness because when I programmed the change function I used the == | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | permanent text file. | those stored in the file. I think that this is a good tool to use because if there is a match it must be definite as the comparison are coming from within the stock file. Furthermore, from the use of strcmpi I can output further information on the customer to allow the user to check it is the correct one because the strcmpi tool has already found the stock in the stock file so it was easy to output further information. | | changes in the future such as they the price of the stock without the change function, they would have to delete the stock and re-add it. However, with the change function it is much easier since the user can now just change the one field which is much more efficient. | check its correct before submission – it just happens. This is a short coming because it means that incorrect data might be being entered. | This is a strength because as the company aren't used to computer based systems the clear easy to understand interface is a great strength in my design. | through the function this is a strength because Stuart and his employees aren't used to a computer system so the easier it is to follow and understand the better for his company personally. | | operator to allow the change to happen however when programming I put a single = sign this caused the program to not work as the two values either side were not being compared properly. However, once I realised my mistake and added another = the change function worked correctly. | |
| Enable the user to search for stock using stock ID to display information. | Allow the user to search for stock producing a list including the colour, price, and quantity. | Similar to the staff and quote search an important tool I used was strcmpi. This allowed me to compare the stock reference entered with those in the fie and then output the correct information if the comparison from strcmpi was a match. Furthermore, the for loop allows the strcmpi to be run for the amount of stock in the file and compare it with the one entered to output the | | The search function is great feature because it allows an employee to find all the relevant information they may need on an item of stock. For example, if they needed a specific colour but didn't know the type of paint it was they could use the search function since without it there would be no way of viewing stock. | A shortcoming of the solution is that unlike other view functions in the system the only field that can be searched for is reference. I think that this is a shortcoming because it means that the users have less flexibility on how to find an item of stock stored in the system. | A strength in my design was the choice of outputs for the search. I believe that this was a strength because the outputs given ensure everything stored about an item of stock can be viewed so an employee will never be missing any information when they do a search. | | | During development I mixed us the different data types. This was a weakness because it caused the search function to not work properly. An error message was profuced even when the search worked because I declared a variable as char instead of int. However, once I changed this to the correct data type the view function worked correctly. In the future, I will be more careful when first declaring my local variables. | A suggestion for improvement would to also be to search for stock by other fields such as colour of paint. This would be anther great way to search because their may be multiple colours but different types and they may want to see which of either they currently have. |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | correct item of stock. I believe that it is a good tool to use for my function because without it, it would be more difficult to find the correct reference. | | | | | | | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Enable the user to sort stock quantities from low to high. | A list of stock with increasing quantities will be shown on screen displaying the stock id and the actual level of stock. | The use of the tool 'sscanf' allowed me to convert the quantities in the stock file to integers this was extremely useful because for the sort to work the quantities needed to be integers. Furthermore, the tool 'strcpy, was also very important as it allowed me to carry the out the bubble sort by being able to move the value of a variable between different local variables to arrange the sort correctly without this tool the actual bubble sort would not take place. | The use of the recursive algorithm was very effective in the sort of stock quantities as it allowed it move through all the items of stock and sort the quantities accordingly before ending the program and outputting the appropriate information | The sorting feature is a great feature to have in the system because it allows Stuart and his employees to keep organised. This is because they can see what stock is low and identify which they will need to go and buy more of. | | A strength in my design was the choice of outputs for the sorted stock. I think that this was a strength in my design because of just outputting the stock references which can be a bit vague it also outputs the colour and type of paint which means that the employee will have a better idea of what that item of stock is so that it is easier to order. | I think that a weakness in my design was the pseudocode beacyse it wasn't very clear on all the steps I would need for the sort in c++. This then caused the development of the sort to be more difficult as I did not have a detailed structure to follow. | A strength in my development was the choice to create another function to output further stock information. I believe that this was a strength in my development because it means that in the future if Stuart wants to change the outputs or add to it, it can be easily added in only one function. | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Produce a message when stock quantities drop to a specific level. | Allow the system to determine when stock quantity drops to a specific point and produce a message to alert the user. | An important tool I used was atoi. This was a great tool to use because it allowed me to convert the quantities stored in the stock file into integers for a range check. This meant that the low stock function could work correctly and compare the new integers with 2. Another useful tool I used was the operator <. This was a useful tool to use because it meant I could check that the quantities were below a certain value which then determined if it was low stock or not. The for loop was also a great technique to use because it means I could loop through and check the quantities for all the stock in the file. This was great because it meant that I had peace of mind that the system was checking all the stock in the file. | The use of local and global variables were very effective in this function. The global variable allowed me to find the quantities of each item of stock in the file and extract the value. This meant I could then store it in a temporary local variable which was a different data type. Furthermore, the use of a local variable meant I could save memory space because once the function is complete the memory space would become free. | I believe that this is a great feature of the system because it helps to keep the company organised. This is great for my company because during the investigation Stuart said he wanted ways to help his organisation so by having a tool that tracks low stock that may need re-ordering it hits this customers requirement. | | | A weakness in my design is that when I created the pseudocode for this function, I stated that the quantity must be 0 for it to be outputted to the search. I think that this is a weakness in my design because if the stock is already at 0 and is needed for a job it is too late, even though the purpose of this function is to keep the company organised. Therefore, in the system I chose to change the value. | As I had already completed many view functions for a serial file it helped to improve the speed and accuracy of my programming because I already knew which local variables I would need and how to use the for loop to find the values I needed. This was a strength in my development because I used the skills I have developed during development to help improve the speed and efficient of my programming in turn cutting down the time it took to develop the system. | | A suggestion for improvement would be to include a message if no stock is low since at the minute nothing appears, and it just closes the function. I believe that a simple output message using an else statement would help make the system less confusing because the company aren't used to a computer system, so if nothing happened when they opened the function they may become confused or worried it was broke. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | the programming language | | | | | | | avoid the problem |
| Ensure information entered is validated. | Validate the price using a format check, producing an error message if incorrect. | The tool shown above in previous sections were also a great use to me in the validation of stock inputs helping to improve accuracy. | The use of modular programming and parameter passing was very effective as describe above. | The validation of all inputs is a great feature to have within a system because it reduces the chance of something wrong or inaccurate being entered. This is a useful tool because if a user went on a search and one of the fields didn't make sense they would be stuck confused however, with the addition of validation there is little chance of confusion as certain things are required for certain inputs. | | A strength in my design was the validation table. This was a strength because it meant I had easily broken down how all inputs would be validated ensuring nothing was left out. | | A strength in my development is that as some of the validation routines had already been used I could quickly copy these functions up and change the error message to be more suited. This was a strength in my development because it meant I was saving time and already reducing the chance of any errors since the original function must've worked for me to have moved on. | A weakness in the development of the range check for the volume of paint is that I allowed a volume of 0L to be entered however this is not realistic and also a waste of time to be entered. This was a weakness in my development because it meant that the validation did not ensure the input was accurate since there is no such item of stock to be bought that is 0L. Therefore, I changed this inside the validation function to something more relaisitic. | |
| Overall, I am very happy with the stock function.<br>It includes many of the feature I set out for it too and allows many links to be made during the pricing of quotes.<br>The use of a serial file was a good choice as most customers will have similar paint choices i.e. magnolia so only a small set of stock will be stored in the file also seen as it's a small company they will not hold lots of stock at once anyway.<br>I believe that the stock section is very accessible for the new users and I implemented a clear UI which can easilt be followed to find the different features they may require. | | | | | | | | | | |
| | | | | | | | | | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to add a new booking to the schedule including: Quote reference and the dates of the job and stores this in a permanent text file. | A permanent text-based file that stores the following information in the schedule: customer information and the starting date of the job. Furthermore, the use of the while | As the schedule information would be stored in an order for the 1st of Jan to the 31st of December a sequential access file was a great file choice as it is also a large amount of data. | Global variables were key to the addition of a booking as the main inputs: staff, date and hour were all global variables. This means that the programming language was very effective in the | The schedule is a great feature to have as if there was no option to add a booking there would be no schedule so no employees would know when they were working which would be a major issues | | A strength in my design was the systems flow chart. This allowed me to see the appropriate links I needed to create when programming the inputs for the booking. For | | | During development of the addition of a booking I had issues with the booking being saved to the file. I deduced that it was the read back function and the length of the variables. After trialling different combinations I found some that worked correctly and allowed me | A suggested improvement would be to allow the user to add a booking across multiple days rather than just 1. This would be better because it would save time for the user as for now they have to |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | loop is a great tool to use because it means that the user does not have to enter the booking hour by hour instead the while loop allows the user to enter just the start and finish time and the while loops adds in the rest of the hours in between. | | sense that it allowed me to use global variables for this purpose. | causing a lot of aggravation for the company. | | example, it allowed to me to see there was a link between the booking and a customer so I knew I needed to include a customer reference for one of the inputs. This is a strength because it allowed my system to be more interlinked and flow together. Another strength in my design is the ERD. This is a strength in my design because it allowed me to understand the process of the addition better so that when I was developing it I understood the steps better so it was easier to program. | | | to see the booking once I closed the program an re-opened it. | enter bookings day by day. I would do this using a similar piece of code to that for the hours implementing a second while loop. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow only those with the highest access to change information such as date of the job. | Allows the user to change the date of the job which is then saved to the permanent text file. | The strcpy tool was a great tool to use on the schedule because it allowed me to change what the user sees for the cancellation. For example changing the quote reference outputted onto the schedule to | Similar to above the programming language was very useful in the fact it allowed me to use global and local variables. Using local variables allowed me to take in the new date of the booking from the user then further | A great feature of the change is that not only does it add in the new booking but deleted the old booking from the schedule. I believe that this is a great feature of my system because without it the user would have | | | A weakness in my design was the pseudocode. I think that this is a weakness because I did not include the deletion of the old booking which meant that during development I | | A weakness in my development was the choice of operator within the while loop. This was a weakness because it meant not all the old hours were being deleted as I set it > instead of >= so the final hour was being left on the schedule. Once I realised it was due to this logic error I changed the | Similar to the add function, I would allow multiple days at a time to be changed rather than one day at a time because it saves time for the user helping to make the system much more efficient for them. |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | an asterisk to show the time is now available. This was also a great tool to use the other way round when adding the new booking. | down convert this to the global variable for adding to the schedule however this could not have been done without the local variable as both the old and new variable could not be stored under the global variable date. | to go into two separate functions which would be a lot more hassle and more complicated for the user. | | | had to code it straight away rather that having a logical plan to follow like other aspects of the program this meant that it took me longer to code. | | operation and the function worked correctly. | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Enable only those with the highest access to delete information such as a cancelled booking. | Allows the user to delete a booking from the system and for all the information related to be deleted from the text file | See above why strcpy was a great tool to use within this function and why the while loop was a great tool. | | I believe that the delete function is a great feature of the system as some customers will need to cancel their bookings for various reasons. However, if there was no option to delete the booking an employee may forget but also they would see the schedule as busy so may dismiss other work which would then lost the company money. Therefore, I believe that this is a great feature of the system. | When a booking is deleted it is just deleted from the schedule however it would be more beneficial to the user if they could write a reason as to why it had been deleted for analytical reasons so they could see if the cancellation was anything they can work on for future bookings. | A strength in my design was the problems and sub-problems. I believe that this was a strength in my design because it allowed me to see all the aspects of the delete function I need to implement. For example, making sure access to this function was clear on the UI and also remembering to include validation on important inputs. | A weakness in my design is that I said to validate the quote reference entered. I think that this was a weakness in my design because it caused confusion for me when developing since the user does not need to enter the quote reference when deleting a booking. | A strength in my development is that because I had already included a delete in the change function so I already knew how to do it and knew it would work without errors. Therefore, I think that this was a strength because it saved me a lot of time when programming. | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Include validation on the new information which is entered including the date. | Validate the start date using a format check, producing an error message if incorrect. | A great tool I used in validation was sscanf. This was an important tool to use because it allowed me to convert the select characters of the date entered into integers. This was very important because it allowed me to convert the date into a number which could then be used on the schedule. Furthermore, similar to what can be seen above itoa and atoi were a great help in the range checks for the start and end times. Another great tool I used during the validation routine was the floor tool. This was important because it allowed me to check if the year entered was a leap year or not for the validation routine to work correctly. | A very effective tool I used was the use of global variables. They were very effective because it allowed me to store the start date entered as global variables, this meant that I could then use them in the range function to compare them with the final date entered. Without global variables I would have not been able to validate the end date in the sense of time. Furthermore, the use of parameter passing was extremely useful for validating the inputs this is be described in more detail above. | A great feature of the complete solution is the validation check which uses the systems clock. I believe that this is a great feature because it means that if the user tries to enter a date in the past it will not allow them, this helps to make the solution more accurate and reduce the chance of any errors since if the company turns up on the wrong day this could look very unprofessional. | | A strength in my design was the flowchart for the systems clock validation. I believe that this was a strength in my design because this was one of the harder validation routines I implemented so by having the flowchart to follow it really helped me create a clearer idea of what I needed to do to help reduce errors when running it. | | A strength in my development is that when I was implementing the validation of quote references to check they exist I had already done this for other serial files such as the staff reference. This was a strength because it meant that I could use the other functions that already worked as a template for this one helping to improve the efficient and correctness of my program. | A weakness in my development was the choice of data type when validating the date entered. This was a weakness in my development because I declared the variable year as an integer. This meant that when I did the division there was no difference between a normal year and a leap year since it was not returning a decimal because it was an integer. However, once I changed the data type to float the date was converted correctly meaning the validation routine worked. | |

| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
|---|---|---|---|---|---|---|---|---|---|---|
| Ensure there is hierarchical access with levels of access of the administrator and | Include levels of access with level 3 being dedicated to the administrator to backup data. | When adding the first employee to the file a great tool I used was strcpy. This tool | The use of global variables were very effective when implementing the | Hierarchical access is a great feature of the system because it implements | | | A weakness in my design is that the access is numbered 1-3. This could be | | A weakness in my development is that originally I created no way of using the hierarchical access | A suggested change for the hierarchical access would be to use words |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| the members of staff below. | Level 2 will be for staff members who have the authority to edit the schedule and then level 1 will be for the staff with limited access who can only view. | allowed me to copy the value of 3 into the level of access for the user automatically since the first person to use the system will be Stuart, the manager. This was a great because it meant I could cut down on the inputs from the user helping to reduce the chance of inaccurate inputs for important fields such as security level. | access. This is because when the user logins in I created a function to extract their level of access and store it as a global variable to be used elsewhere in the program such as checking the access level when attempting to enter the delete function. | security throughout. Security is very important when handling private information such as employee details which should only be accessed by those with managerial positions therefore hierarchical access solves this issue. Furthermore, it ensures new employees won't be able to edit the schedule and cause issues since the manager (Stuart) will decide their position in the company when their information is added to the system. | | | considered a weakness in my design because the user may get confused with which number is the highest level and which is the lowest. Therefore, the manager may accidentally give a new employee the highest level of access instead of the lowest. | | through the system. This then because a great weakness in my development because it meant that there was no security through different menus because there was no level of access to check. However, once I created the function to store the access level as a global variable it all worked correctly and the security was implemented throughout the system. | rather than numbers. For example, admin, supervisor, and member. I believe that they would be better because they are more descriptive than numbers so it could make it much easier for the user to remember. I would implement this in a similar way to the type of paint validation and use strcmpi statements to check that the user has entered one of the 3 options and nothing else. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to view the schedule | Output the following information to the screen: Quote reference along with the dates and hours. | A technique I used was a for loop. This was a great tool to use because it allowed me to output the number of columns corresponding to the hours worked so that the user can easily see what hours they are working. | | A great feature of the completed solution is the todays work function. I think that this is a great feature of my system because it allows the user to see where they will be for that day, but it also gives more detail by linking the quote information. This | | A strength in my design was the creation of the representation of the multi-dimensional array. I think that this was a strength in my design because it helped me to visualise what I needed to do for the view function and helped to | | A strength in my development is that when I programmed the view by staff function I only programmed the staff member 1 dirst and checked that this all worked correctly because one I knew that this was all working all I had to do for the second | A weakness in my development is that at the start I struggled with the understanding of what to pass to the function to find the quote information for the todays work function. Once I understood that I needed to pass the whole of the booking[date][hour][staff] as this is was where the quote reference was stored it made much more sense and the view | A suggested improvement for my view function is to allow the user to also view by month. I believe that this would be a great addition to my system because it would allow the user to see in advance but not to the extreme of the whole year which can be |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | I also used sscanf, strcpy and floor which were great tools these can be seen described above in the validation section. | | is great because it means they will have more information of what they will be doing and also where they will be for that day rather than just a number on the schedule. | | make it easier to code for me. | | member of staff was copy the function and just change the value of the variable staff. This was a strength in my development because it meant I had a lot less errors as I only was only really doing the programming once. | function all worked correctly. | overwhelming to see. |
| Overall I am extremely happy with the schedule function.<br>I believe that it meets all the points required by the objectives and includes all the points Stuart asked for during his interview.<br>I am happy with the layout of the schedule menus because I think that they are easy to understand and use helping both Stuart and his employees quickly get to grip with the system. | | | | | | | | | | |
| | | | | | | | | | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to add a new invoice to the system storing this in a permanent text file including the following information: Customer information, Date of the job, Description of the job and a new price. | A permanent text-based file that stores the Customer information, Date of the job, Description of the job and a new price. | The tool I used strcpy was a great tool to use as it allowed a buffer to be entered this can be seen described above. | Similar to the other add functions the modular programming was very effective as it allowed me to use the function to read and write to the invoice file. This can be seen described above. | The addition of an invoice was a great function because it allows a staff member to wrap up a job and store a complete record of the job in the system once it had been complete. This is good because if an employee needs to go back to it for any reason it would be there saved to the file. | | A strength in my design was the choice of inputs. The input of payment status was a great choice because it mans that the users can track what customers have paid for their bookings and who still need to pay. | | A strength in my development is that I chose to link most of the information from the quote rather than asking the user to re-enter the information. This was a strength because it meant that less memory space was being used and also saved time for the staff as they would have less information to enter. | | A suggested change is that when an invoice is saved to the file there is no way for the customer to see the invoice unless a staff member shows them. Therefore an improvement would be to allow the customer to see the invoices added to file so they have a copy of the booking also. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Allow the user to be able to change information including date of the job and whether the job has been paid for. | Allows the user to change the date of the job and to be able to edit when the invoice has been paid which is then saved to the permanent text file. | Similar to the other change functions seen above some great tools to use were strcmpi and the for loops which can be described above. | Parameter passing was a great aspect of the language as it allowed me to do the checks before the information was changed. Parameter passing meant that I could pass the customer reference to a function to find a customer and output their information this was very useful because it allowed me to help reduce the chance of anything being entered incorrectly. | A great feature of the change function is allowing the employee to change certain inputs. This is good because most of the information is taken from the quote however if things change the user would not need to create a brand new quote they can just use the change function to edit that required input. This is good because it helps speed up the process for the staff member. | | | A weakness in my design is the choice of inputs. I believe that this is a weakness because if the dates change it is likely that the price of the job may also need updating. Therefore, this is a weakness because in this case the user would then have to add a new quote. | A strength in my development can be seen in the validation of the booking. Since the change functions is the same for all serial file I used one from above in the program that already worked as a template to help reduce any chance of errors but also to speed up the time taken. | A weakness in the development is that I changed a variable incorrectly on an output. This meant that no information was outputted for the checks as there was no data stored. Once I realised I needed to use the variable ‘find’, as this is what actually stored the reference, the correct information was outputted. | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Allow the user to delete an invoice from the system once it is no longer needed and the job has been paid. | Allows the user to delete an invoice from the system and for all the information related to be deleted from the text file. | A great tool I used similar to the other delete functions is the for loop as it ensure all invoices move position in the file. | As can be seen above an aspect of the programming language that was really effective was the use of global variables to delete things from the invoice file. | A good feature of the completed solution is that is allows an employee to delete invoices that are paid for and complete. This is a good feature because it means the company don’t have to risk holding any information they don’t need to but it also helps to save memory space as new invoices can be added instead. | A shortcoming of the compleed solution is that the only way to delete an invoice is by using the reference however it may be more beneficial to the employees of they could do more things by entering the last name of a customer instead as they are much more likely to remember this compared to a number. | Another strength in my design was the UI. I believe that this was a strength because similar to the other sections the menus follow a similar pattern this is a strength because Stuart and his company are not used to a computerised system so by keeping the system as easy to follow and remember, it helps to make it | | Similar to the above strengths by the time I implemented the invoice functions I had already dealt with the delete function for multiple serial files so it was much quicker and simpler for me to do compared to the first time on the staff function as all the errors I had already dealt with previously. | | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | much easier for them to use. | | | | |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Enable the user to be able to search for an invoice by reference | Allow the user to search for an invoice producing a list including the customer's name, price, and dates of the job. | Strcmpi was a great tool to use in all of the view functions because it allowed me to check that whatever the user wanted to see matched with that in the file before outputting. This is important because it meant that I could ensure the right information was outputted for the user from the file and not another references details for example. | The use of modular programming allowed me to use the read back function I created to read to the invoice file to retrieve all the information I would need to output within the search functions. | | Similar to what is seen above I think that a short coming of the solution is that viewing by a customer's last name is not an option as it would be a great extra feature for the user. For example, if they were on the phone with a customer and they needed to access an old booking the best way to find it would be to search by the customers last name. | A strength in my design was the choice of outputs. I believe that this was a strength because not only did it allow me to output booking information but also the customer information so if there was anything a staff member needs to check to validate the details of the booking they don't have to look anywhere else as the customer information is also outputted due to my choice of outputs. | | Similar to above, as I had already implemented many view functions for a serial file I could use one of these as a template for this. It was a strength because it was a great way at helping to reduce the chance of any errors since it had already been briefly tested above. | | A suggested improvement to the view function would be to include pound signs when viewing the breakdown of the price. I think that this would be better as it would me more consistent for the program as pound signs are used on view quote and it also makes the system look more put together. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Enable the user to be able to search for paid or unpaid invoices. | Allow the user to search for an invoice producing a list including the customer information and whether the invoice has been paid. | The same tools were used as what can be seen above. | Similar to above the use of modular programming and the read back file allowed me to easily find the unpaid invoices stored in the file. | A great feature of the view is allowing an employee to see the unpaid invoices. This is good because it allows the staff to keep track of what customers still need to pay | | Similar to above, I believe that a strength in my design was the outputs because I chose to include customer information. This is a strength because it means that when the | | | A weakness in my development was in my strcmpi statement for unpaid invoices I was not using the speech marks correctly to compare the value of the variable unpaid invoice and at the time 1. This was a weakness because it meant that the view | A suggestion for improvement would be to introduce a time frame to the unpaid invoices so that the oldest invoices are shown first, like a sort, to help emphasise these |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | for their job as it also links the customer information, helping to improve the organisation skills of the company. | | unpaid invoices are outputted customer names and mobile numbers are also outputted giving the employee a chance to chase up why the job had not yet been paid. Further improving the organisation of the company. | | | unpaid invoice did not work correctly and no invoices were shown. However once I corrected this error it all worked correctly and I could see the unpaid invoices. | invoices to the employees so that they get dealt with. |
| Objective | Success Criteria | Justification of the tools and techniques | Evaluation of the effectiveness of the programming language | Good features of the completed solution | Short comings of the completed solution | My strengths in the design | My weakness in the design | My strengths in the development | My weakness in the development | Suggested change of approach to avoid the problem |
| Ensure there is validation in place for all information entered excluding the description of the job. | Validate the date and price using a format check, producing an error message if incorrect. | Similar to the validation in the schedule section the use of sscanf to extract the dates for checking they are logically correct and in the correct format. | The use of global variables to store the start date similar to the validation on booking was really important which can be seen described above. | Validation was an important feature of the invoice function as it allowed all data entered by the user to be as accurate as possible, helping to reduce the chance of any wrong inputs or inputs that may become confusing overtime. | | My strength in the design was the flowcharts. These were a great help when implementing the validation routines as it allowed me to break them down into smaller more manageable step-by-step processes which then formed the whole routine. | | Similar to other aspects of the invoice section, I had already created many of the validation routines so I could use these as a template for the invoice file. This was a strength as it allowed me to complete the system quicker but also reduce the chance of any errors since most of them would've been spotted previously. | | A suggested improvement to the system would be that when you enter a quote the number of days cannot be greater than 21 however when you enter the dates for the invoice there is no limit on the time gap between them. Therefore, to help improve the consistency of the program I would include validation on the start and end dates entered to check that they were at the most 21 days apart. |
| | | | | | | | | | | |
| | | | | | | | | | | |
| Ensure all information saved is backed up for all files | In the user interface provide a menu option to back up data | | | A good feature of my system is that it easy to back up and recover the data as the menu | When backing up the data there is no option to only back up one file its all of them or | | | During development I commented out the backup and recovery | | A change I would make is that I would allow the user to have the option to only |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | stored in the text files. | | | options are very clear on the UI. Furthermore, it is easy for the user to see if the backup has worked because it outputs a message for each file to say it has been copied. | not at all. This is not as good because the user may only need to back up one file which then wastes space on their external storage device. | | | functions so I could concentrate on getting the whole of the system working first. I think that this was a good idea because it's better to make sure all the file handling is correct and all the files work correctly before trying to back them up or recover them. | | back up specific files rather than all. Furthermore, I would produce a message to output to the user if the files were recovered since there is no way inside of the program for the user to know. |

