Design

Customer Relationship Management System (CRM)

Kieran Williams

Break Down Of the System

This breakdown will be referenced throughout the rest of the design as it includes all objectives and will allow for me to link everything to the various objectives

*** number's contained in brackets reference the objectives that the sub-problems relate to ***

The site needs to be accessible through the use of a web browser when the domain name is typed into the browser (1)

↳ The site will need to be hosted on a web server

↳ A domain name will need to be bought and set to point to the ip address that the site is hosted at

↳ SSL will need to be set up so that the transfer of data from the server to the client is encrypted. (12)

The site will need to be easy to navigate through. (9)

↳ The system will need to remain simplistic

- ↳ Language used will need to. Be simple not complex
- ↳ The number of links on a page should be kept to a minimum
- ↳ There will need to be good HCI
- ↳ To get to any page, it should be only few links deep to get to.

↳ The navigation bar will need to have 2 different modes dependent on if a user is logged in or not

↪Navigation bar when user is not logged in

↪ will contain links to re-direct the user to the home page, log in page and register page

↪Navigation bar when user is logged in

↪ will contain links to re-direct the user to the home page (dashboard), contacts page, the analytics page and the settings page.

↪ When no user is logged in there will need to be a home page

↪ There will need to be a greeting message and links to re-direct them to the log in and to register account page.

The CRM needs to contain registration and login features

↪ The Registration form will require entry of: first name & last name, business name, email address and password along with a submit button. (2)

↪ The fields will go through validation before being submitted to ensure they all meet the requirements for security reasons and to ensure the system works as intended (eg. emails can't be sent to non valid email addresses). (20)

↪ There will then need to be a screen verification that makes the user confirm that all details are correct to try prevent human error in the system (21)

↪ A hashing algorithm will be run on the password with a salt

↪ The password will be stored as a hash with a salt so that it isn't directly stored in the database (5) (12)

↪ An account with a unique id will then be created and added to the users table in the database which will then be used to relate the data associated with that account (10)

↪ The login requires a username, a password and submit button in a form (3)

↪ The password will get checked that it contains all password look up validation requirements before querying the database to login so that the server isn't unnecessarily processing data. (20)

↪ The username will be located in the table and then the hash algorithm with the salt used will be run on the inputted password to see if the output matches the hash in the database

↪ If the credentials match, the user will be logged in and a transit cookie will be stored in the browser's memory to allow access to the data in the database related to that user id.

↳ There will need to be a way of getting into the account if the user has forgotten the password.

↳ There will be a button on the log in page that will take the user's email in a dialog and then that user will receive an email with a link confirming that the request was made by them.

↳ Upon clicking the link, the user's password will be changed to something random and the user will be emailed the new password so that they can gain access to their account.

There needs to be a way of importing and exporting contacts as CSV files (6)

↳ A file upload object on a page can be used to import the contacts CSV file

↳ The File uploader will have to check that the file is of a CSV file type (20)

↳ Some checks should be done on the CSV file to ensure it is in the correct format to be imported

↳ There should be a certain number of rows

↳ The validation methods used when adding a contact should be appropriately preformed on each record of each contact in the CSV file.

↳ Any errors found in the file should be outputted to the user as a message on how to fix the problem

↳ If the file passes all validation, the data in the file should be used to create contacts.

↳ A button to download all contacts in a CSV file should be on the same page as the file uploader

↳ A program will be run to collect all contacts stored and their data, then put it into the correct format in a CSV file before sending it to the client's machine to be downloaded

There needs to be a way of viewing all contacts stored (27)

↳ A page containing links to all contact pages with a few contact identifying fields for each will be displayed on the page in a list.

↳ There will have to be a way of sorting the contacts by different methods and different fields in both ascending and descending order (22)

↳ sorting methods will be preformed on the appropriate fields and then the sorted list will get outputted

↳ Sorts will include alphabetically by name, business name, city and by date the contact was created.

↳ The contacts in the list will need to be searchable. (24)

↳ There will need to be a method that looks for a matching sequence of characters in the fields to what is in the search bar.

↳ All contacts that don't have the matching sequence of charters in any displayed fields will get removed from the list when the search bar gets entered

↳ This page would also be an appropriate page for the buttons: add contact, import contacts via csv and export contacts as csv.

↳ The import contacts would open a file upload dialog for uploading the csv file

↳ The export csv file would download the users's contacts as a csv file onto their machine.

↳ clicking the "add contact" button should re-direct the user to the create contact page.

There needs to be a way of viewing all details stored on each individual contact

↳ There will need to be a page for each contact

↳ It will need to display the details stored on the contact (28)

↳ It will include a text area to write notes and save them (15)

↳ Any notes that get saved will be put in a list, ordered chronologically, displayed on the page. (16)

↳ It will include a modifiable text area that displays whatever is in it upon being saved. (19)

↳ This text area will be able to encrypt the text if a password is put in for it (17)(12)

↳ The password will need to have check preformed to ensure that the password is different to that of the account login password. (18)

There needs to be a way of adding editing and deleting contacts (26)

↳ There will need to be a link for creating a contact

↳ All details that will need input relating to a contact will be put into a form with blank text boxes to be filled in.

↳ Validation will be done on the text boxes where necessary. (20)

↳ Upon successful submission, a new record will be appended to the contact's table.

↳ There will need to be a a link for modifying any contact

↳ A editing page will then open with a form containing text boxes filled out with the data stored on the contact.

↳ Changes to the data in the text boxes will be possible.

↳ Validation will be done on the text boxes where necessary. (20)

↳ Verification that the user has checked all the details over will be required (21)

↳ Upon successful submission, the record will be updated with the new data.

↳ There will need to be a link for deleting a contact next to the contact in the view page

↳ A confirmation dialog for verification of the deletion should then open (21)

↳ The contacts details should then get removed from the database

↳ The page should then reload with that contact no longer present in the list

There needs to be visual representations of analytical data stored

↳ There will be a graphical representation of clients gained and lost over a period of time (7)

↳ There will be a way of changing the time scale (yearly/ monthly/ weekly/ daily)

↳ The x-axis will show the time scale. The y-axis will be changeable to show: clients lost, clients gained and the number of clients

↳ The averages used as data points on the graph, will be calculated with different data dependent on the time scale chosen.

↳ daily won't require averaging and will show 14 consecutive days of data points on the graph

↳ weekly will take the average of each 7 consecutive days and show 12 consecutive weeks of data points on the graph

↳ monthly will take the average of 4 consecutive weeks and show 12 consecutive months of data points on the graph

↳ yearly will take the average of 12 months and show 10 consecutive years of data points on the graph

↳All of the above could utilise a recursive algorithm

↳ There will need to be a way of inputting the data that a client has been gained/ lost and one for a lead being lost. (11)

↳ A link on the analytics page could take the user to a new page to enter data

↳ There could be 3 buttons: client gained, client lost and lead lost.

↳ There could also be a reset button to remove all data stored

↳ This data should get stored with a time stamp.

↳ By using the data stored in the gained clients, lost clients and lost leads tables, A conversion rate for the month could be calculated and displayed in the analytics page.

↳ The algorithm will get the number of contacts gained and all the leads lost within the month then find a common factor to divide them both by. The output will be a in the form of a ratio (eg. “Conversion Rate = 2 : 3”) (8)

↳ the conversion rate could also have a button which alternates the conversion rate from a ratio into a percentage.

↳ This would be done by dividing the number of clients gained within the month by (the number of clients gained + the number of leads lost within the month)

There needs to be a dashboard that shows important data that needs to be readily viewable to the user

↳ There must be a way of scheduling tasks and appointments from the dashboard (13)

↳ There could be a form which has an input for the date and time along with a text area which will be for entry of what the task or appointment is.

↳ Validation will be preformed on the date and time to ensure it is both in the correct format and that nothing else is booked for that date and time.

↳ A message could pop up to warn the user if the task or appointment is within an 8 hour interval of another task or appointment.

↳ There will need to be a way to view all tasks or appointments that are upcoming and anything scheduled for the current day should be made prominent on the dashboard. (14)

↳ There can be a scrollable area where the appointments and scheduled tasks are put into in chronological order. The scheduled tasks and appointments for the day could be made larger and separated from the other tasks and appointments.

↳ There should also be a way to delete anything that has been scheduled.

↳ The scheduling’s should get removed from the database after the day the appointment was scheduled for has passed.

There needs to be accessibility, security and privacy measures in place (12)

↳ Users should be able to change their details such as their email address, business name and their password. (4)

↳ There will be a link on the settings page that will redirect them to a new page with their email, password and business name in text fields contained in a form. This will allow the user to edit their information.

↳ The data will undergo appropriate validation. (20)

↳ The user will then be asked to verify the changes they wish to make. (21)

↳ Upon successful submission, the database will get updated with the new data.

↳ The system must be GDPR compliant to adhere to the law. (25)

↳ There will need to be a terms and conditions agreement page that is clearly available for the user to access at any time

↳ This will outline the data that is being kept and collected on them, outlining how long the data will be kept on them.

↳ There should be use of SSL as a security measure for the users. (12)

↳ Any consent forms that the user could have to submit such as the terms and conditions agreement form with a checkbox will need to be unchecked when displayed to them.

↳ There should be an easy point of access for the user to query what data is being stored on them.

↳ There must also be a way for the user delete their account and all data stored on them.

Inputs And Outputs

Inputs

↳ Navigation bar buttons

↳ Settings links

↳ Account details

↳ Create account

↳ Log into account

↳ Forgotten password

        - ↳ Edit details
        - ↳ Delete Account
    - ↳ Terms of use page
        - ↳ Checkbox for accepting terms of use
    - ↳ Data held on user request link
- ↳ Contact details
    - ↳ Create contact
    - ↳ Edit contact details
    - ↳ Delete contact
- ↳ Contacts search bar
- ↳ Contacts filtering
- ↳ Contacts sorting
- ↳ Scheduled tasks and appointments
- ↳ Buttons for: lead lost, client gained, client lost, reset values
- ↳ CSV file uploader to upload contacts and their details into the database

Outputs

- ↳ View account details
- ↳ View contacts in a list
- ↳ View a contact's details
- ↳ Graph for client's lost, client's gained and number of clients.
- ↳ Conversion rate for the month
- ↳ Viewing scheduled appointments and tasks
- ↳ Downloading a CSV file of all contacts and their details

The Pages containing the input and outputs

- ↳ Login page.
- ↳ Register page.
- ↳ Home page when user is not logged in.
- ↳ Dashboard page.
- ↳ Contacts list page.
- ↳ Create contact page.
- ↳ Individual Contact's page.

↳ Editing contact's data page.

↳ Analytics page.

↳ Settings page.

↳ Terms and conditions page.

↳ Edit account details page.

The pages will be mocked out below with a description of what the page is showing in regards to inputs and outputs with a reference (denoted by the "*" either side) to the title it comes under in the "breaking down of the system" part of this design. I have done this because the sub-problems make up for all of the objectives outlined, and therefore justify the reasoning for the inputs and outputs with respect to the objectives. The "breaking down of the system" will also describe the processes that will occur after an input has taken place and any outputs that will occur as a result.

Navigation bar

The site will need to be easy to navigate through

The navigation bar will be a horizontal strip on top of every page on the site. It will take up the entire width of the browser.

Navigation bar when no user is logged in

CRM		Log in	Register

The navigation bar will consist of three links (inputs): "CRM", "Log in", and "Register".

Navigation bar when a user is logged in

CRM	Home	Contacts	Analytics	*business name* 

The navigation bar will consist of a label "CRM", and four links to the right of the label.

The four links (inputs) include: “Home”, “Contacts”, “Analytics”, and Settings.

The Settings link will have the (output) text as the user’s business name and next to it, a ‘gear’ glyphic.

Home page when user is not logged in

*** The site will need to be easy to navigate through ***

There Will be a title/ label as “Welcome To The CRM”.

Below the title will be two buttons (inputs): Log in, and Register Account.

Analytics Page

*** There needs to be visual representations of analytical data stored ***

At the top there is 3 graphs with titles appropriately named above them (Outputs). The Y-axis of the graph will represent what the title of the graph. The X-axis will represent the time scale. The data points of the graph will change dependent on the time scale.

The Time scale can be set by the drop down menu to the right of the graphs. The drop down menu options will include: “Yearly”, “Monthly”, “Weekly” and “Daily” (Input).

Below the graphs there is the label “conversion rate this month” right to that is the conversion rate as a ratio and as a percentage (Outputs).