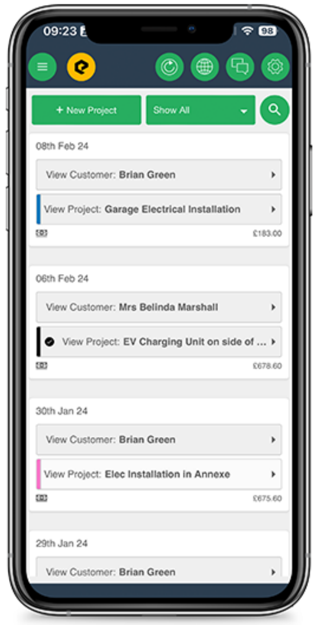
## Comparison of commercially available systems

## Comparison of the UI

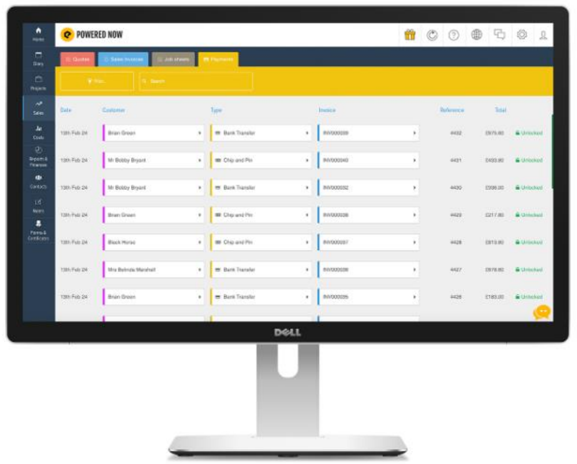
For my system I chose to use a menu driven interface however the Powered Now system uses a GUI. I think that a menu driven interface is better for Stuart's company because it is more simplistic. This is more useful because the company have come from completely paper based so a basic interface that relies on simply entering a number is much easier than trying to figure your way through the GUI of Powered now. Furthermore, a menu driven interface will have much quicker processing times and will not need software updates like the Powered now may need. I think that this makes my user interface better because the company does not need a complicated UI which may need updating because it is only a small company so the larger scale of powered now is unnecessary.

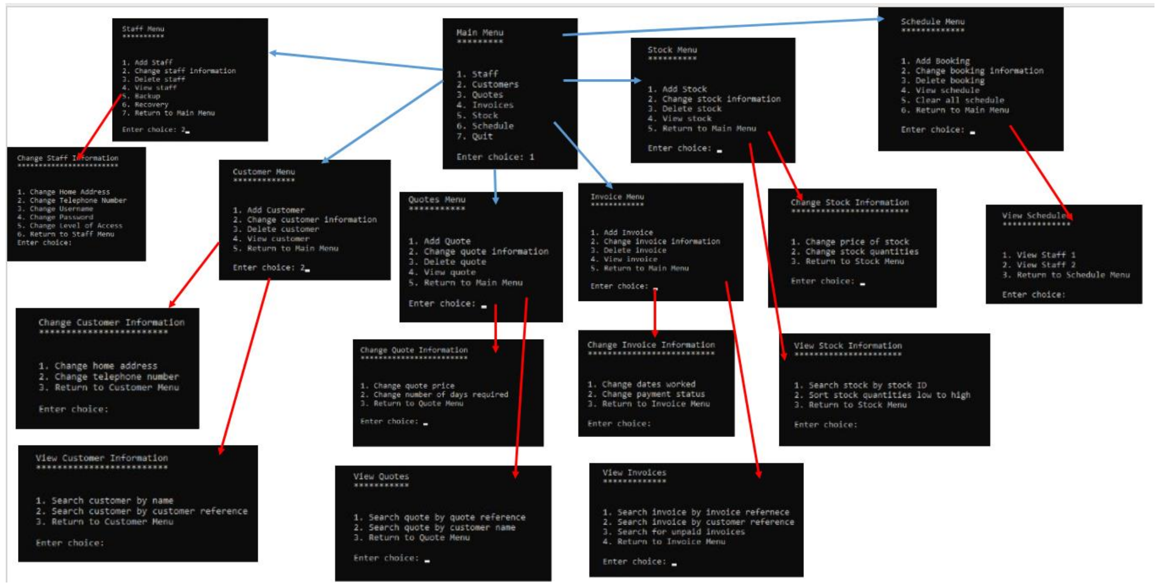
The powered now UI is a more vibrant colourful interface which is more aesthetically pleasing for the user which could have been nice to implement in my system. However, the employees will not spend long using the software as most of their working time is spent in customers home so therefore the colourful appeal of the powered now UI is not necessary for my system. Also, it may become more confusing for the employees since they aren't used to any form of computer system so it could overwhelm them so I think that my UI is more suited.

Furthermore, the powered now UI is a generic piece of software that can be downloaded by any company so the UI may not cover everything Stuart wants it to cover or it may do things that he does not need causing wasted processing time and un-used code. This shows me that my UI is a better fit since I interviewed Stuart so I could determine what he did and didn't need for his system therefore my UI is better for the company as it is unique to him unlike the powered now software.


09:23
+ New Project
Show All
08th Feb 24
View Customer: Brian Green
View Project: Garage Electrical Installation
£183.00
06th Feb 24
View Customer: Mrs Belinda Marshall
View Project: EV Charging Unit on side of ...
£678.60
30th Jan 24
View Customer: Brian Green
View Project: Elec Installation in Annexe
£675.60
29th Jan 24
View Customer: Brian Green

https://www.powerednow.com/





```
Main Menu
*********

1. Staff
2. Customers
3. Quotes
4. Invoices
5. Stock
6. Schedule
7. Quit

Enter choice: 1
```

```
Staff Menu
**********

1. Add Staff
2. Change staff information
3. Delete staff
4. View staff
5. Backup
6. Recovery
7. Return to Main Menu

Enter choice: 2
```

```
Change Staff Information
************************

1. Change Home Address
2. Change Telephone Number
3. Change Username
4. Change Password
5. Change Level of Access
6. Return to Staff Menu
Enter choice:
```

```
Customer Menu
*************

1. Add Customer
2. Change customer information
3. Delete customer
4. View customer
5. Return to Main Menu

Enter choice: 2
```

```
Change Customer Information
***************************

1. Change home address
2. Change telephone number
3. Return to Customer Menu

Enter choice:
```

```
View Customer Information
*************************

1. Search customer by name
2. Search customer by customer reference
3. Return to Customer Menu

Enter choice:
```

```
Quotes Menu
***********

1. Add Quote
2. Change quote information
3. Delete quote
4. View quote
5. Return to Main Menu

Enter choice:
```

```
Change Quote Information
************************

1. Change quote price
2. Change number of days required
3. Return to Quote Menu

Enter choice:
```

```
View Quotes
***********

1. Search quote by quote reference
2. Search quote by customer name
3. Return to Quote Menu

Enter choice:
```

```
Invoice Menu
************

1. Add Invoice
2. Change invoice information
3. Delete invoice
4. View invoice
5. Return to Main Menu

Enter choice:
```

```
Change Invoice Information
**************************

1. Change dates worked
2. Change payment status
3. Return to Invoice Menu

Enter choice:
```

```
View Invoices
*************

1. Search invoice by invoice refernece
2. Search invoice by customer reference
3. Search for unpaid invoices
4. Return to Invoice Menu

Enter choice:
```

```
Stock Menu
**********

1. Add Stock
2. Change stock information
3. Delete stock
4. View stock
5. Return to Main Menu

Enter choice:
```

```
Change Stock Information
************************

1. Change price of stock
2. Change stock quantities
3. Return to Stock Menu

Enter choice:
```

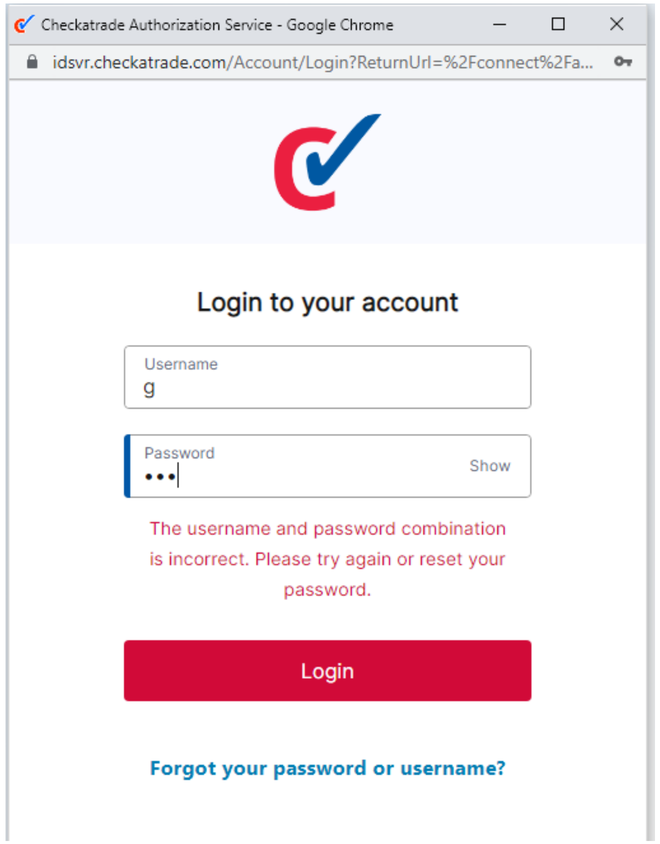
```
View Stock Information
**********************

1. Search stock by stock ID
2. Sort stock quantities low to high
3. Return to Stock Menu

Enter choice:
```

```
Schedule Menu
*************

1. Add Booking
2. Change booking information
3. Delete booking
4. View schedule
5. Clear all schedule
6. Return to Main Menu

Enter choice:
```

```
View Schedule
*************

1. View Staff 1
2. View Staff 2
3. Return to Schedule Menu

Enter choice:
```

# Login - Security

When logging on to the check a trade system the user enters both the username and password at the same time whereas my system validates them both separately. I believe that my systems approach for logging on is better because it suits the need of the company I am designing it for better. This is because as they are not used to having a username and password so they are more likely to forget it. So by having the individual error messages it helps them to know if their username or password is correct as if their username is wrong it won't allow them to move on. This is also more useful because if they are on the phone with a customer and need to log on quickly they can easily see what is wrong to allow them to try something different and log on as efficiently as possible


Checkatrade Authorization Service - Google Chrome

idsvr.checkatrade.com/Account/Login?ReturnUrl=%2Fconnect%2Fa...