To the left of the conversion rate, there will be four buttons (Inputs) for: “client lost”, “client gained”, “lead lost", and to “reset all data”.

Settings Page

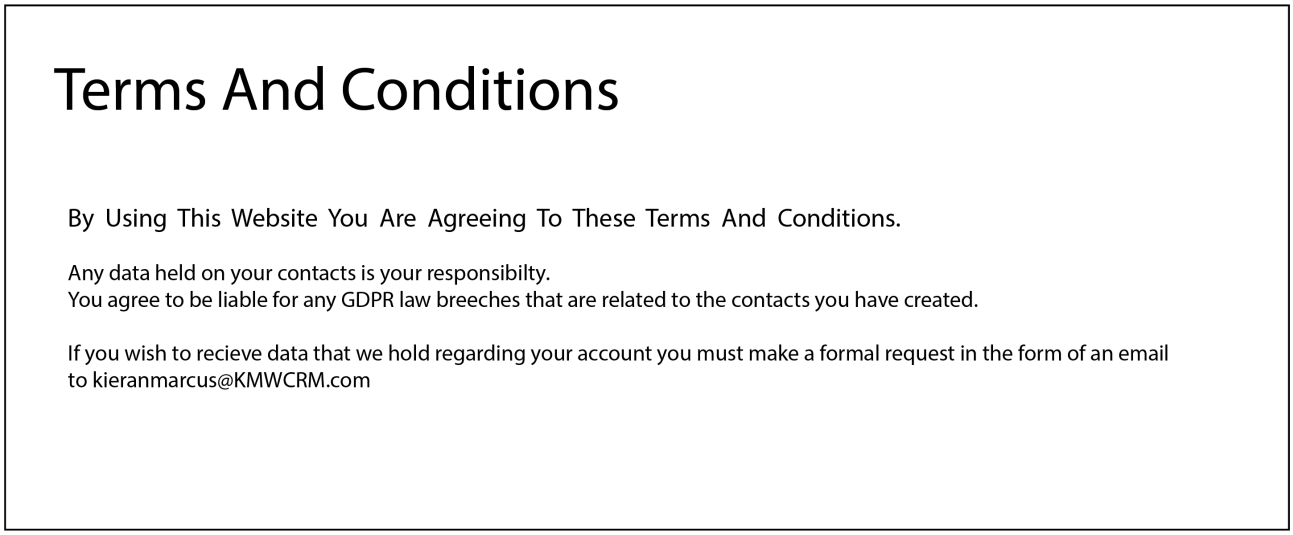
*** There needs to be accessibility, security and privacy measures in place ***

There will be a label/ title “Settings”.

Below the the title there will be two links (inputs): “Edit Account Details”, and “Terms And Conditions”.

Terms And Conditions Page

*** There needs to be accessibility, security and privacy measures in place ***

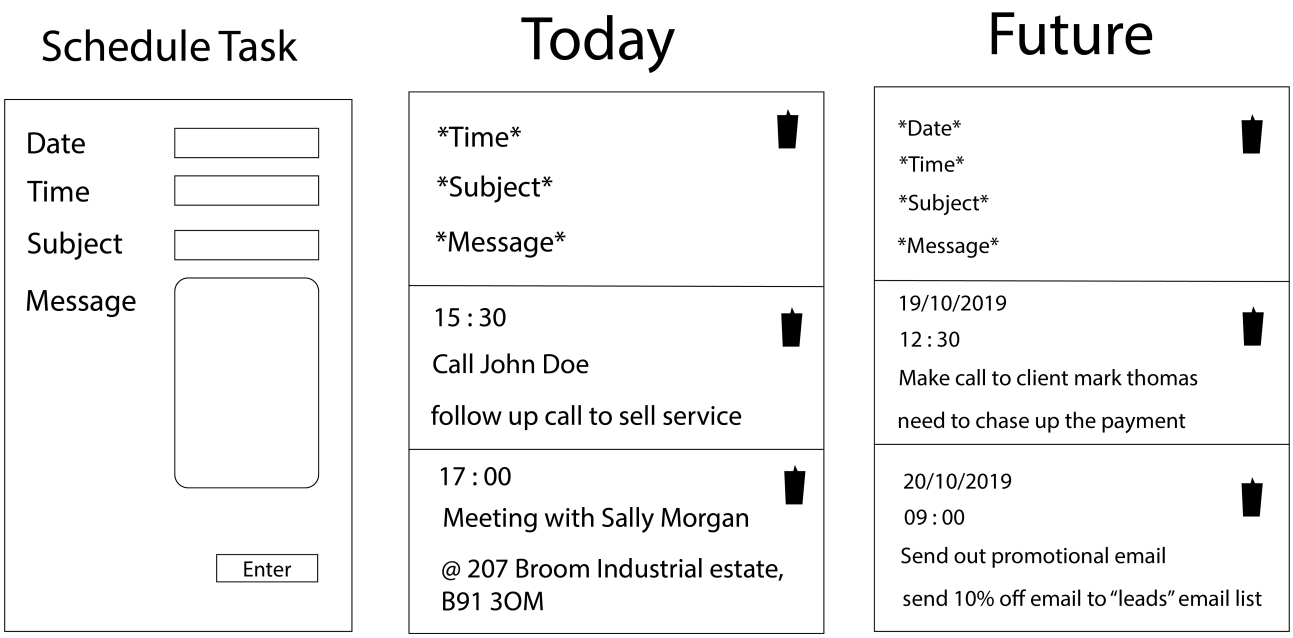


There will be a title/ label on the top of the page “Terms And Conditions”.

Bellow the title will be the terms and conditions for use of the service in paragraphs as plain text

Dashboard Page

*** There needs to be a dashboard that shows important data that needs to be readily viewable to the user ***



The Dashboard will be divided into 3 sections

The first third/ on the left, will be a form.
On the top of the form there will be a title/ label “Schedule Task”.
In the form there will be labels: “date”, “time” “subject”, and “message”.
To the right of the first 3 labels will be text boxes (inputs).
The the right of the “message” label, will be a text area (input).
At the bottom of the form will be a submit button “enter” (input).

The second third/ in the middle will be a list.
On top of the list will be a title/ label “Today”.
Each list item will contain a label for the time, the subject and a message (outputs).
Next to each list item will be a button (input) with a ‘trash can’/ ‘bin’ glyphic displayed as the button.
The list will be contained in the rectangle. There will be a scroll bar when there is an overflow of list items.

The last third/ to the right will be a list.
On top of the list will be a title/ label “Future”.
Each list item will contain a label for the date, time, the subject and a message (outputs).
Next to each list item will be a button (input) with a ‘trash can’/ ‘bin’ glyphic displayed as the button.
The list will be contained in the rectangle. There will be a scroll bar when there is an overflow of list items.

Contact's List Page

* **There needs to be a way of viewing all contacts stored ***

* **There needs to be a way of importing and exporting contacts as CSV files ***

* **There needs to be a way of adding editing and deleting contacts ***

The contacts list page will be split into two sections.

On the left of the page/ in the first section, there will be two containers, one higher on the page than the other.

On top of the first/ top container, there will be a label/ title "Contacts".

In the first/ top container, there will be a form which will contain: a label "Search" to the left of a text-field (input).

Under the "Search" label, there will be another label "Sort By".

Under the "Sort By" label will be 3 radio boxes (inputs) where only one will be selectable at a time. The radio boxes will have the appropriate label's next to them. The labels next to the radio boxes will be: "Date", "City", "Business Name".

Under the radio boxes there will be a dropdown menu (input) with the choices being "ascending" and "descending".

At the bottom of the form, there will be a submit button (input).

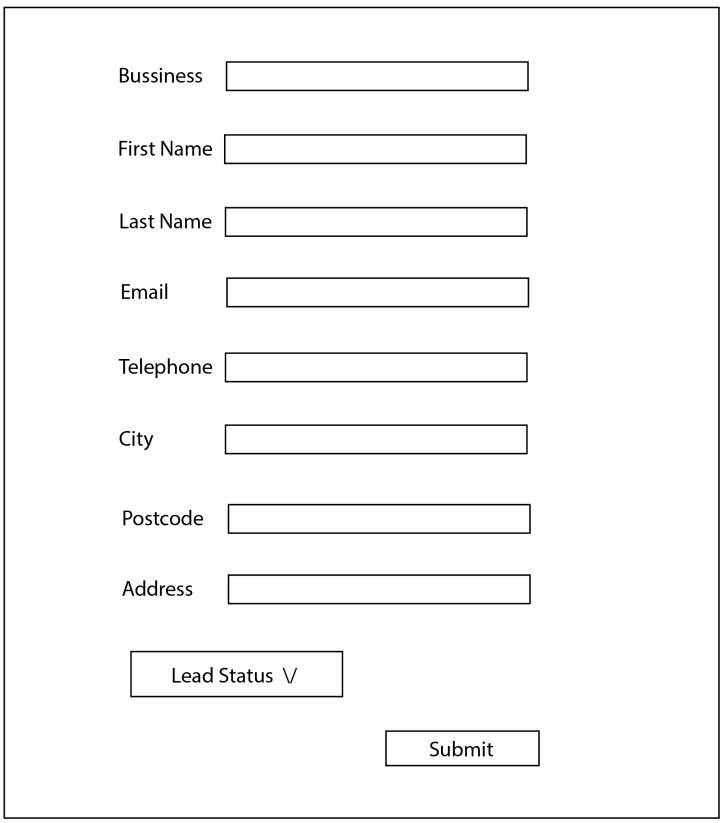
In the second/ bottom container there will be 3 buttons (inputs): “Add Contact”, “Import Contacts”, “Export Contacts”. These buttons will be one above the other.

On the right of the page/ in the second section there will be a table
On top of the table’s columns will be labels/ titles for what data each column holds.
The labels will be: “Name”, “Business”, “City”, “Telephone”.
Each row will represent a contact and their details.

In each element of the first column will be a link to that contact’s individual page with the text being the contact’s name (inputs/ outputs).
In the other elements of the table, there will be a label (output) containing textual data relevant to that column and contact.
The table will be contained in the rectangle. There will be a scroll bar when there is an overflow of rows.

Create Contact Page

*** There needs to be a way of adding editing and deleting contacts ***



The Create Contact Page will contain eight labels: “Business”, “First Name”, “Last Name”, “Email”, “Telephone”, “City”, “Postcode”, “Address”. Next to each label will be a text-box (inputs).

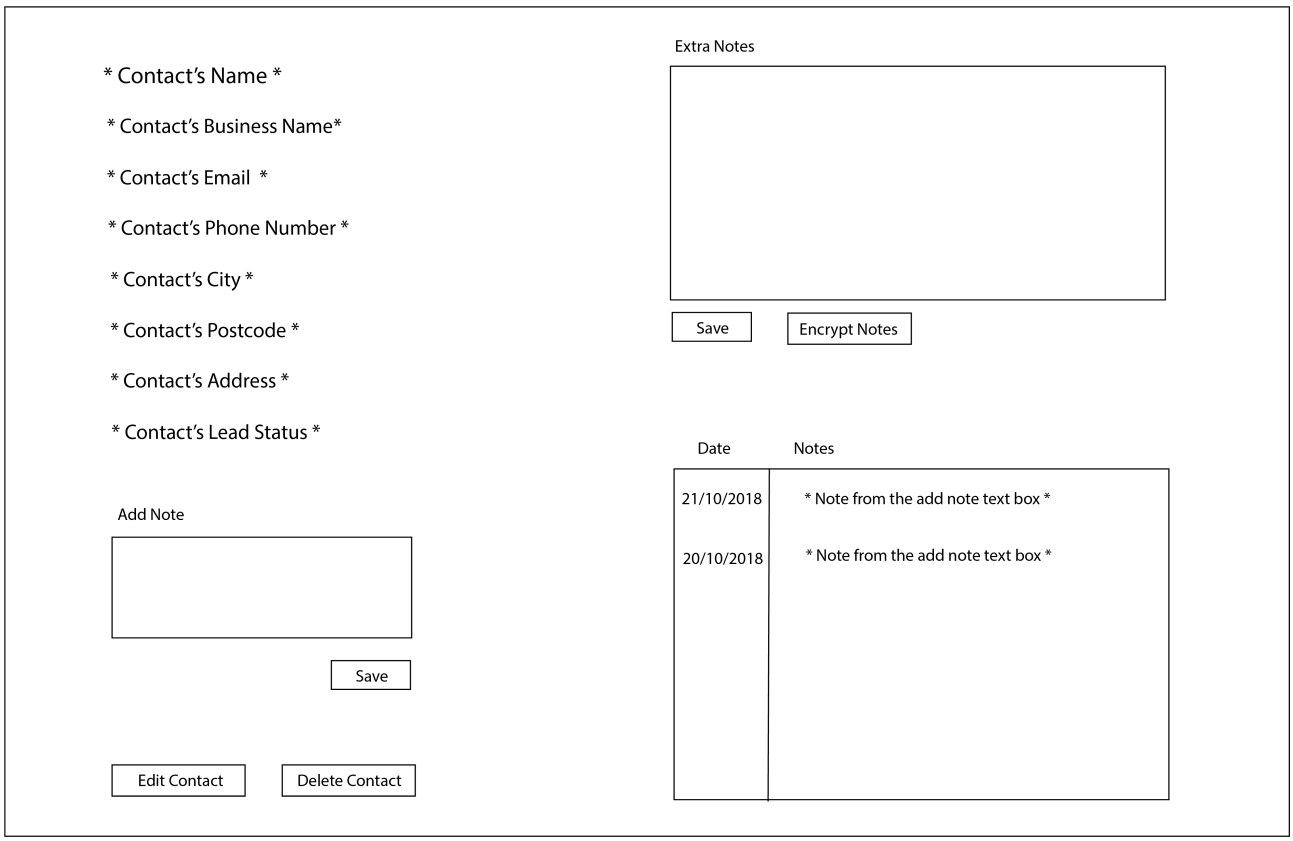
At the bottom, under all the text boxes and labels will be a drop down menu (input) with the options: “Recontact Lead”, “Dead Lead”, “Current Client”, and “Past Client”.

At the bottom of the form will be a submit button (input).

View Individual Contact Page

* **There needs to be a way of adding editing and deleting contacts ***

* **There needs to be a way of viewing all details stored on each individual contact ***



On any contact's individual page there will be two halves to the page separated by a container.
On the first half of the screen/ left side, there will be three sections one above the other.

In the top section on the left, there will be the contact's details as labels (outputs).
The labels will be the contact's: name, business name, email, phone number, city, postcode, address, and contact status. The labels will be one above the other.

In the middle section there will be a form (input).
The form will contain the label/ title "Add Note".
Below the "Add Note" label will be a text area (input).
At the bottom of the form will be a submit button (input) "Save".

In the bottom/ last section there will be two buttons (inputs): "Edit Contact" and "Delete Contact"

On the right side there will be two sections, one above the other.

In the top right section, there will be a form (input). The form will contain a label/ title “Extra Notes”.
Under the “Extra Notes” label, there will be a text area (input/ output) this will be editable and will also display whatever is saved in it.
At the bottom of the form there will be two buttons (inputs): “Save” and “Encrypt Notes”.
Upon the “Encrypt Notes” button being clicked, a dialog (input) will appear with a text field for password entry, this will encrypt or decrypt the text in the text area.

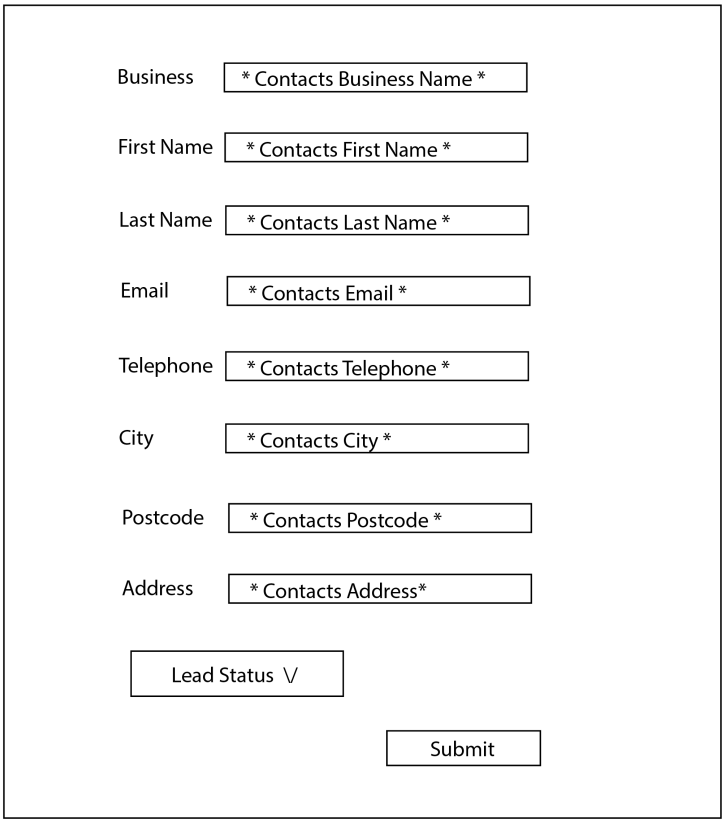
In the bottom right section there will be a table with 2 columns, the second column will be wider than the first.
There will be a label above the first column “Date” and a label above the second column “Notes”
In each row of the first column there will be a label containing a date (output), and in the second column of that row, there will be a note that was created on that date (output).

The table will be contained in the rectangle. There will be a scroll bar when there is an overflow of rows.

Edit Contact Page

*** There needs to be a way of adding editing and deleting contacts ***

Edit Contact



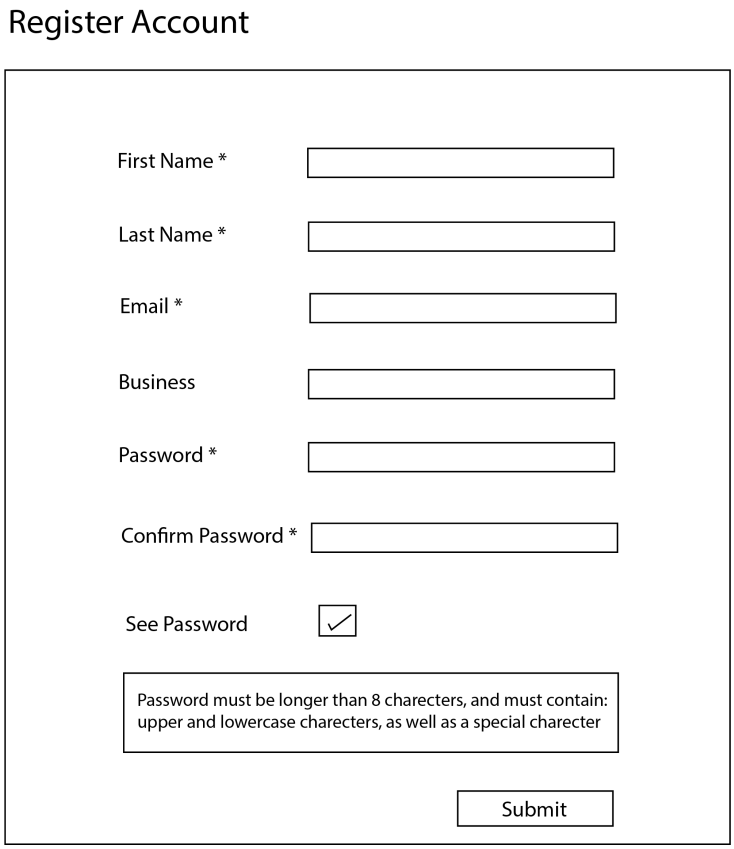
The Edit Contact Page will contain eight labels: “Business”, “First Name”, “Last Name”, “Email”, “Telephone”, “City”, “Postcode”, “Address”. Next to each label will be a text-box with the data currently saved on that contact in them (inputs/ outputs).

At the bottom, under all the text boxes and labels will be a drop down menu with the current status of the contact shown (input/ output). The drop down menu will have the options: “Recontact Lead”, “Dead Lead”, “Current Client”, and “Past Client”.

At the bottom of the form will be a submit button (input) which will save any changes.

Register Account Page

*** The CRM needs to contain registration and login features ***



The Register Account Page will contain seven labels: “First Name”, “Last Name”, “Email”, “Business”, “Password”, and “Confirm Password”. The passwords will be seen as black dots when the user types in the password. Next to each label will be a text-box (inputs).

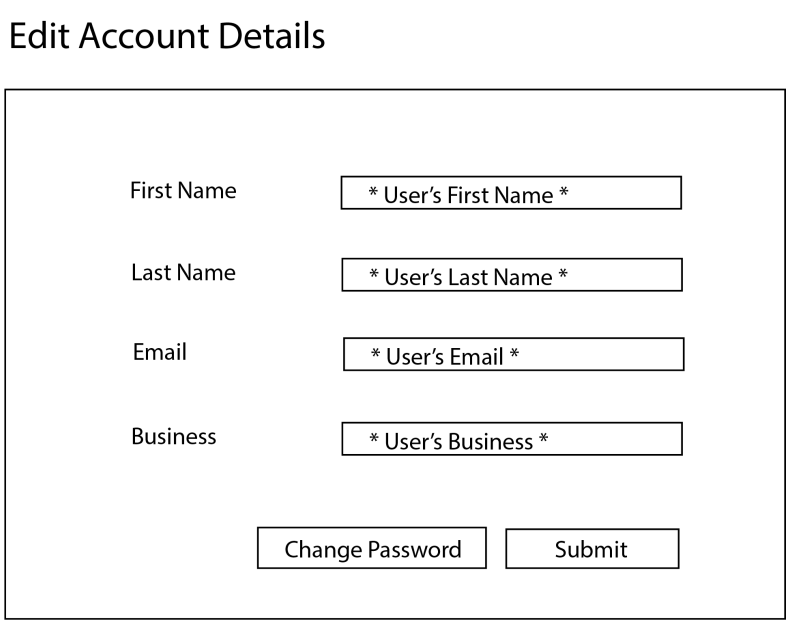
Under all the text boxes there will be a checkbox (input) with the label beside it “See Password”, if the checkbox is checked, then the passwords will become visible.

Below the check box will be a message in a box describing the conditions that are to be met for a password to be accepted and hence to be able to register an account.

At the bottom of the form will be a submit button (input).

Edit Account Details Page

*** There needs to be accessibility, security and privacy measures in place ***



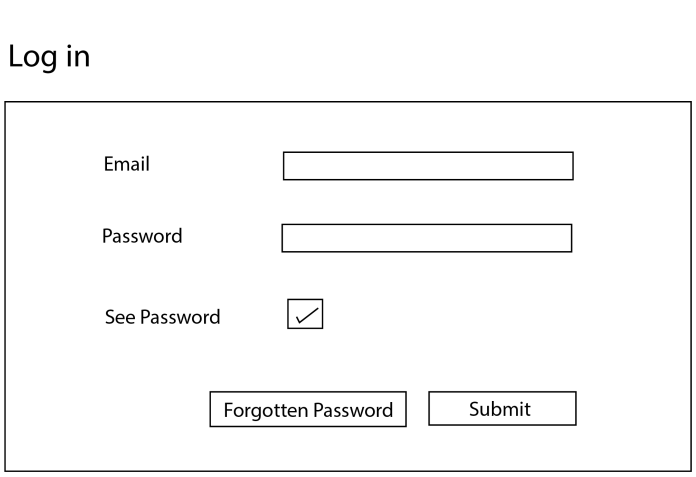
The Edit Account Details Page will contain four labels: "First Name", "Last Name", "Email", "Business". Next to each label will be a text-box with the data currently saved on that user in them (inputs/ outputs).

At the bottom of the form will be a "change password button" and a "Submit" button (inputs).

(The "Change Password" button will re-direct the user to the change password page).