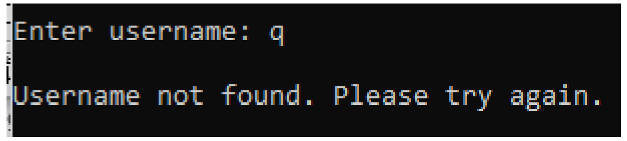
Login to your account

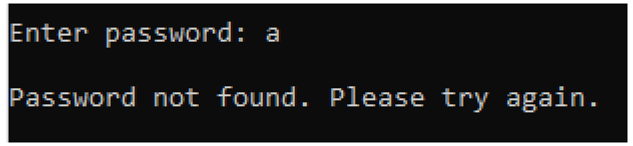
Username
g

Password
•••

The username and password combination is incorrect. Please try again or reset your password.

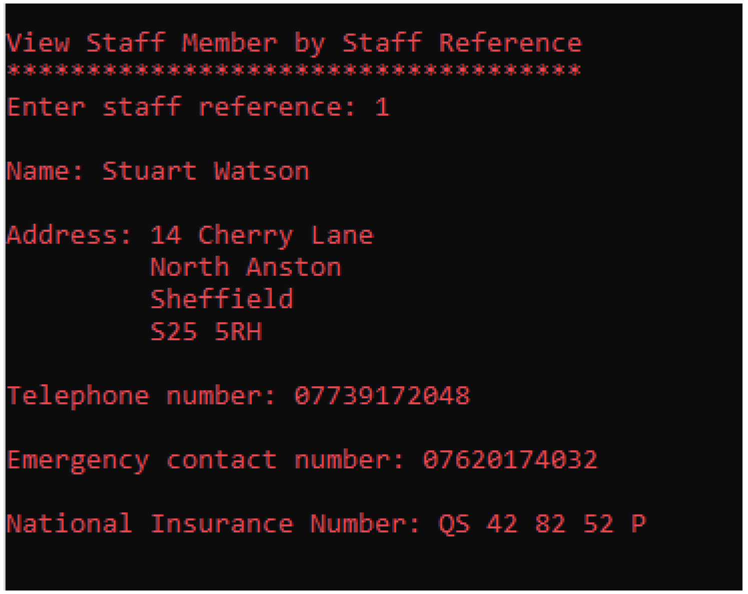
https://membersapp.checkatrade.com/login


```
Enter username: q

Username not found. Please try again.
```

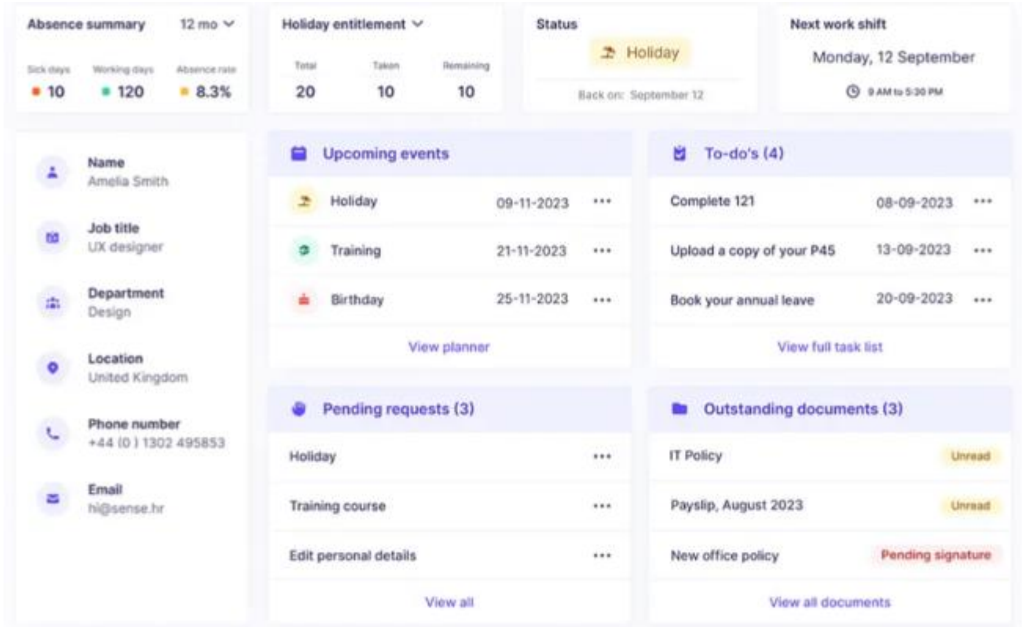

```
Enter password: a

Password not found. Please try again.
```

## View staff

A great feature of the Sense software for managing staff is that when you view an employee you can see extra information such as when they are not working whereas my system only shows contact details. I think that the Sense software is better because it allows the manger, Stuart, to see more detailed descriptions of an employee and allow him to keep up-to-date and organised about what his employees are doing. For example, if he needed to see a staff member wit a feature similar to that on the sense software he would be able to quickly see when he will be next working. However, with my system he would have to go on the schedule and find the dates. Therefore, I believe that the sense software is better because it is much quicker and efficient and would be a great feature to implement to my system.



```
View Staff Member by Staff Reference
************************************
Enter staff reference: 1

Name: Stuart Watson

Address: 14 Cherry Lane
         North Anston
         Sheffield
         S25 5RH

Telephone number: 07739172048

Emergency contact number: 07620174032

National Insurance Number: QS 42 82 52 P
```



Absence summary 12 mo
Sick days ● 10 | Working days ● 120 | Absence rate ● 8.3%

Holiday entitlement
Total 20 | Taken 10 | Remaining 10

Status
Holiday
Back on: September 12

Next work shift
Monday, 12 September
9 AM to 5:30 PM

Name
Amelia Smith

Job title
UX designer

Department
Design

Location
United Kingdom

Phone number
+44 (0 ) 1302 495853

Email
hi@sense.hr

Upcoming events
Holiday 09-11-2023
Training 21-11-2023
Birthday 25-11-2023
View planner

To-do's (4)
Complete 121 08-09-2023
Upload a copy of your P45 13-09-2023
Book your annual leave 20-09-2023
View full task list

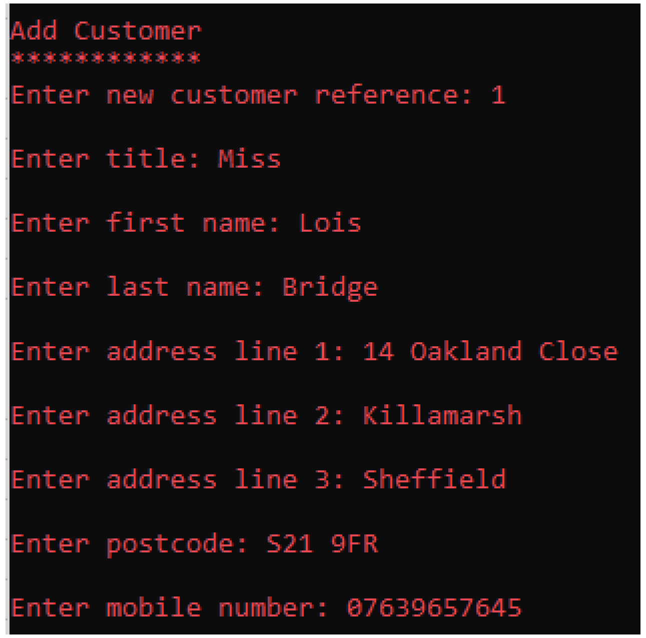
Pending requests (3)
Holiday
Training course
Edit personal details
View all

Outstanding documents (3)
IT Policy Unread
Payslip, August 2023 Unread
New office policy Pending signature
View all documents

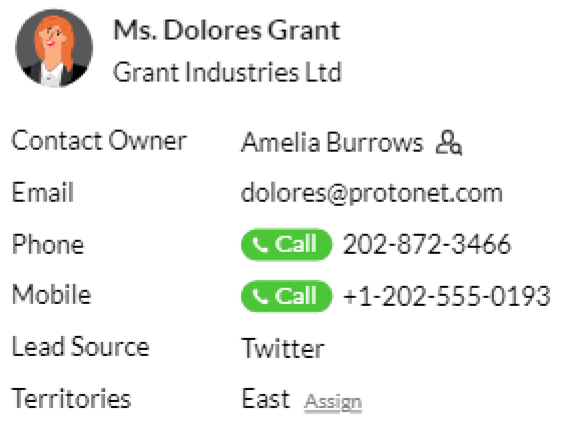
https://sense.hr/

# Adding a customer

When adding a customer to their system a choice of input for Zoho is the customers email address however, my system does not offer this. I think that my system is better because it is more tailored to my system since the employees are new to a computerised system they would most likely not use the customers email addresses as a point of contact as they may not know how to do it. Stuart's company prefer to contact customers over the phone as I found out when observing the system therefore I believe that  the choice of inputs on my system is much better than that of Zoho because they are much more likely to actually use the ones I chose compared to those on a generic system.



```
Add Customer
************
Enter new customer reference: 1

Enter title: Miss

Enter first name: Lois

Enter last name: Bridge

Enter address line 1: 14 Oakland Close

Enter address line 2: Killamarsh

Enter address line 3: Sheffield

Enter postcode: S21 9FR

Enter mobile number: 07639657645
```



Ms. Dolores Grant
Grant Industries Ltd

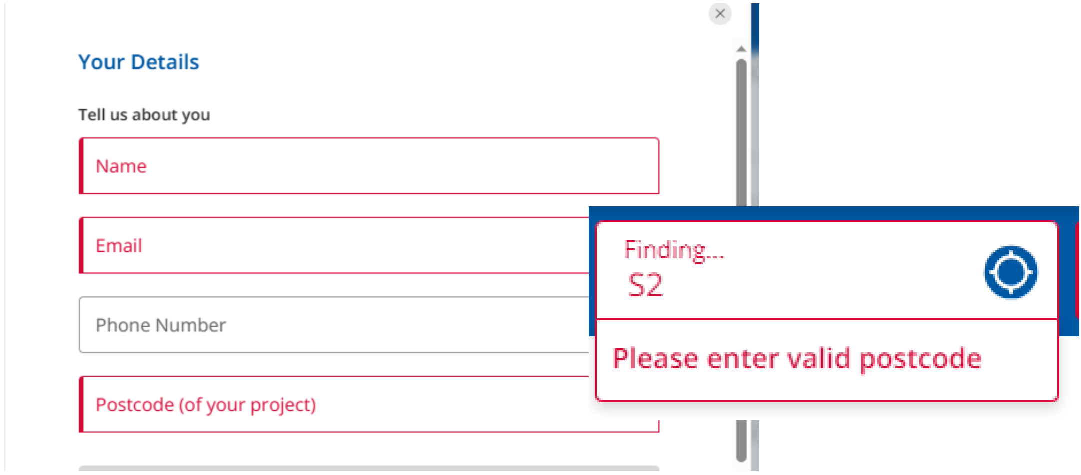
| | |
|---|---|
| Contact Owner | Amelia Burrows |
| Email | dolores@protonet.com |
| Phone | Call 202-872-3466 |
| Mobile | Call +1-202-555-0193 |
| Lead Source | Twitter |
| Territories | East Assign |

https://www.zoho.com/crm/

# Customer validation

When entering customer information for adding to the system both check a trade and my system ensure all information is correctly inputted and doesn't allow the user to carry on saving the customer to the system if it believes the data entered by the user is incorrect. I think that this is a positive of both systems because it ensures when the customer is saved to the system all the saved data is as accurate and in as much detail as possible. This is important because when a job is booked the staff will need to know their home address since if they don't know that, then the job cant happen and also contact details in case something goes wrong and something about the booking needs amending.

Furthermore, when comparing the addition of an address on my system to that for Wickes I can see one similarity between the two which I think is very important. When you enter the address, address line 2 is not mandatory on either. I think that this a great feature of my system because if Stuart or his employees take a job in the city centre they would not be able to enter the address properly if it was a required field. Therefore, by removing the validation on this it allows the company to take on jobs within the city centre and also store their data accurately as well the same as the more local villages.