Log in Page

*** The CRM needs to contain registration and login features ***



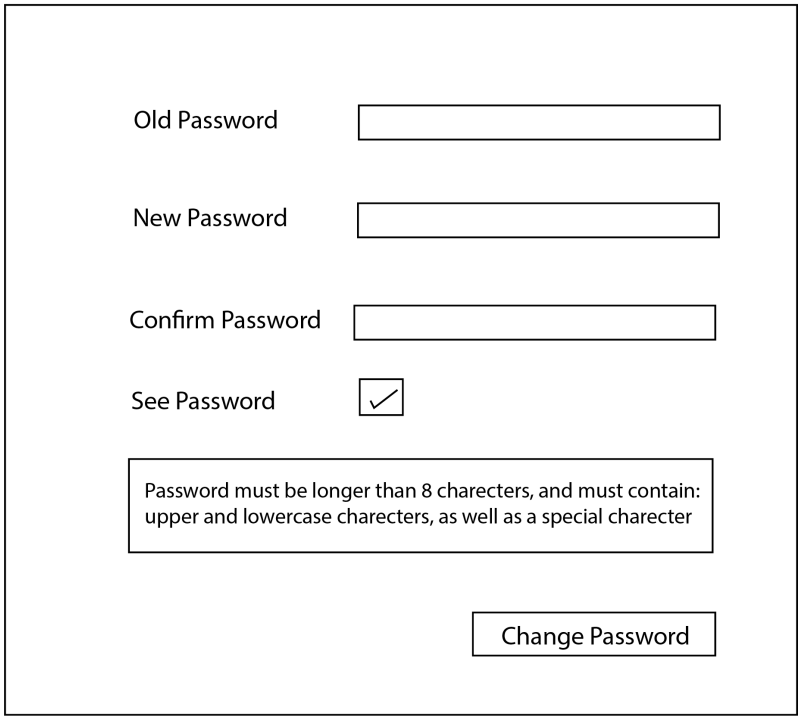
The Log in Page will contain labels: “Email”, “Password” and “See Password”. The passwords will be seen as black dots when the user types in the password. Next to the “Email” and “Password” labels will be a text-box (inputs).

Next to the “See Password” label will be a checkbox (input), if the checkbox is checked, then the passwords will become visible.

At the bottom of the form will be a “Forgotten Password” and a “Submit” button (input).

Change Password Page

*** There needs to be accessibility, security and privacy measures in place ***



The Change Password Page will contain three labels: “Old Password”, “New Password”, and “Confirm Password”. Next to each label will be a text-box (inputs). The passwords will be seen as black dots when the user types in the text boxes.

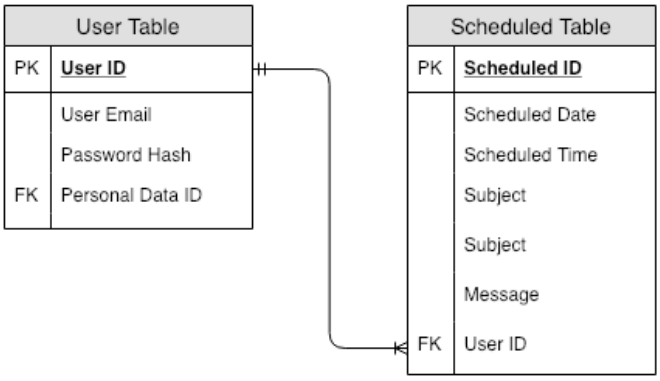
Under all the text boxes there will be a checkbox (input) with the label beside it “See Password”, if the checkbox is checked, then the passwords will become visible.

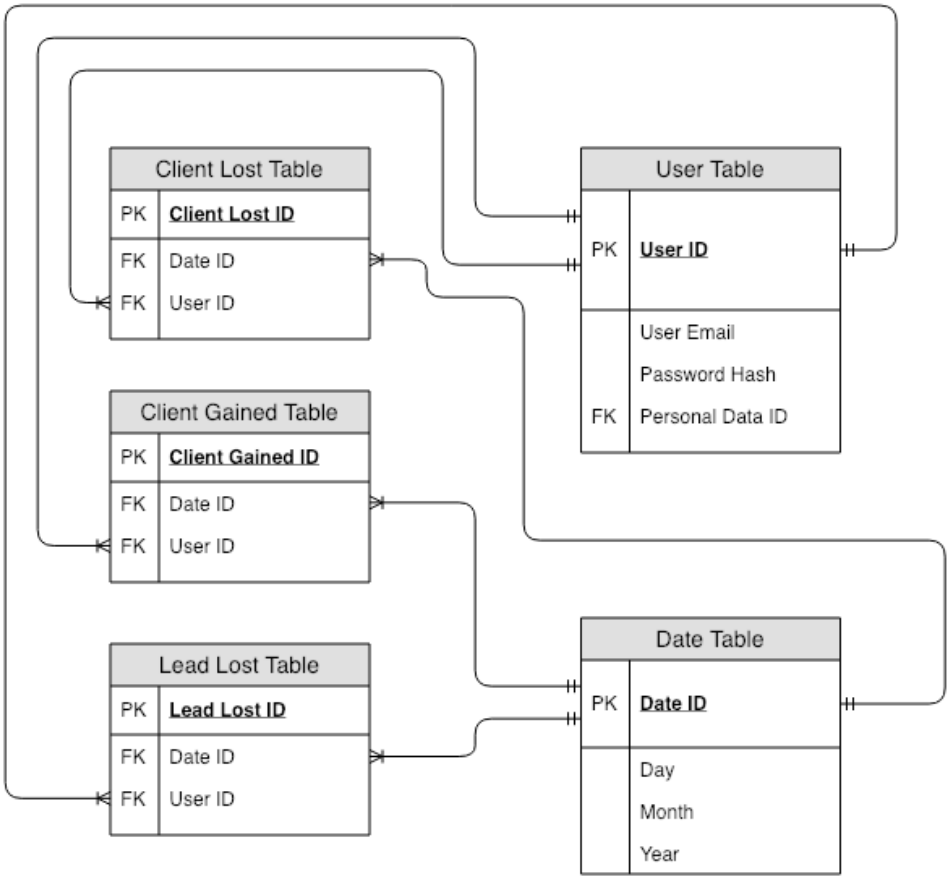
Below the check box will be a message in a box describing the conditions that are to be met for a password to be accepted and hence to be able to change the password to the account.

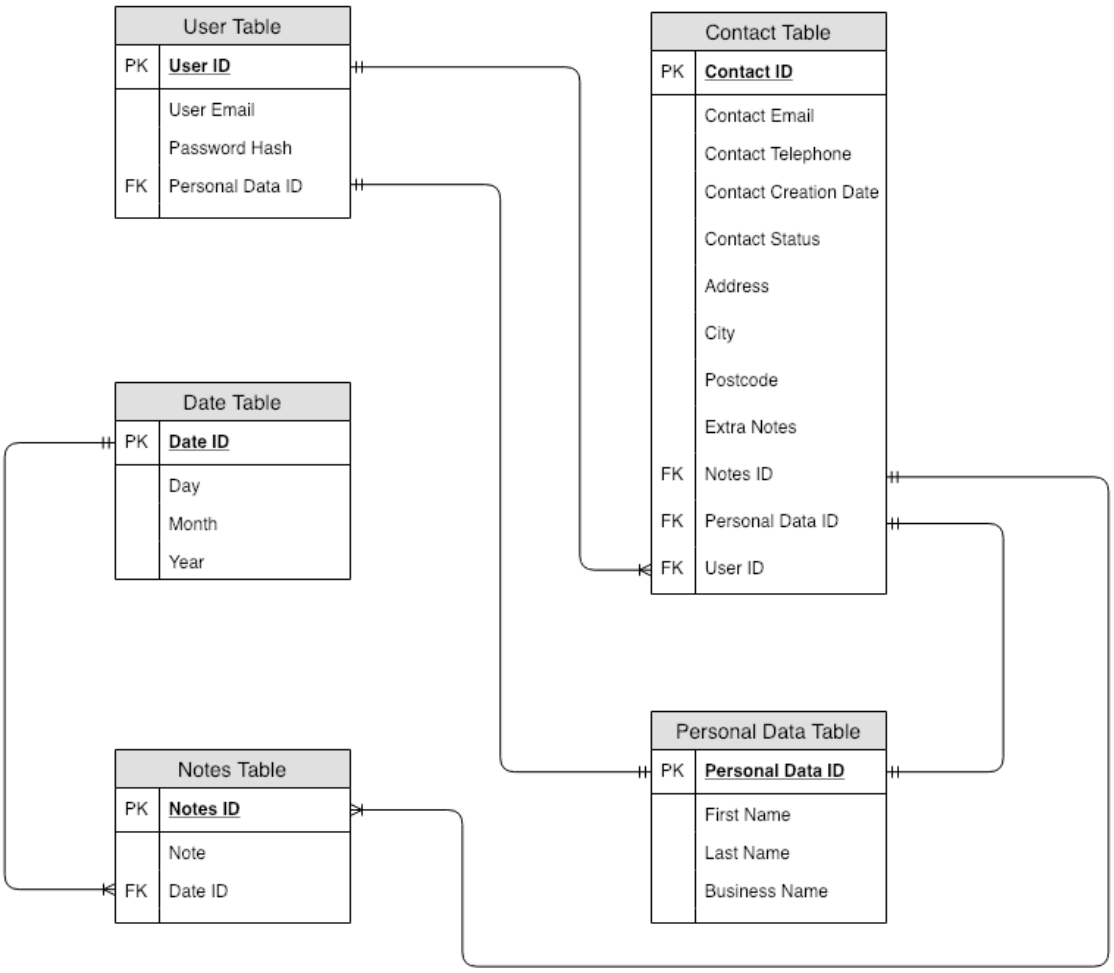
At the bottom of the form will be a submit button (input)

Data Handling

Entity Relationship Diagram






User Table
PK User ID
User Email
Password Hash
FK Personal Data ID
Contact Table
PK Contact ID
Contact Email
Contact Telephone
Contact Creation Date
Contact Status
Address
City
Postcode
Extra Notes
FK Notes ID
FK Personal Data ID
FK User ID
Date Table
PK Date ID
Day
Month
Year
Notes Table
PK Notes ID
Note
FK Date ID
Personal Data Table
PK Personal Data ID
First Name
Last Name
Business Name

*** = Primary key**

// = Foreign Key

Users Table

Table Description: This Table will contain data on the User. Will be created when an account is made. Will be accessed when the user is logging in or updating account details.

Expected Number Of Users: The number of User's is likely to increase over time as people tend not to delete their accounts.

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as it is possible user's will be deleting accounts and so the id's would then be in a random order so random access is the most suitable.

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
User ID *	The unique identifying field for users. Will be used to identify one user from another	Integer	10	1
Email	Will store the user's email address. Used for logging in and sending the user emails	String	30	kieran@kmwcrm.com
Password Hash	Will store the calculation that is achieved when the users's password has the hash algorithm preformed on it. Used for logging in	String	100	qiyh4XPJGsOZ2MEAyLkfWqeQ
Personal Data ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	453

Contacts Table

Table Description: This Table will contain data on a Contact. Will be created when a contact is made. Will be accessed when the user is: viewing contacts, viewing a contact, editing a contact, deleting a contact, .

Expected Number Of Contacts: The number of contacts is likely to increase over time as people as the user will be adding contacts and are likely to be adding more contacts than deleting them.

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as it is possible user's will be deleting contacts and so the primary key fields would be out of order

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Contact's ID *	The unique identifying field. Will be used to identify one contact from another	Integer	10	7896
Email	Will store the contact's email address.	String	30	kieran@kmwcrm.com
Telephone	Will store the contact's telephone number.	String	11	01792947726
Contact Creation Date	Will store the date that the contact was created.	String	70	Mon Jan 21 2019 13:44:15 GMT+0000 (Greenwich Mean Time)
Contact Status	Will store the contact's status.	String	30	Recontact lead
Address	Will store the contact's Address details.	String	60	32 James Street, Pontardawe
City	Will store the contact's City.	String	50	Swansea
Postcode	Will store the contact's Postcode.	String	15	SA8 5AW
Extra Notes	Will store the contact's Extra notes can be in plain text or as an encrypted string.	String	10,000	1f-0a-1b-1b-00-57-03-07-12-05-0a
Notes ID //	The unique identifying field for Notes (Foreign Key). Will store user's notes	Integer	10	453
Personal Data ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	453
User ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	453

Scheduling Data Table

Table Description: This Table will contain data on a scheduling created by the user. Will be created when a user adds a scheduling. Will be accessed when the user is: viewing what they have scheduled (open the dashboard page), delete something scheduled, when a scheduling is deleted because the time and date have passed.

Expected Number Of Scheduling's: The number of Scheduling's is likely to increase over time as it is dependent on the number of users.

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as it is possible user's will be deleting scheduling's and accounts, so the primary key fields would be out of order.

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Scheduled ID	The unique identifying field. Will be used to identify one scheduling from another	Integer	10	8
Scheduled Date	Will hold the scheduling's date	String	10	21/10/2019
Scheduled Time	Will hold the scheduling's time	String	5	18:30
Subject	Will hold the scheduling's Subject Line	String	40	Contact John to get decision
Message	Will hold the scheduling's message/ further details about the scheduling	String	500	Find out if he wants to go ahead with the services. He was apprehensive if he need it. Go with the slow sale, go over case studies again.
User ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	3242

Personal Data Table

Table Description: This Table will contain further data on contacts and users. Will be created when a contact or a user is made. Will be accessed when the user is: updating account details, viewing contacts, viewing a contact, editing a contact, deleting a contact .

Expected Number Of Personal Data tables: The number of Personal Data is likely to increase over time as it is dependent on the number of users and the number of contacts

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as it is possible user's will be deleting contacts and accounts, so the primary key fields would be out of order

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Personal Data ID	The unique identifying field. Will be used to identify one set of personal data from another	Integer	10	67985
Business Name	Will store the contact's business name	String	30	Jimmy's Marketing
First Name	Will store the contact's first name	String	30	Jimmy
Last Name	Will store the contact's last name	String	30	Drool

Clients Lost Table

Table Description: This Table will contain all records of client losses. Will be accessed when the user is: adding a client loss and every time the analytics page is open to update the data points on the graph.

Expected Number Of Client lost tables: The number of clients lost is likely to increase over time as it is dependent on the number of users and it is unlikely the user will be resetting the data.

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as there is likely to be many records of client losses that will build up and so retrieving data will be quicker with random access compared to sequential access.

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Client Lost ID *	The unique identifying field. Will be used to identify one client loss from another	Integer	10	245522
Date ID //	The unique identifying field for a Date (Foreign Key). Will be used to store data on the date a client was lost	Integer	10	487574
User ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	3242

Clients Gained Table

Table Description: This Table will contain all records of client gains. Will be accessed when the user is: adding a client gain and every time the analytics page is open to update the data points on the graph as well as conversion rate calculations.

Expected Number Of Client Gained tables: The number of clients lost is likely to increase over time as it is dependent on the number of users and it is unlikely the user will be resetting the data.

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as there is likely to be many records of client gains that will build up and so retrieving data will be quicker with random access compared to sequential access.

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Client Gained ID *	The unique identifying field. Will be used to identify one client gain from another	Integer	10	8457535
Date ID //	The unique identifying field for a Date (Foreign Key). Will be used to store data on the date a client was lost	Integer	10	487574
User ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	3242

Lead Lost Table

Table Description: This Table will contain all records of lead losses. Will be accessed when the user is: adding a lead loss and every time the analytics page is open to update the data points on the graph as well as conversion rate calculations.

Expected Number Of Lead Loss tables: The number of lead losses lost is likely to increase rapidly over time as it is dependent on the number of users and it is unlikely the user will be resetting the data.

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as there is likely to be many records of lead losses that will build up and so retrieving data will be quicker with random access compared to sequential access.

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Lead Lost ID *	The unique identifying field. Will be used to identify one lead loss from another	Integer	10	245522
Date ID //	The unique identifying field for a Date (Foreign Key). Will be used to store data on the date a client was lost	Integer	10	487574
User ID //	The unique identifying field for Personal Data (Foreign Key). Will store further data on the user	Integer	10	3242

Notes Table

Table Description: This Table will contain all notes. Will be accessed when the user is: adding a note to a contact or when a contact's page is opened to retrieve the notes relating to that contact.

Expected Number Of Notes tables: The number of notes is likely to increase rapidly over time as it is dependent on the number of users and it is unlikely the user will be deleting contacts (also removing notes related to that contact).

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as there is likely to be many records of notes that will build up and so retrieving data will be quicker with random access compared to sequential access.

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Note ID *	The unique identifying field. Will be used to identify one lead loss from another	Integer	10	245522
Note	Will store the note the user adds	String	500	Bob wants a new logo created with red stripes and his brand name "bob stripes"
Date ID //	The unique identifying field for a Date (Foreign Key). Will be used to store data on the date a client was lost	Integer	10	487574

Date Table

Table Description: This Table will contain all notes. Will be accessed when the user is: adding a note to a contact or when a contact's page is opened to retrieve the notes relating to that contact.

Expected Number Of Notes tables: The number of notes is likely to increase rapidly over time as it is dependent on the number of users and it is unlikely the user will be deleting contacts (also removing notes related to that contact).

Access Method: For this table, data will be stored and retrieved via Direct/ Random Access. This will be done as there is likely to be many records of notes that will build up and so retrieving data will be quicker with random access compared to sequential access.

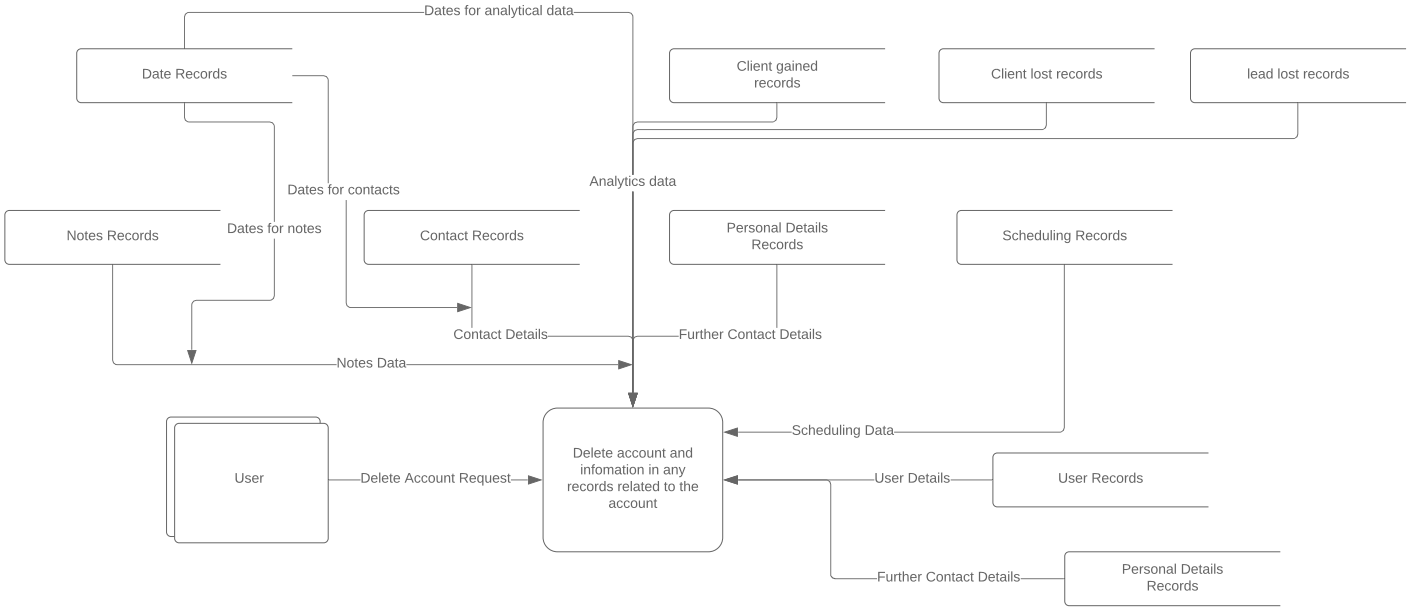
Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Date ID *	The unique identifying field. Will be used to identify one date from another	Integer	10	875585
Day	Will store the date's day	Integer	2	26
Month	Will store the date's month	Integer	2	2
Year	Will store the date's year	Integer	4	2019