Your Details

Tell us about you

Name

Email

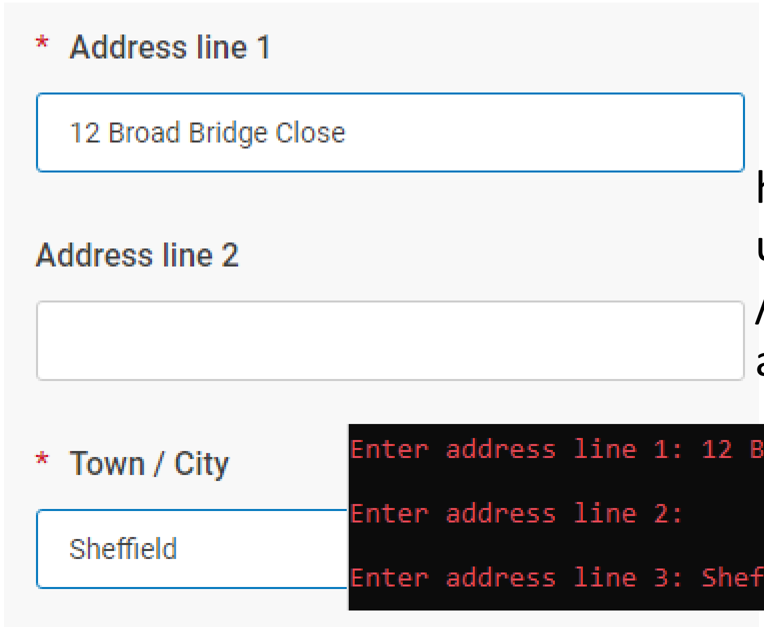
Phone Number

Postcode (of your project)

Finding...
S2

Please enter valid postcode

[Checkatrade: Book A Guaranteed Tradesperson](#)


* Address line 1

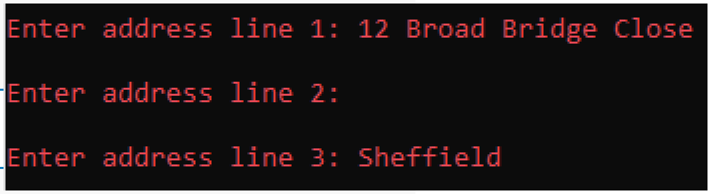
12 Broad Bridge Close

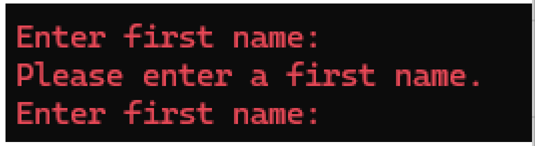
Address line 2

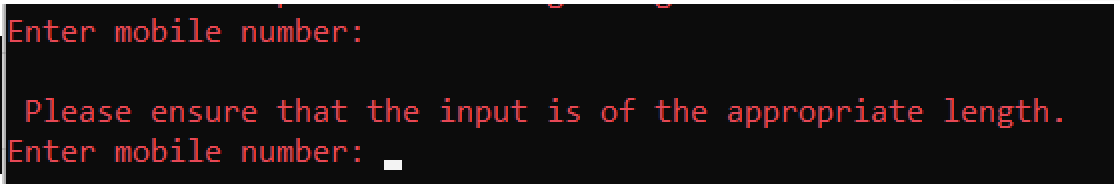
* Town / City

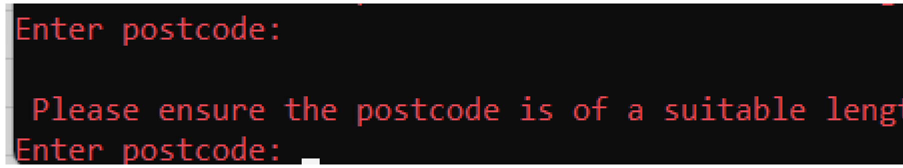
Sheffield

https://checkout.wickes.co.uk/#/delivery-addresses


```
Enter address line 1: 12 Broad Bridge Close
Enter address line 2:
Enter address line 3: Sheffield
```

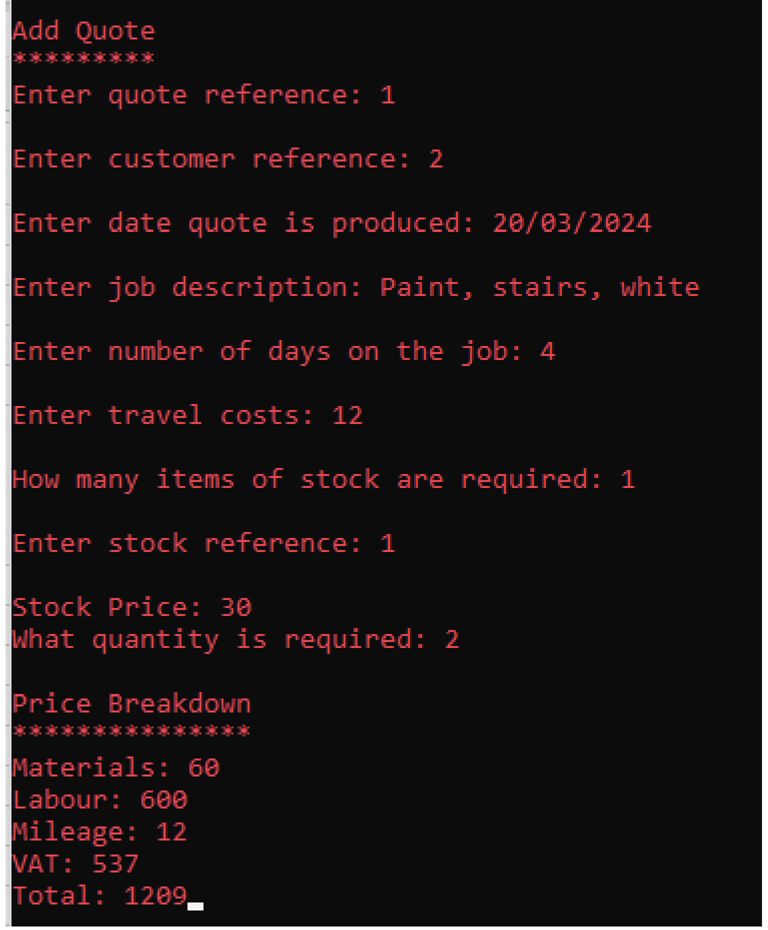

```
Enter first name:
Please enter a first name.
Enter first name:
```


```
Enter mobile number:

 Please ensure that the input is of the appropriate length.
Enter mobile number:
```

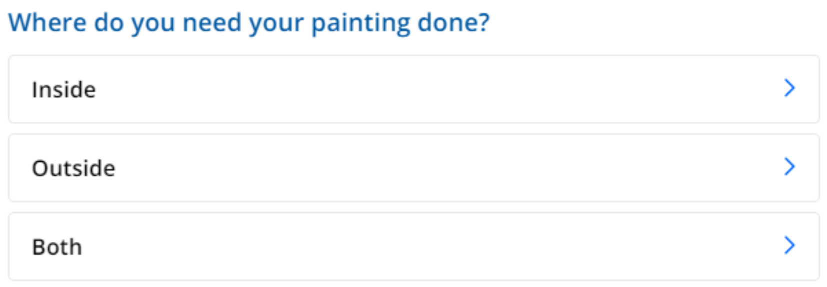

```
Enter postcode:

 Please ensure the postcode is of a suitable leng
Enter postcode:
```

# Creating a quote

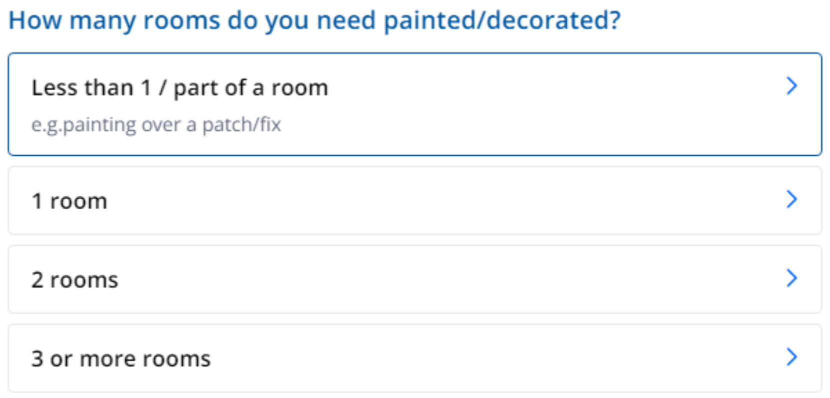
For adding information about a quote check a trade use forms to enter the information however my system allows the user to input text. For example, the job description input is open text in my system I think that this is better than the check a trade way because it allows the user to enter as much or as little information about the quote as they like unlike the check a trade which means only a certain amount of information can be entered and there is no way to add more if the user wanted too.
Furthermore, my system allows the user to enter stock costs which would be specific to that quote this means that the estimated cost would be more unique to the customer. I think that this is good because it means there is less chance of the price changing since so much of the aspects of the quote are already include unlike the check a trade system which takes little into account.



```
Add Quote
*********
Enter quote reference: 1

Enter customer reference: 2

Enter date quote is produced: 20/03/2024

Enter job description: Paint, stairs, white

Enter number of days on the job: 4

Enter travel costs: 12

How many items of stock are required: 1

Enter stock reference: 1

Stock Price: 30
What quantity is required: 2

Price Breakdown
***************
Materials: 60
Labour: 600
Mileage: 12
VAT: 537
Total: 1209
```



Where do you need your painting done?



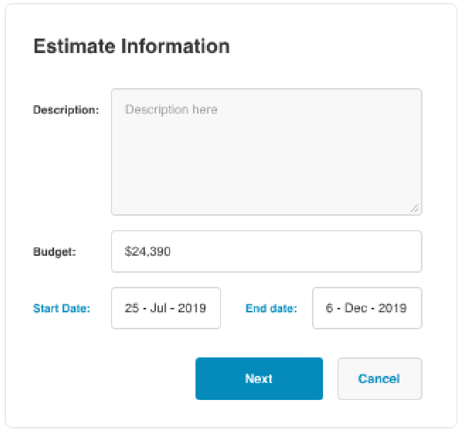
How many rooms do you need painted/decorated?

https://www.checkatrade.com/

# Creating a quote

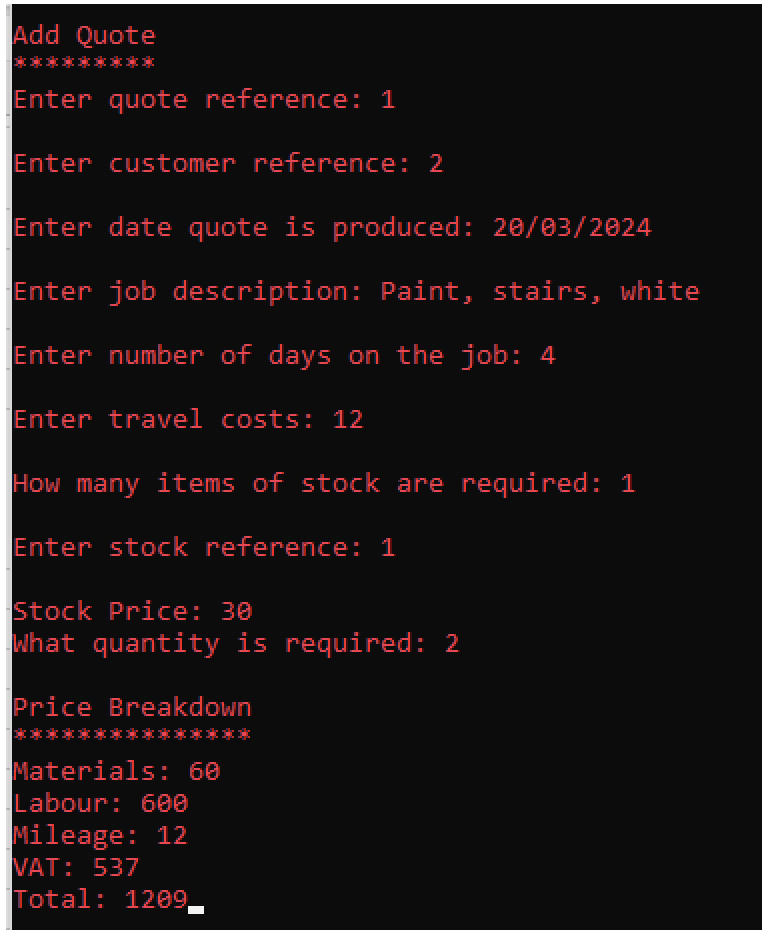
Furthermore, other available quoting systems available such as Work flow max make the user enter the total price of the quote unlike my system which does this for the user. I think that allowing the computer to do the calculation is better because it leaves less room for error since a human is more likely to miscalculate something therefore I think that the pricing of the quote is more reliable on my system compared to that on work flow max.