Processes

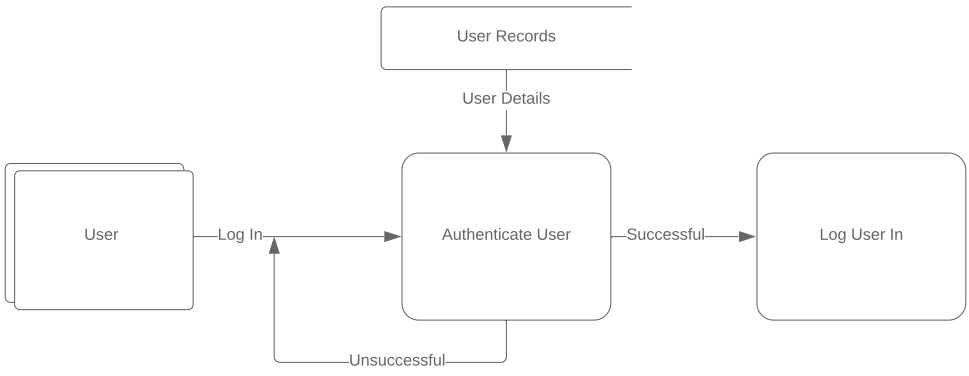
Processes On Data - Data Flow Diagrams

Users Data Flow Diagrams

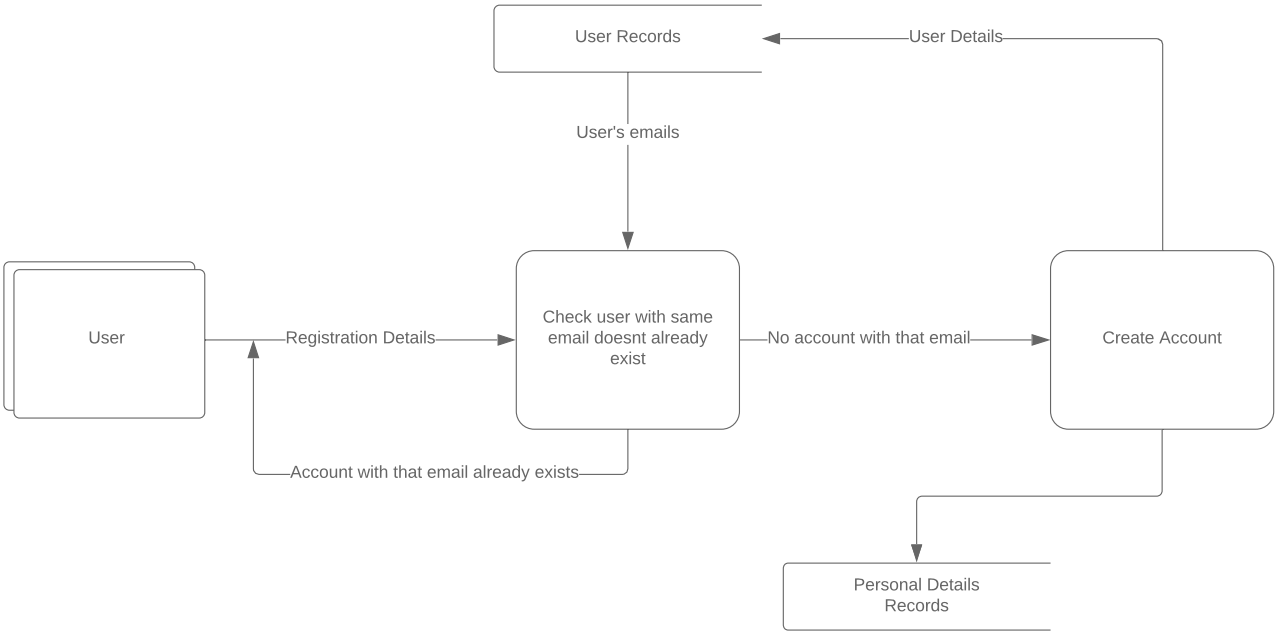
Delete Account



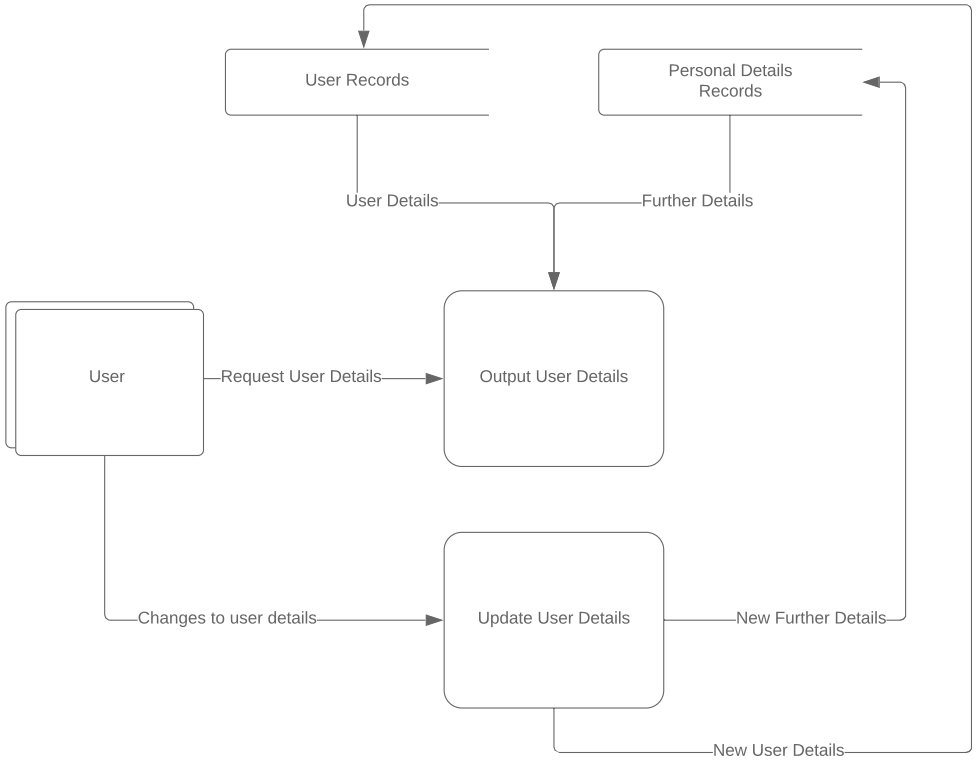
Log In To Account



Register An Account

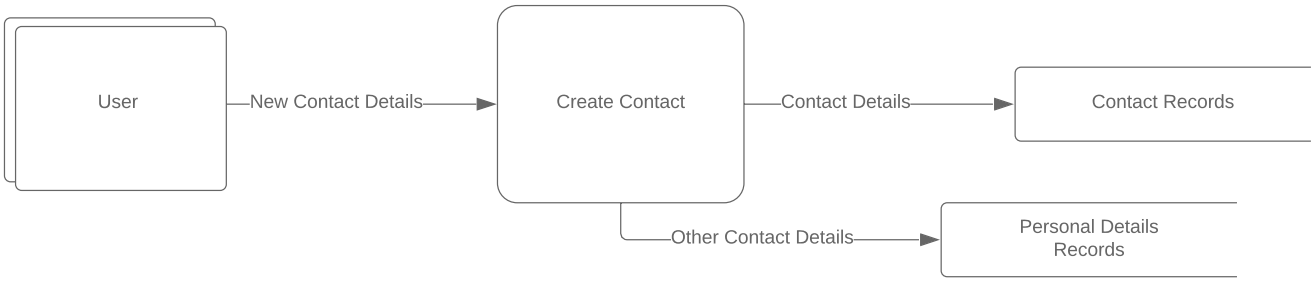


Update Account Details

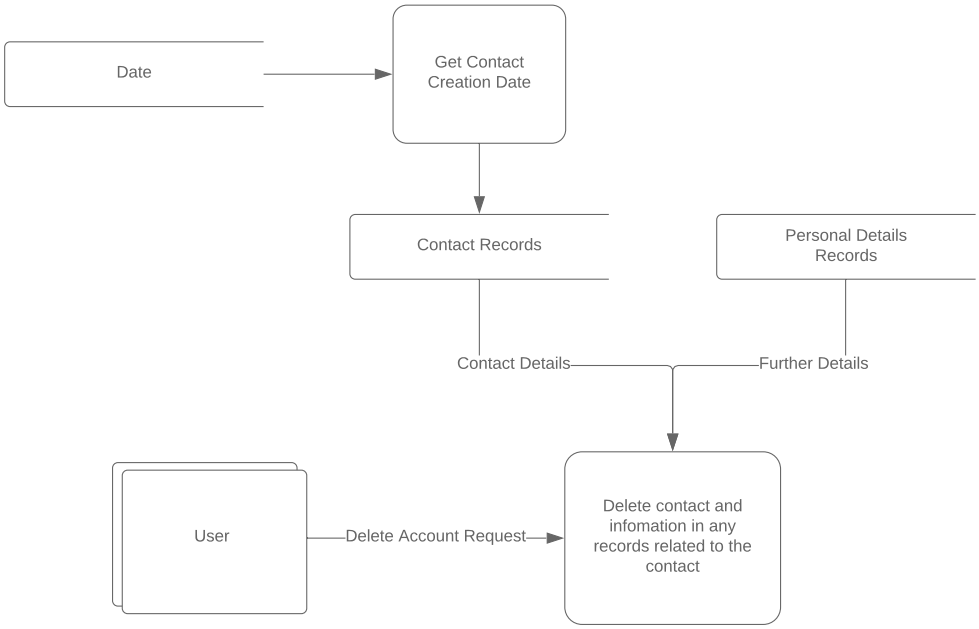


Contacts Data Flow Diagrams

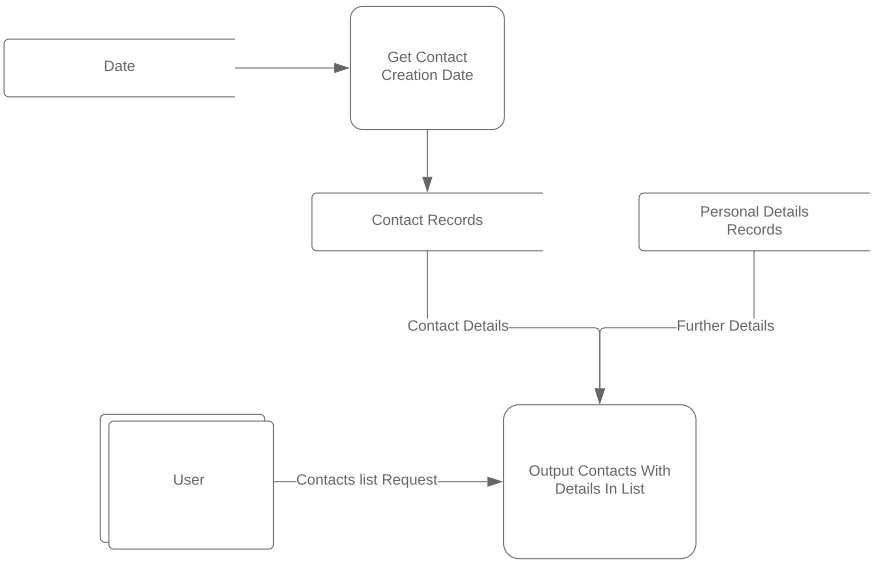
Create Contact



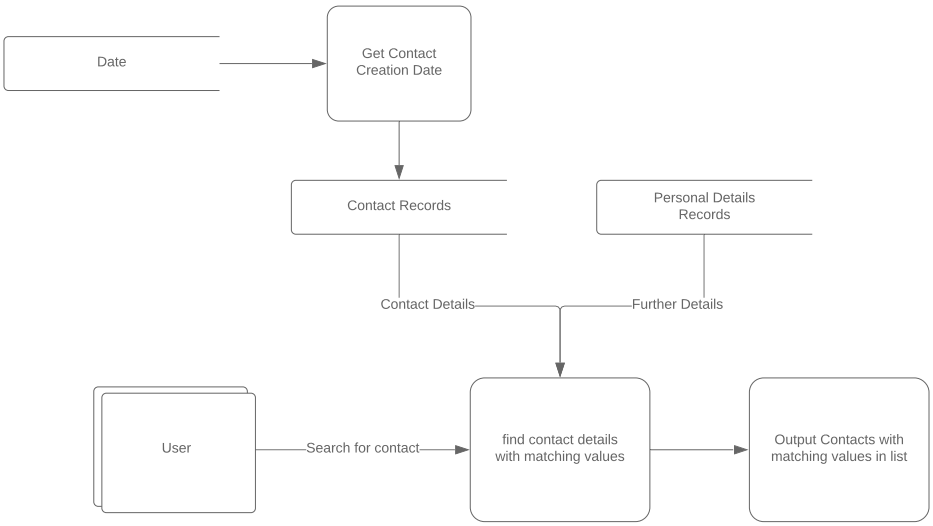
Delete Contact



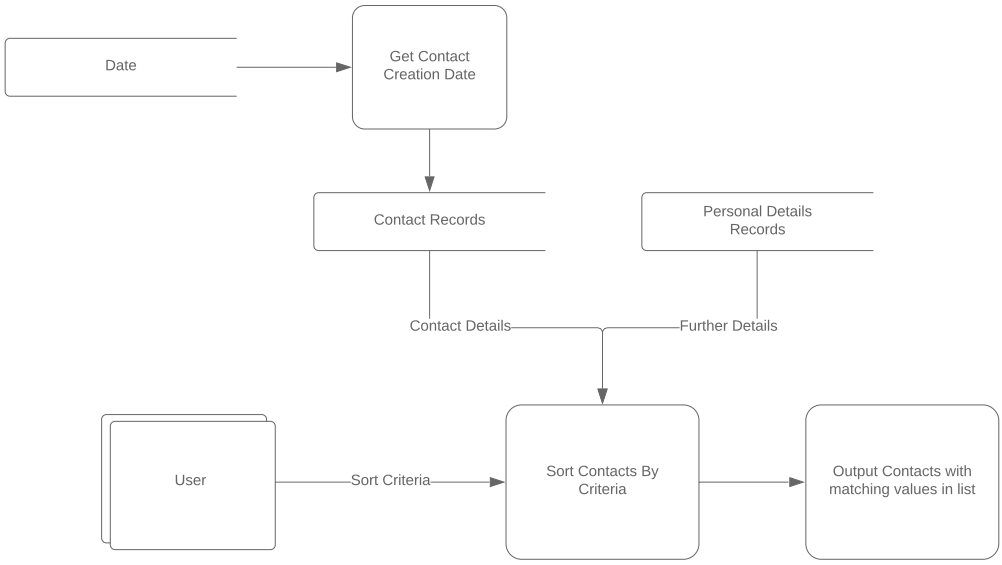
Output Contact Details To List



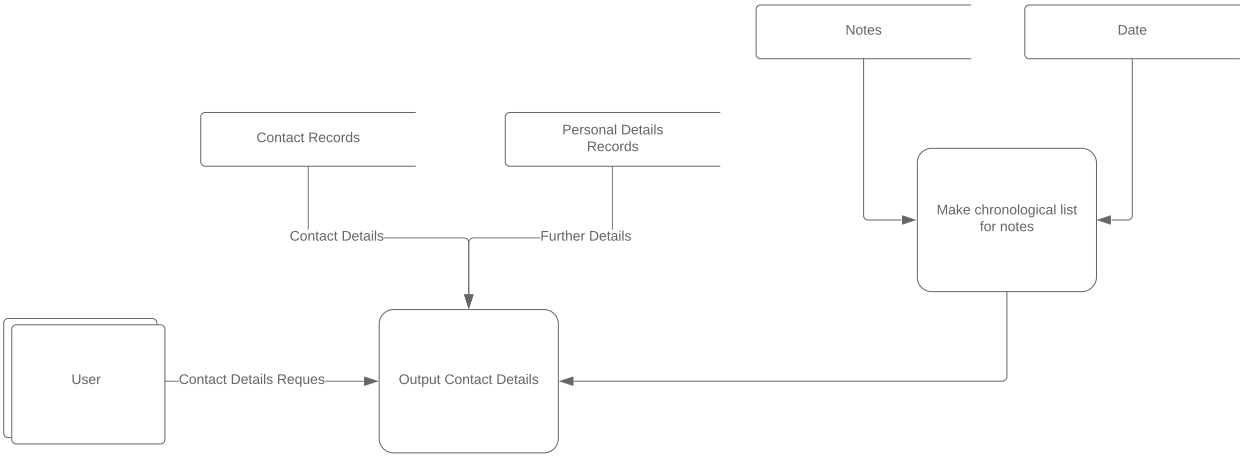
Output Contact Details With Search List



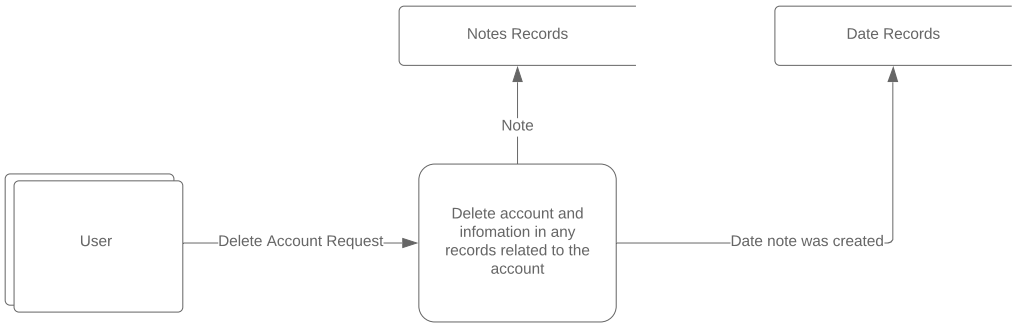
Output Contact Details With Sort List



View Individual Contact Details

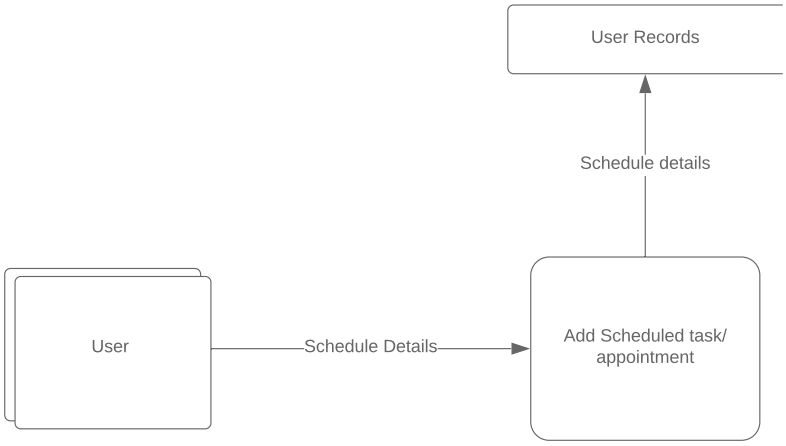


Add Note In A Contact's Page

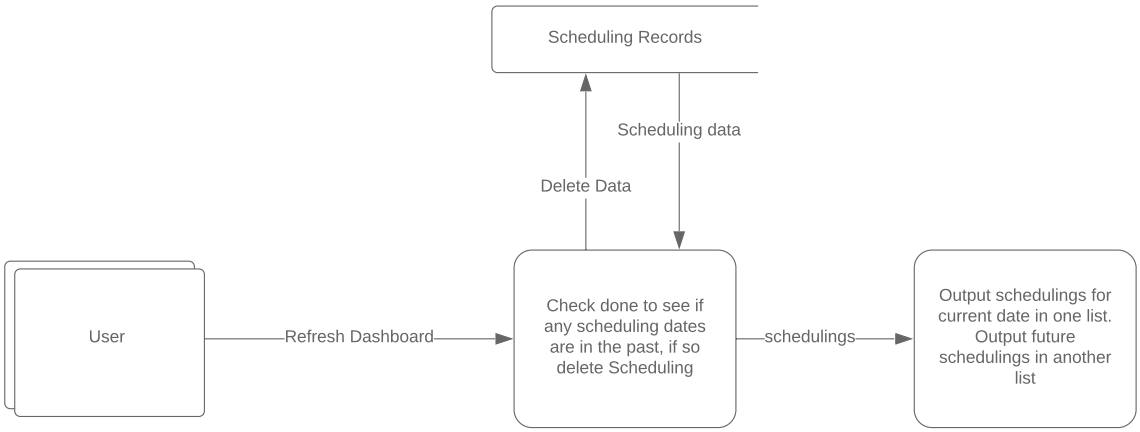


Scheduled Data Flow Diagrams

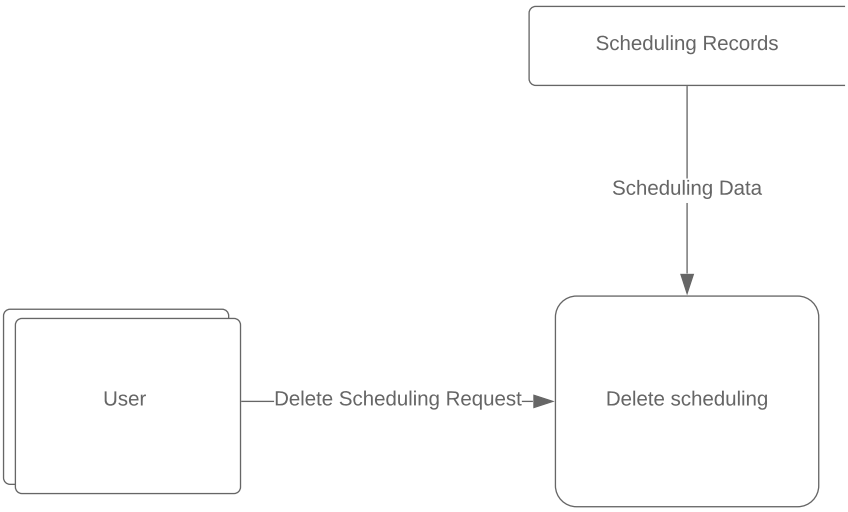
Add Scheduling



View Scheduling's

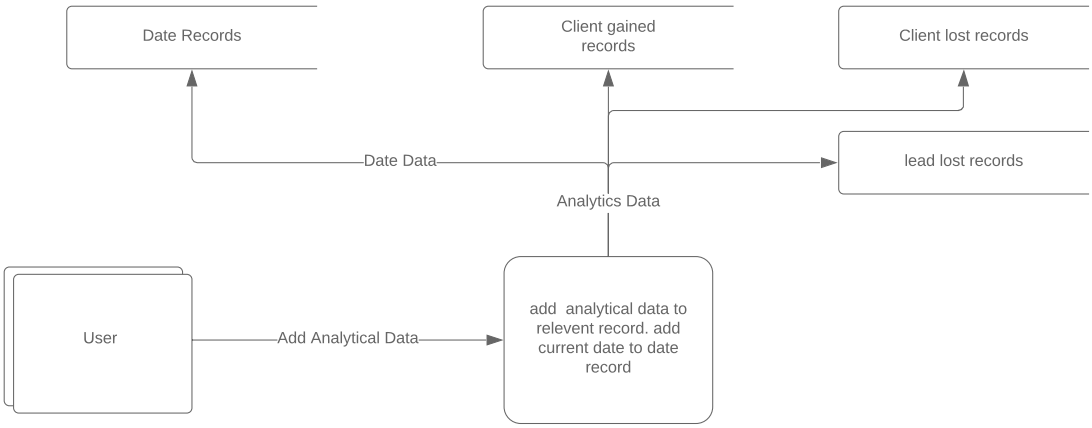


Delete Scheduling

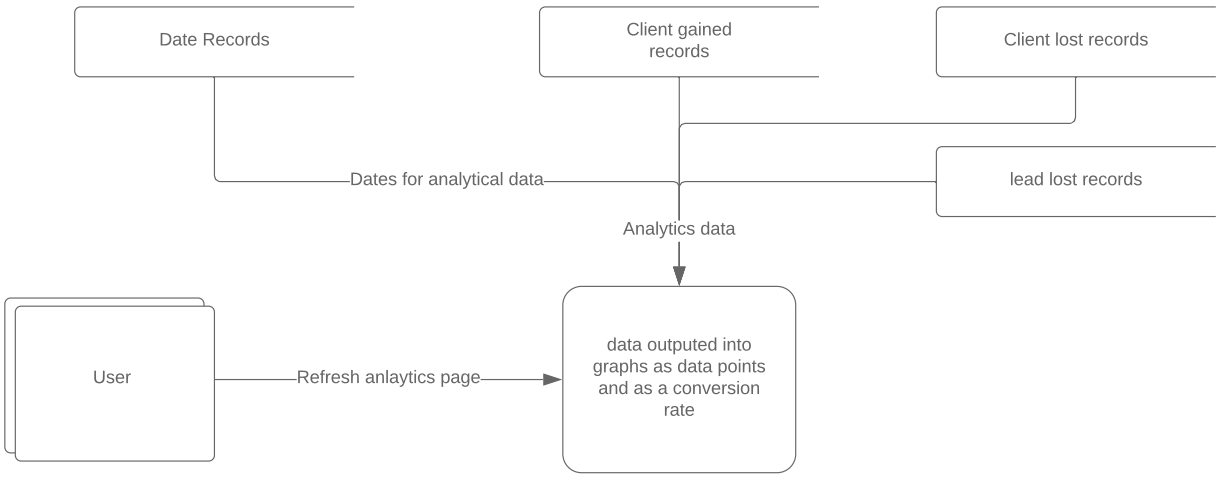


Analytics Data Flow Diagrams

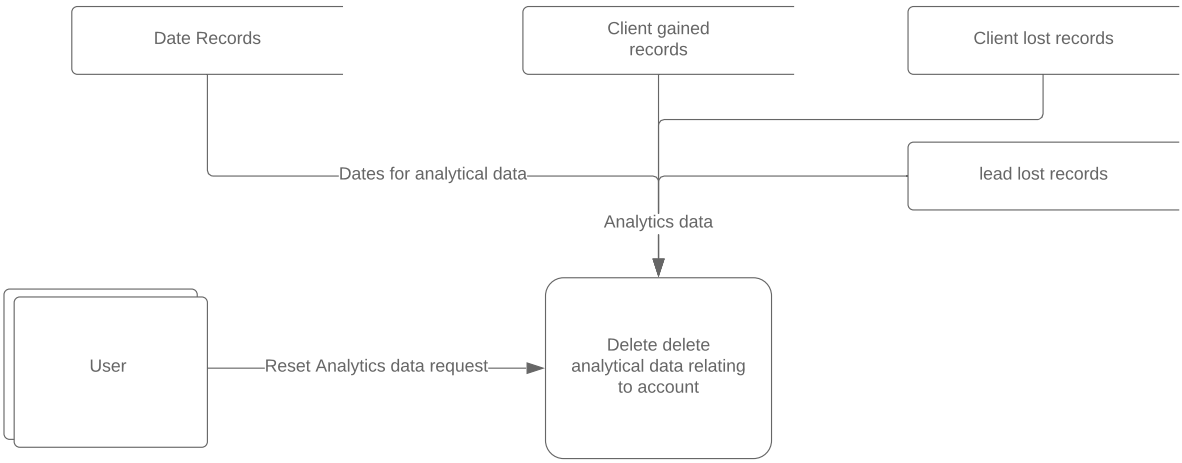
Add Analytical Data



View Analytical Data



Delete Analytical Data



Algorithms

Comments are denoted like this <// comments //> used where the code might seem a bit convoluted but many of the algorithms should be quite easy to follow and have a suitable sub-procedure name that will define what the following pseudocode after it does

XOR Encryption - will be used to encrypt and decrypt a notes section in the web app

```
generateKey (input: password, input: textLength) <// will be used get the key the same length as the text input />
        array: passwordArray = []
        integer: difference = 0
        while length(password) <= textLength
                password = password + password
        passwordArray = split( password by ‘’ )
        difference = length ( passwordArray ) - textLength
        For i = 0 to difference - 1
                removeLastLastArrayElement (passwordArray)
        Return passwordArray
```

```
Encrypt (input: text, input: password)
        array: inputArr = split( text by ‘’ )
        array: xorValues = []
        array: keyArray = generateKey (password, length(text))
        for i = 0 to length(keyArray) - 1
                keyArray[i] = binaryValue(keyArray[i]).toString
                inputArray[i] = binaryValue(inputArray[i]).toString
                xorValues[i] = keyArray[i]  XOR  inputArray[i]
        return xorValues.toString  +  “,”  + binaryValue (length ( text )) <// comma separated  last value stores the length of dec//>


Decrypt ( input: text, input: password)
        array: inputArray = split ( text by ‘,’)
        array: keyArray = []
        array: xorValues = []
        integer: DECRYPTED_TEXT_LENGTH = binaryValueToDenary( array[ length ( array ) ] )
        array: keyArray = generateKey ( password )
        for i = 0 to length ( keyArray ) - 1
                keyArray[i] = binaryValue(keyArray[i]).toString
                xorValues[i] = binaryToUTF(keyArray[i]  XOR  inputArray[i])
        return xorValues.toString <// joined by ‘’ //>
```

Array Manipulation - will be used to manipulate an array so that data can be added or removed easily and the length of an array can be found which can be used to find the length of a string

```
removeLastElementFromArray ( input: array)
        integer: arrayLength = length ( array )
        array: newArray = []
        for i = 0 to arrayLength - 2
                newArray[i] = array[i]
        return newArray


removeElementFromIndex ( input: array, input: index)
        array: newArray = []
        for i = 0 to index - 1
                newArray[i] = array[i]
        for x = index + 1 to length ( array ) - 1
                newArray[x-1] = array[x]
        return newArray


Length ( array )
        integer: length = 0
        while true
                try
                        array[a] + 1
                        length = length + 1
                catch ( exception where array index is out of bounds )
                        break loop
        return length


addToArray ( array, value)
        array: newArray = []
```

```
        for i = 0 to length (array)
                newArray[i] = array[i]
        newArray[ (length (array) - 1] = value
        return newArray
```