https://www.workflowmax.com/



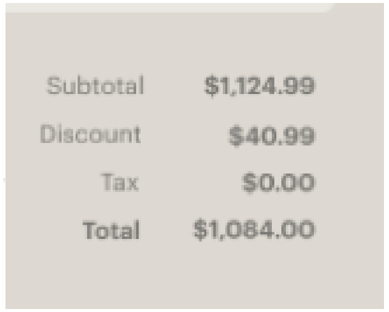
Estimate Information

| | |
|---|---|
| Description: | Description here |
| Budget: | $24,390 |
| Start Date: | 25 - Jul - 2019 |
| End date: | 6 - Dec - 2019 |



```
Add Quote
*********
Enter quote reference: 1

Enter customer reference: 2

Enter date quote is produced: 20/03/2024

Enter job description: Paint, stairs, white

Enter number of days on the job: 4

Enter travel costs: 12

How many items of stock are required: 1

Enter stock reference: 1

Stock Price: 30
What quantity is required: 2

Price Breakdown
***************
Materials: 60
Labour: 600
Mileage: 12
VAT: 537
Total: 1209
```

The panda doc quoting software uses a similar feature to my system by outputting the price of the quote broken down. However, I think that my system is better because it goes into more detail about where each aspect of the quote is so when talking to a customer or showing a customer the quote they can confidently understand where the price of the quote has come from unlike the panda doc software which only shows the tax added on.
A similarity between the two pieces of software is that we both included the VAT added on which I think is an important feature of both systems.

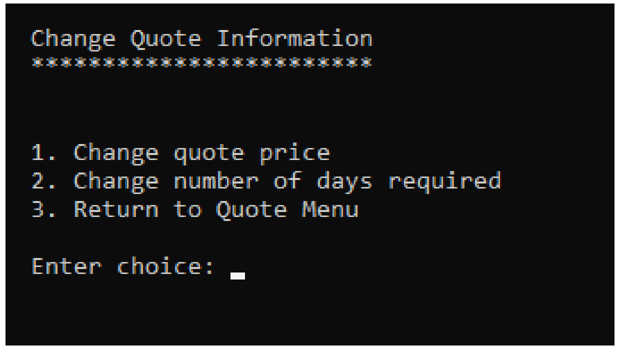


| | |
|---|---|
| Subtotal | $1,124.99 |
| Discount | $40.99 |
| Tax | $0.00 |
| Total | $1,084.00 |

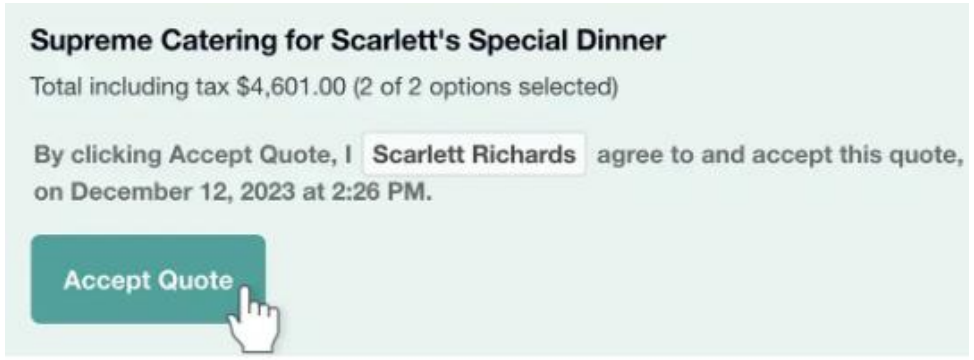
https://www.pandadoc.com

# Changing a quote

A great feature of the quotient software is that it allows a user to accept or reject the quote once the user has seen it. However, the only changes that can be made to a quote on my system is the price and number of days. I think that the quotient option to update if a customer accepts the quote is important because it helps a user to see what quotes will need to go further and be added to schedule and which quotes can either be left on the system or deleted. This would be useful because if a staff member looks back on a quote from a month ago they may not remember if a customer asked to be booked in or not, so by having the option to accept or reject the quote and the option to change this status would be really helpful for the staff as it would also improve their organisation.



```
Change Quote Information
************************


1. Change quote price
2. Change number of days required
3. Return to Quote Menu

Enter choice: _
```



**Supreme Catering for Scarlett's Special Dinner**

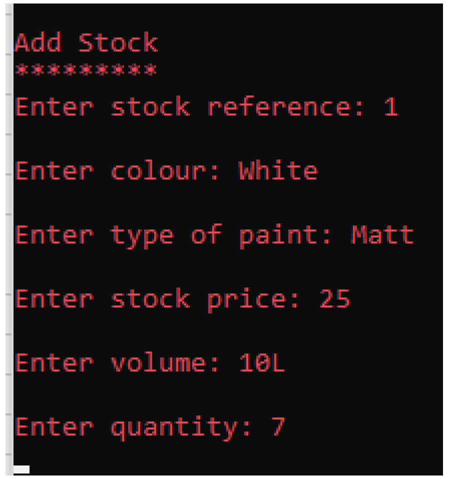
Total including tax $4,601.00 (2 of 2 options selected)

By clicking Accept Quote, I Scarlett Richards agree to and accept this quote, on December 12, 2023 at 2:26 PM.

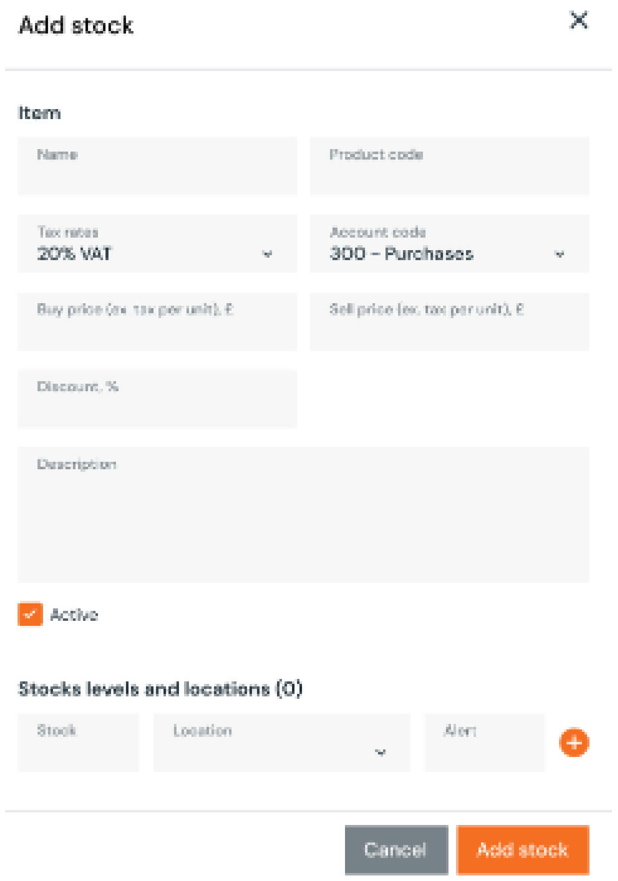
https://www.quotientapp.com/

## Adding Stock

When adding stock on the workever software there is little guidance on what information to enter there is just a box that says description however on my system there are clear inputs. I believe that my system is better because it is much clearer and easier for a staff member to use because there is much more guidance and structure on what to enter and where to enter it. I think that this is better because the employees wont be used to the system so the more help the better the system will be for them. Therefore, I think that the add stock feature in my system is better.



```
Add Stock
*********
Enter stock reference: 1

Enter colour: White

Enter type of paint: Matt

Enter stock price: 25

Enter volume: 10L

Enter quantity: 7
```



Add stock

Item

Name | Product code

Tax rates: 20% VAT | Account code: 300 – Purchases

Buy price (ex. tax per unit), £ | Sell price (ex. tax per unit), £

Discount, %

Description

[✓] Active

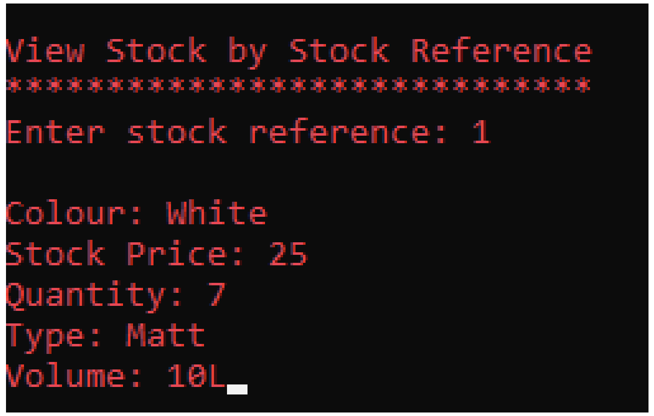
Stocks levels and locations (0)

Stock | Location | Alert

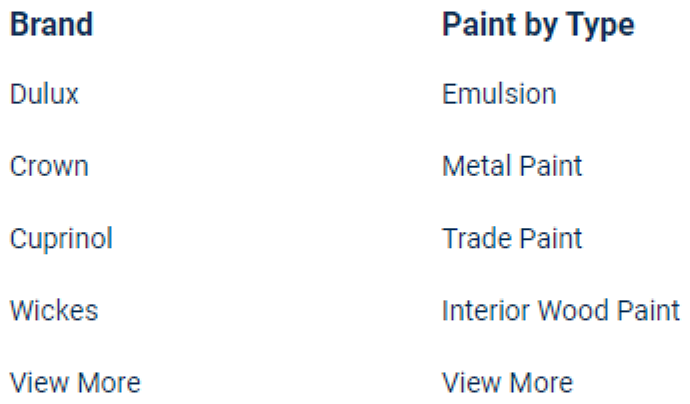
https://workever.com/us/features/stock-management/

# View stock

When viewing stock on my system the only way to find stock is to search by the reference however the Wickes system allows a user to view stock by different fields such as the type of paint. I believe that the Wickes system is better because it gives the employees much more variation on what they can search for since they may want to know what matt paints they have however, my system only lets them search for a reference therefore I think that the Wickes system has a better view feature that my system and this would've been great to include in my system.



```
View Stock by Stock Reference
*****************************
Enter stock reference: 1

Colour: White
Stock Price: 25
Quantity: 7
Type: Matt
Volume: 10L
```



| Brand | Paint by Type |
|---|---|
| Dulux | Emulsion |
| Crown | Metal Paint |
| Cuprinol | Trade Paint |
| Wickes | Interior Wood Paint |
| View More | View More |

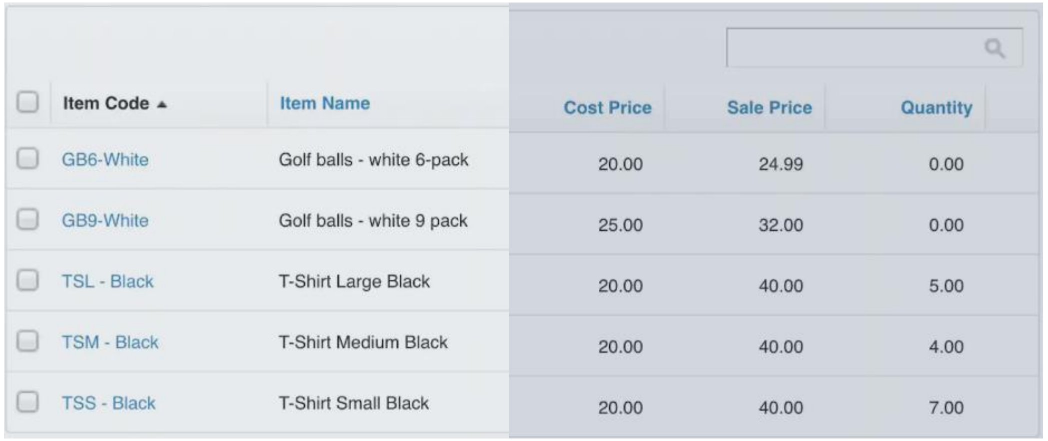
https://www.wickes.co.uk/

# Stock - Sort

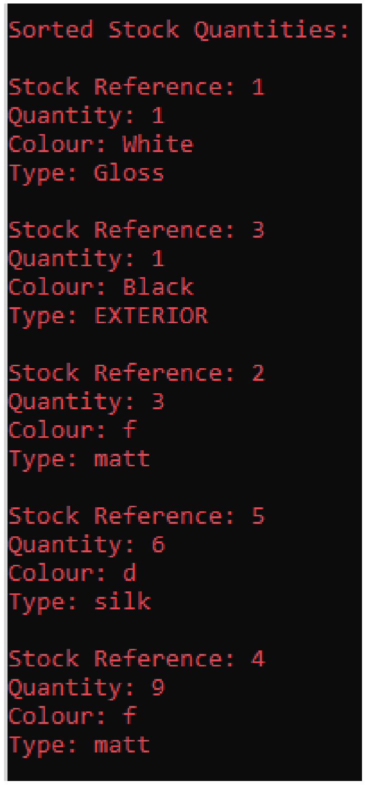
In my system I allow the user to view stock in sorted quantities this is similar to the Xero software since both systems list the quantities alongside other important stock information.

I think that this is an important feature because it allows the company to keep on top of stock control and makes sure they have always got the required stock they need before completing a job. This is important because if they turned up to a customer with none of the right stock it looks unprofessional so by using the sort to allow them to see what they have and haven't got it allows for preparation and organisation. Therefore, I think the similarity in this feature in both systems is essential.

https://www.xero.com/uk/accounting-software/manage-inventory/



| Item Code ▲ | Item Name | Cost Price | Sale Price | Quantity |
|---|---|---|---|---|
| GB6-White | Golf balls - white 6-pack | 20.00 | 24.99 | 0.00 |
| GB9-White | Golf balls - white 9 pack | 25.00 | 32.00 | 0.00 |
| TSL - Black | T-Shirt Large Black | 20.00 | 40.00 | 5.00 |
| TSM - Black | T-Shirt Medium Black | 20.00 | 40.00 | 4.00 |
| TSS - Black | T-Shirt Small Black | 20.00 | 40.00 | 7.00 |

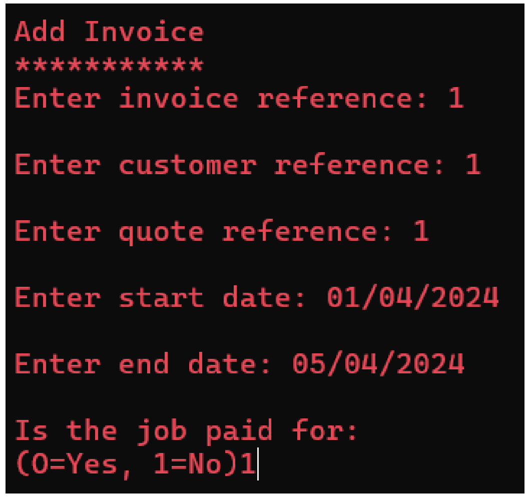


```
Sorted Stock Quantities:

Stock Reference: 1
Quantity: 1
Colour: White
Type: Gloss

Stock Reference: 3
Quantity: 1
Colour: Black
Type: EXTERIOR

Stock Reference: 2
Quantity: 3
Colour: f
Type: matt

Stock Reference: 5
Quantity: 6
Colour: d
Type: silk

Stock Reference: 4
Quantity: 9
Colour: f
Type: matt
```

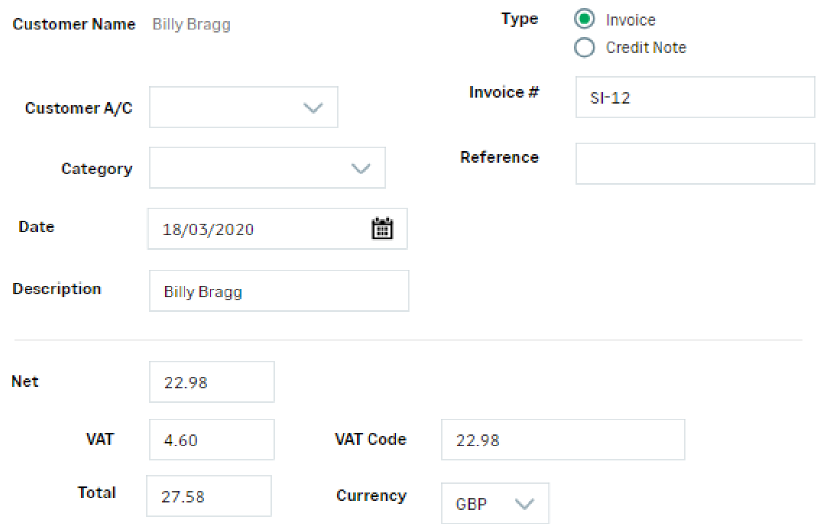
# Add Invoice

When entering information for the invoice the sage system doesn't store the current payment status of the job however, my system does. I think that my systems way of storing the status is better because an important requirement for the company was to help with organisation. Therefore, by having the payment status it helps to keep on top of the financial aspect of the system in a simple way. If they were to use the sage system it would be difficult for the staff to keep track of who had paid and who had not. This leads me to believe that the addition of an invoice on my system is much better for the company.

However, an aspect of the sage system that I think is better is that the user enters the customer name compared to a reference in my system. I think that the sage software's way of linking the customer is better because an employee is much more likely to remember the name of  the customer compared to a number. If I used this feature in my system it would have made the addition potentially easier for the staff so therefore I think that the sage software did this better.



```
Add Invoice
***********
Enter invoice reference: 1

Enter customer reference: 1

Enter quote reference: 1

Enter start date: 01/04/2024

Enter end date: 05/04/2024

Is the job paid for:
(0=Yes, 1=No)1
```

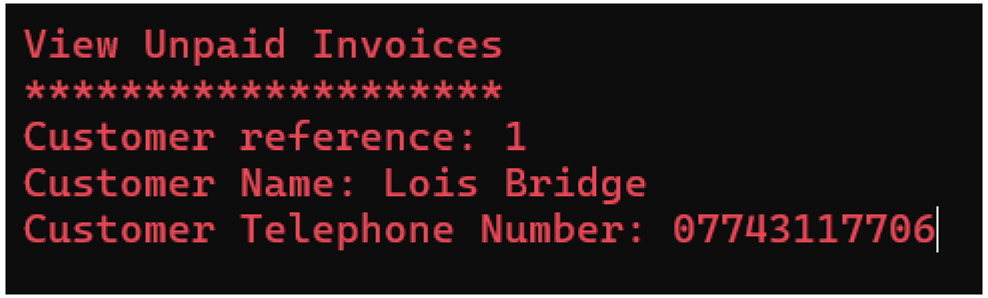


| | | | |
|---|---|---|---|
| Customer Name | Billy Bragg | Type | [✓ Invoice] Credit Note |
| Customer A/C | | Invoice # | SI-12 |
| Category | | Reference | |
| Date | 18/03/2020 | | |
| Description | Billy Bragg | | |
| Net | 22.98 | | |
| VAT | 4.60 | VAT Code | 22.98 |
| Total | 27.58 | Currency | GBP |

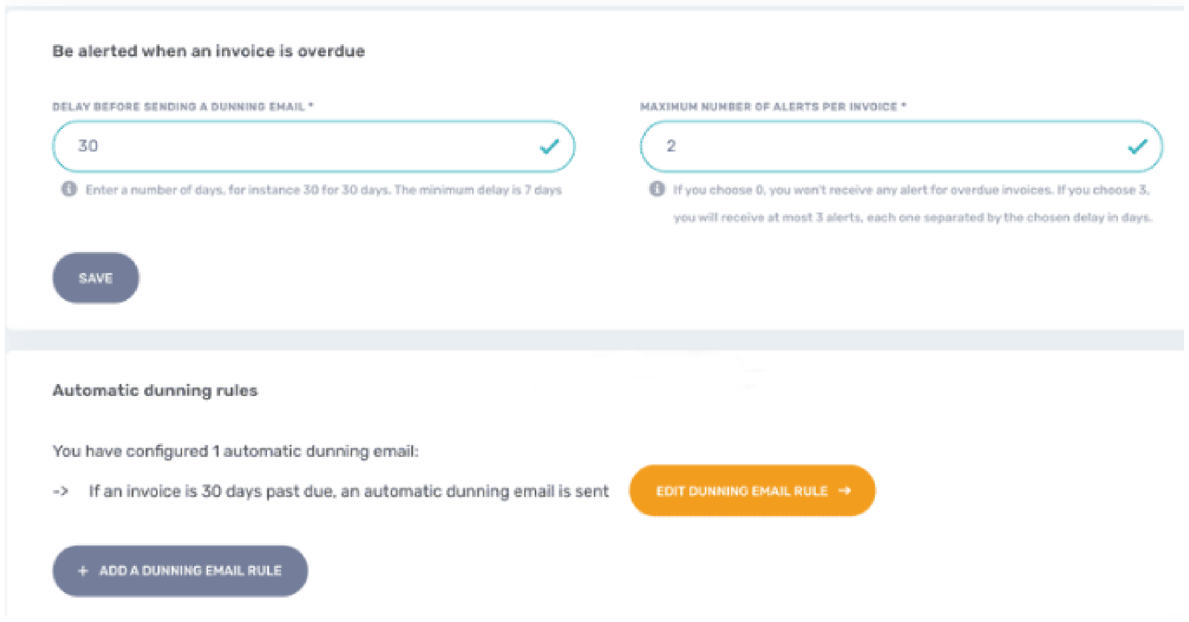
https://www.sage.com/en-gb/accountants/automated-data-processing/

# Viewing unpaid invoices

When looking at the silex system I found that they had a feature to alert customers after a certain time frame had passed. However my system only shows the staff members the unpaid invoices. I think that the silex system's feature is more useful to a system because it gives the employee's one less thing to think about as reminders are automatically sent to the customer. A great similarity of both is that there is no time frame of what unpaid invoices can be viewed unlike other systems available which only show them up to 30 days. My system and the silex system show all unpaid invoices, this is good because it means no invoices are forgotten about, ensuring all customers pay and the company does not lose money. However, as this may have not been a feasible feature for my system I chose to output the customer mobile number so that the customer was easily contactable for the employee this is as similar as I could get to a feature such as the one in the silex system.



```
View Unpaid Invoices
********************
Customer reference: 1
Customer Name: Lois Bridge
Customer Telephone Number: 07743117706
```



Be alerted when an invoice is overdue

DELAY BEFORE SENDING A DUNNING EMAIL *: 30

Enter a number of days, for instance 30 for 30 days. The minimum delay is 7 days

MAXIMUM NUMBER OF ALERTS PER INVOICE *: 2

If you choose 0, you won't receive any alert for overdue invoices. If you choose 3, you will receive at most 3 alerts, each one separated by the chosen delay in days.