String Manipulation - will be used in gaining extra data from a string or to present the string in a different way. The capitalise method will be used to capitalise fields such as names before saving them to the database

```
Length ( input: string)
        array: strArray = split (string by ‘’)
        return length (strArray) <// uses the arrays length sub procedure

capitalizeString (input: inputString)
        array: words = split(inputString by “ ”)  <// splits by a space character />
        array: currentWordsCharacters = []
        array: lowerCaseArray = split (“abcdefghijklmnopqrstuvwxyz” by ‘’)
        array: upperCaseArray = = split (“ABCDEFGHIJKLMNOPQRSTUVWXYZ” by ‘’)
         for  i = 0 to length (inputString)
                currentWordsCharacters = split( words[i] by “” )
                for x = 0 to 25
                        if lowerCaseArray[x] == currentWordsCharacters[0]
                                currentWordsCharacters[0] = upperCaseArray[x]
                                break out of inside loop
                 currentWordsCharacters = currentWordsCharacters.join(“”)
                words[i] = currentWordsCharacters
    words = words.join(" ")
    return words
```

Sorting - bubbleSort can be used to sort data in a one dimensional array so that data can found easier. The bubble sort 2d method can be used to sort the records by any column

```
bubbleSort ( input: array )
        integer: n = length (array)
        boolean: sorted = false
        integer: temp = 0
        while sorted == false
                sorted = true
                for i = 0 to n
                        if array[i] > array[i+1]
                                temp = array[i]
                                array[i] = array[i+1]
                                array[i+1] = temp
                                sorted = false
        return array
```

```
bubbleSort2D (input: array2D, input: columnToSortBy)
        integer: n = length (array2D)
        boolean: sorted = false
        array: temp = []
        while sorted == false
                sorted = true
                for i = 0 to n - 1
                        if array[i][columnToSortBy] > array[i+1][columnToSortBy]
                                temp = array2D[i]
                                array2D[i] = array2D[i+1]
                                array2D[i+1] = temp
                                sorted = false
        return array2D
```

Searching - will be used to find anything in a one dimensional array, could be utilised in finding values from a partial search value

```
linearSearch (input: array, input: searchValue)
        boolean: found = false
        integer: i = 0
        integer: length =  length ( array )
        while (found == false) OR (i < length)
                if array[i] == searchValue
                        found = true
                        return i
                else
                        i = i + 1
        return -1
```

```
binarySearch (input: array, input: searchValue)
        array: array = bubbleSort(array)
        integer: first = 0
        integer: last = length ( array )
        integer: middle = truncate ((first + last)/2)
        boolean: found = false
        while (found == false) OR (first <= last)
                if array[middle] == searchValue
                        found = true
                        return middle
                if searchValue < array[middle]
                        last = middle - 1
                        middle = Math.floor((first + last)/2)
                if(searchValue > array[middle]){
                        first = middle + 1;
                        middle = truncate ((first + last)/2)
        return -1
```

Binary Search On 2D Array, on multiple columns, return all matches

- will be used to find matches when the user searches for something in the search bar

```
binarySearch2D (input: array, input: columnsToBeSearched, input: searchValue)
        array: recordsFoundInAllColumnsSearched = []
        array: recordsFoundInOneColumn = []
        integer: col = 0
        for i = 0 to length ( columnsToBeSearched ) - 1
                col = columnsToBeSearched[i]
                array = bubbleSort2D(array, col)
                recordsFoundInOneColumn = binarySearch2DFinder(array, col, searchValue)
                if length ( recordsFoundInOneColumn ) > 0
                        for i = 0 to length ( recordsFoundInOneColumn ) - 1
                                recordsFoundInAllColumnsSearched[i] = recordsFoundInOneColumn[i]
        return recordsFoundInAllColumnsSearched
```

```
binarySearch2DFinder (input: array, input: col, input: searchValue)
        integer: first = 0
        integer: last = length ( array )
        integer: middle = truncate ((first + last)/2)
        integer: upperAndLowerBoundMatches = []
        array: recordsFound = []

        while first <= last
                if array[middle][col] == searchValue) <//case where the search value is found//>
                        recordsFound = addToArray ( recordsFound, middle )
                        <// checks for more matches at the bounds//>
                        upperAndLowerBoundMatches = checkBoundsForMatches(array, middle, col, searchValue)
                        <//puts the bounds matches into the recordsFound array//>
                        for i = 0 to length ( upperAndLowerBoundMatches )
                                recordsFound = addToArray ( recordsFound, upperAndLowerBoundMatches)
                        break out of loop
                if searchValue < array[middle][col]
                        last = middle - 1
                        middle = truncate ((first + last)/2)
                if(searchValue > array[middle][col])
                        first = middle + 1
                        middle = truncate ((first + last)/2)
        return
```

```
checkBoundsForMatches (input: array, input: middle, input: col, input: searchValue)
        boolean: matchAtBoundsFoundFlag = true
        array: recordsFound = []
        integer: upperBounds = middle
        integer: lowerBounds = middle
        while matchAtBoundsFoundFlag == true <// loops through the code as long as a match has been found //>
                matchAtBoundsFoundFlag = false
                if ((upperBounds + 1) > (array.length - 1))
                        break out of loop
                if (array[upperBounds + 1][col] == searchValue)
                        matchAtBoundsFoundFlag = true
                        upperBounds = upperBounds + 1
                        recordsFound = addToArray( recordsFound, upperBounds )

        matchAtBoundsFoundFlag = true <// resets the flag back to true to check the other sides bounds //>

        while matchAtBoundsFoundFlag == true
                matchAtBoundsFoundFlag = false
        if (lowerBounds - 1) < 0
                break out of loop
        if array[lowerBounds - 1][col] == searchValue
                matchAtBoundsFoundFlag = true
                lowerBounds = lowerBounds - 1
                recordsFound = addToArray ( recordsFound , lowerBounds )
        return recordsFound
```

Conversion Rate - will be used to calculate the conversion rate for the month on the analytics page

```
conversionRate ( input: numOfClientsGained, input: numOfLeadsLost )
        string: conversionRateRatio = getConversionRateRatio (numOfClientsGained, numOfLeadsLost)
        string: conversionRatePercentage= getConversionRatePercentage (numOfClientsGained, numOfLeadsLost)
        return [conversionRateRatio, conversionRatePercentage]



getConversionRateRatioAsString (input: numOfClientsGained, input: numberOfLeadsLost)
        integer: gcd = greatestCommonDivisor(numberOfClientsGained, numberOfLeadsLost)
        return (numOfClientLost/gcd).toString() + ':' + (numOfLeadsLost/gcd).toString()



greatestCommonDivisor( input: valueOne, input: valueTwo)
        integer: remainder = 0
        if ( valueTwo == 0 )
                return valueOne
        remainder = valueOne MOD valueTwo
        greatestCommonDivisor (valueTwo, remainder)

getConversionRateAsPercentage (input: numOfClientsGained, input: numOfLeadsLost)
        integer: total = numOfClientsGained + numOfLeadsLost
        integer: percentage = truncate (( numOfClientsGained / total) * 100)
        return percentage.toString + "%"
```

Validation - used in the input forms to ensure data entered is valid

Type Checks (will use the language's syntax and standard libraries to check the types) :

number, integer, real, string, character, boolean, and other types that are native to the languages I will use

LengthCheck (input: value, input: length)

return length (value) == length

numberRange (input: value, input: lower, input: upper)

return value >= lower AND value <= upper

stringRange (input: value, input: lower, input: upper)

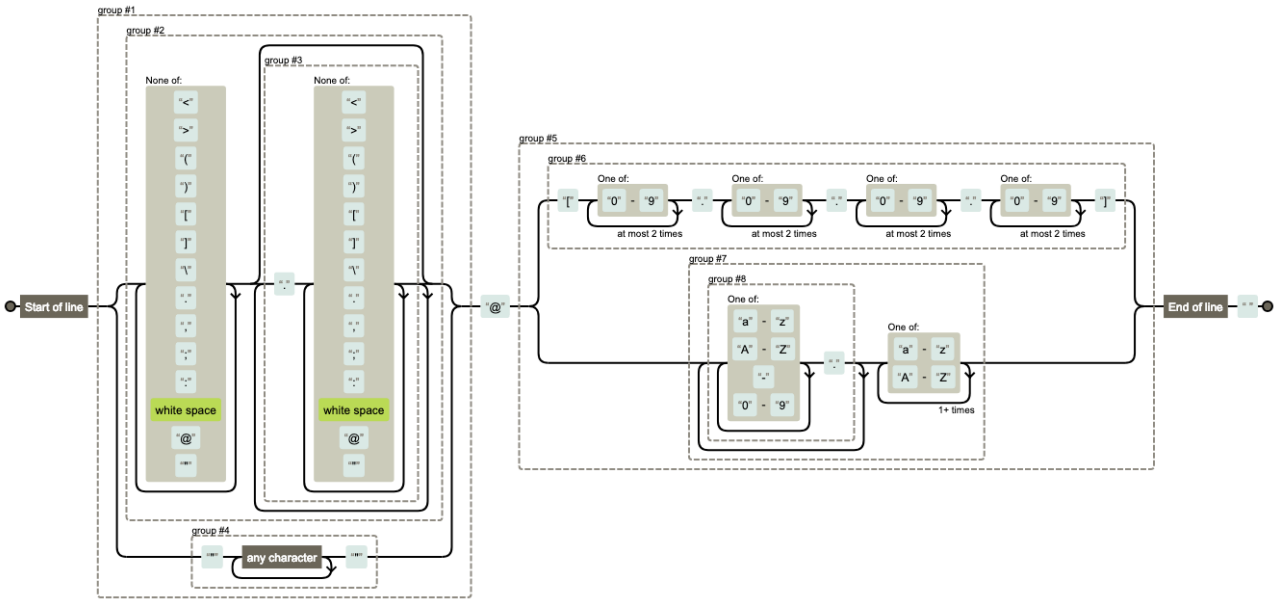
value = length (value)

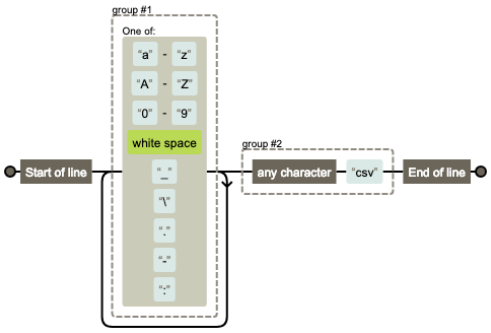
return value >= lower AND value <= upper

emailFormatCheck (input: value) <// found an accurate regex for an email online //>

regex: emailFormat = ^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$

return value matches emailFormat






```
csvFormatCheck ( input: value)
        regex: csvFileNameFormat = ^([a-zA-Z0-9\s_\\.\-:])+(.csv)$
        return value matches csvFileNameFormat
```

```
timeFormat ( input: value)
        array: valueArray = []
        integer: val = 0
        string: hour
        string: minute
        <// checks that it is a string and that it is also of length 5 //>
        if typeString(value) == true AND lengthCheck(value, 5) == true
                valueArray = value.split("")
                for let i = 0 to length ( valueArray )
                        val = valueArray[i]
                        if i == 2
                                if val NOT== ":"
                                        return false
                        else
                                if Validation.typeNumber(stringToNumber(val) == true
                                        return false
                        hour = valueArray[0] + valueArray[1]
                        minute = valueArray[3] + valueArray[4]

                         return Number(hour) >= 0 AND Number(hour) <= 23 AND Number(minute) >= 0  AND Number(minute) <= 59
        else
                return false
```

```
comparissonCheckVerification (input: valueOne, input:  valueTwo)
        return valueOne == valueTwo


lookupCheckForLowerCase(input: stringValue)
        boolean: lowerCaseFlag = false
        array: lowerCaseArray = "abcdefghijklmnopqrstuvwxyz".split("")
        array: stringArray = stringValue.split("")
        character: currentCharacter = “”
        for i = 0 to length ( stringArray )
                currentCharacter = stringArray[i]
                for j = 0 to 25
                        if currentCharacter == lowerCaseArray[j]
                                lowerCaseFlag = true

        return lowerCaseFlag
```