Automatic dunning rules

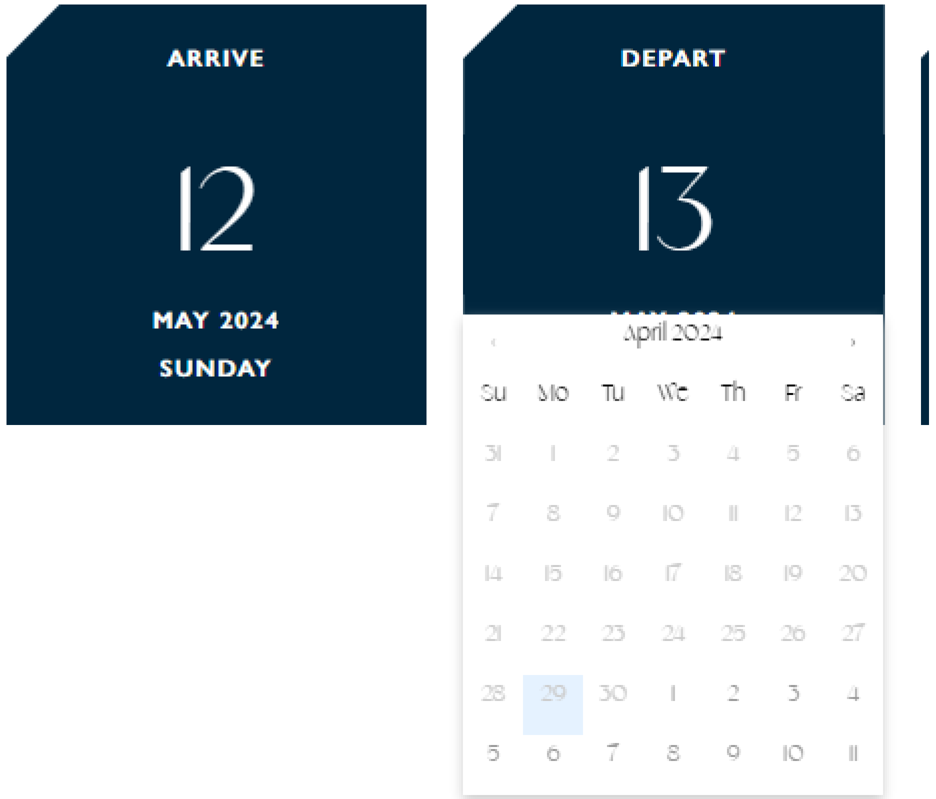
You have configured 1 automatic dunning email:

-> If an invoice is 30 days past due, an automatic dunning email is sent

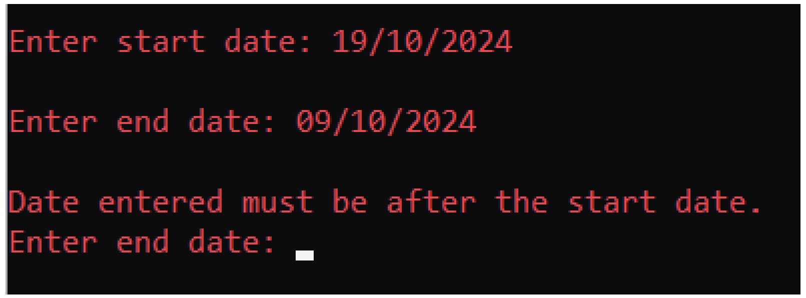
https://silex.pro/en/software/outstanding-invoices-management/

# Date range - Validation

A similarity of both systems is that when a booking is being entered into each of the systems the user cannot enter an end date which was before the start date. This is a great feature of both our systems because without it incorrect bookings could be made which would create confusion and problems for both the staff and the customer. However, this is executed differently in both systems. In my system I allow the user to enter the date via the keyboard whereas this hotels system uses a calendar where you select the date. I believe that my system is better since it is more tailored to Stuart and the company because they are not used to a computer system so having to select the date from such a small icon would not help to make the system easier to use for them.



https://www.thequeensleeds.co.uk/?gad_source=1&gclid=EAIaIQobChMIsaOooIXnhQMVH4lQBh0EuAldEAAYASAAEgJnHfD_BwE

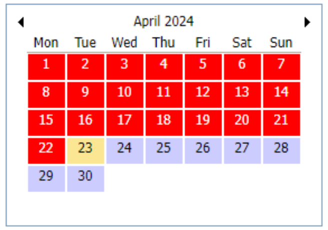

```
Enter start date: 19/10/2024

Enter end date: 09/10/2024

Date entered must be after the start date.
Enter end date: _
```

# Schedule – Addition / Validation

For my system I allowed the user to add to the schedule by entering the date they required I think that this is better than the online booking system shown here because you are more likely to press the wrong date on a mini calendar whereas if you are entering the date yourself you are actively thinking of the date required so it means you are less likely to enter an incorrect date.

Furthermore, on the Tradify schedule it allows a user to enter anything onto the schedule however my system only allows a booking to be added if the quote reference exists. I think that this is better since it means the schedule is much less likely to hold errors or even to book up a staff member when the quotes not even valid which would waste a staff members time.

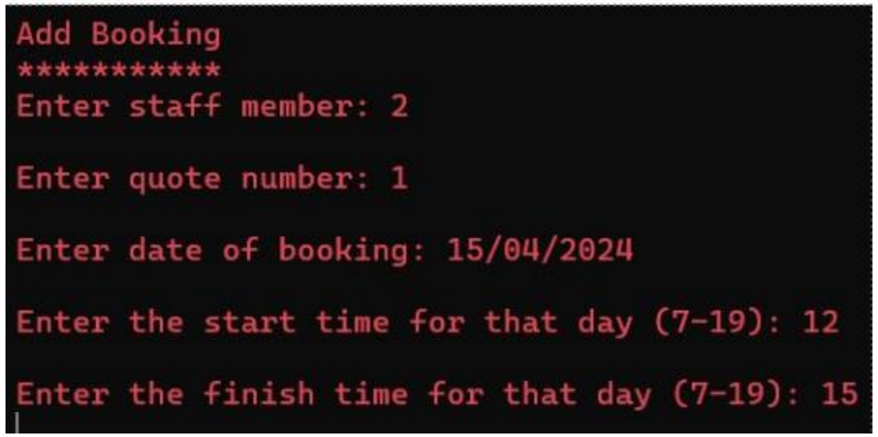
**Step 02 - Date Selection**



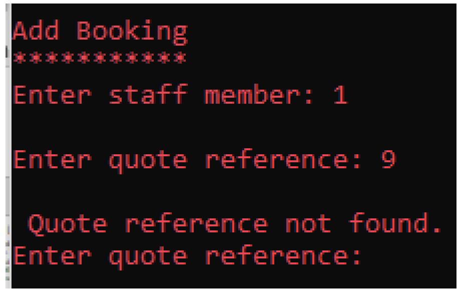
April 2024

| Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | | | | | |

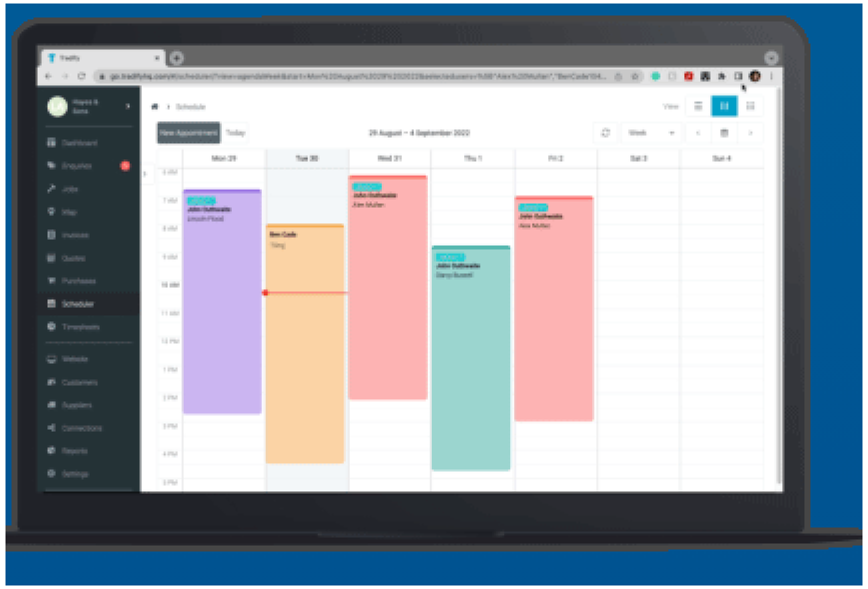
https://accessvam.accessacloud.com/BletchleyParkBookings/BookDate.aspx?eid=0022



```
Add Booking
***********
Enter staff member: 2

Enter quote number: 1

Enter date of booking: 15/04/2024

Enter the start time for that day (7-19): 12

Enter the finish time for that day (7-19): 15
```



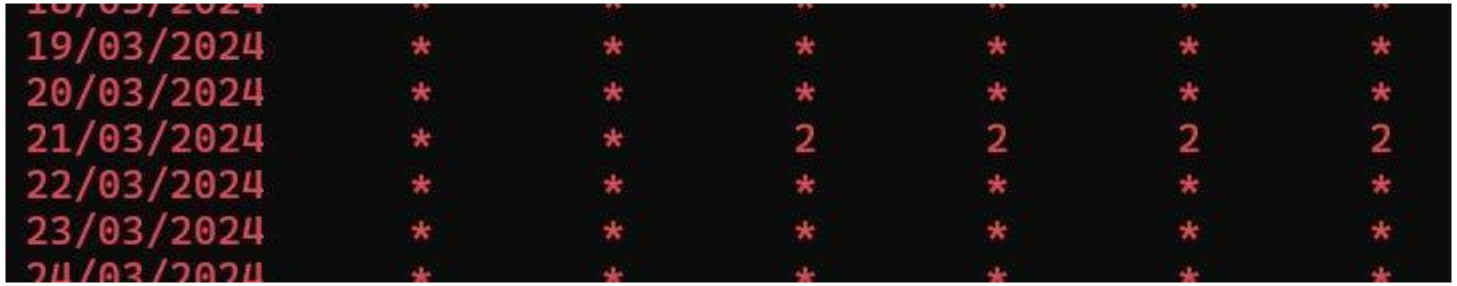
```
Add Booking
***********
Enter staff member: 1

Enter quote reference: 9

 Quote reference not found.
Enter quote reference:
```



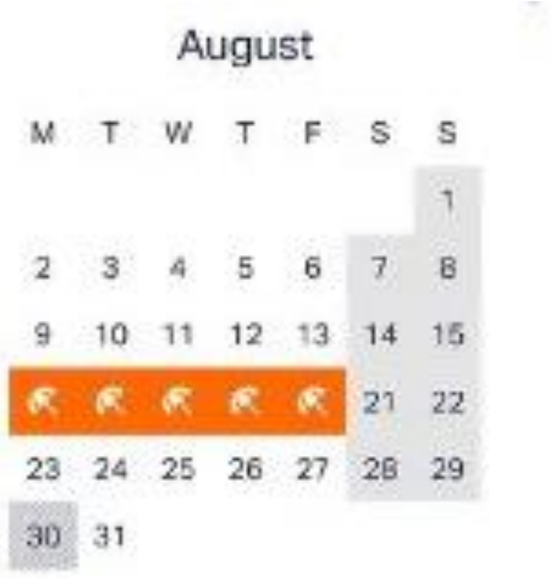
https://www.tradifyhq.com/uk/features/scheduling-software

# View schedule

When viewing the staff schedule  a great feature of the SW is the use of small icons to represent different things unlike my system which shows either free or a quote reference. I think that the SW is better because it allows more information to be stored on the schedule in quite a simple way. I think that this would be useful because it allows other staff members to see when each other is in or when they are on holiday so they don't contact them.



```
19/03/2024      *       *       *       *       *       *
20/03/2024      *       *       *       *       *       *
21/03/2024      *       *       2       2       2       2
22/03/2024      *       *       *       *       *       *
23/03/2024      *       *       *       *       *       *
```



https://timetastic.co.uk/