

System Maintenance

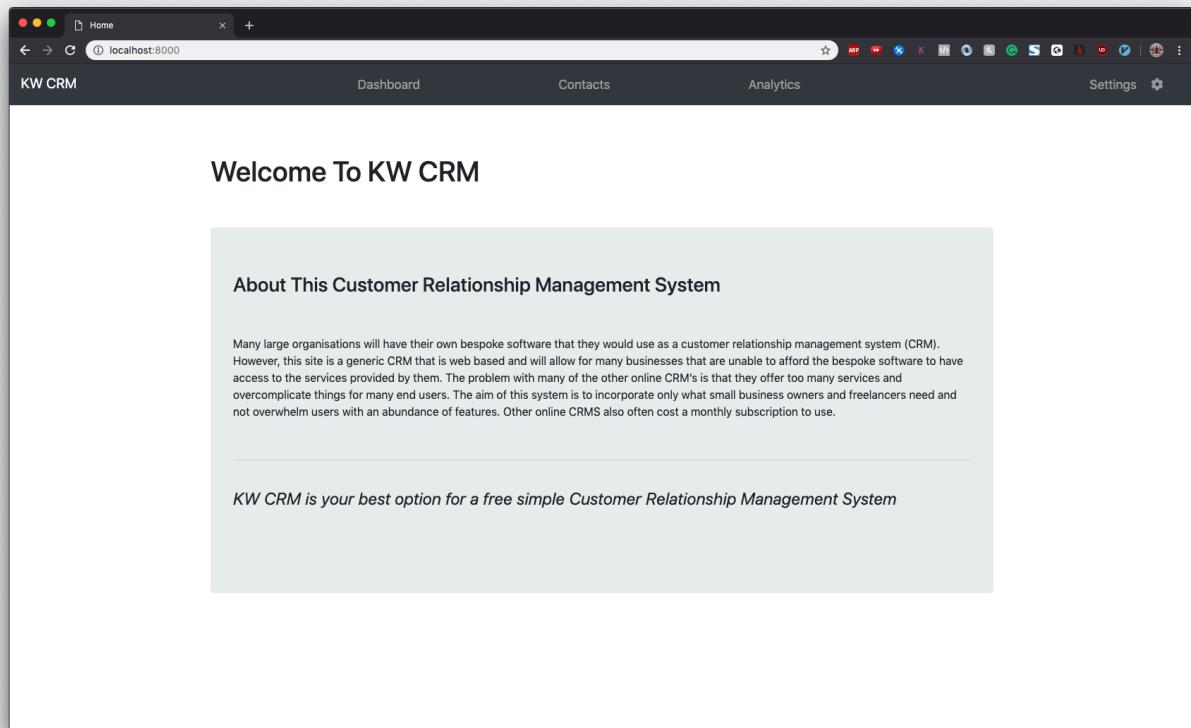
Documentation

Customer Relationship Management System (CRM)

Kieran Williams

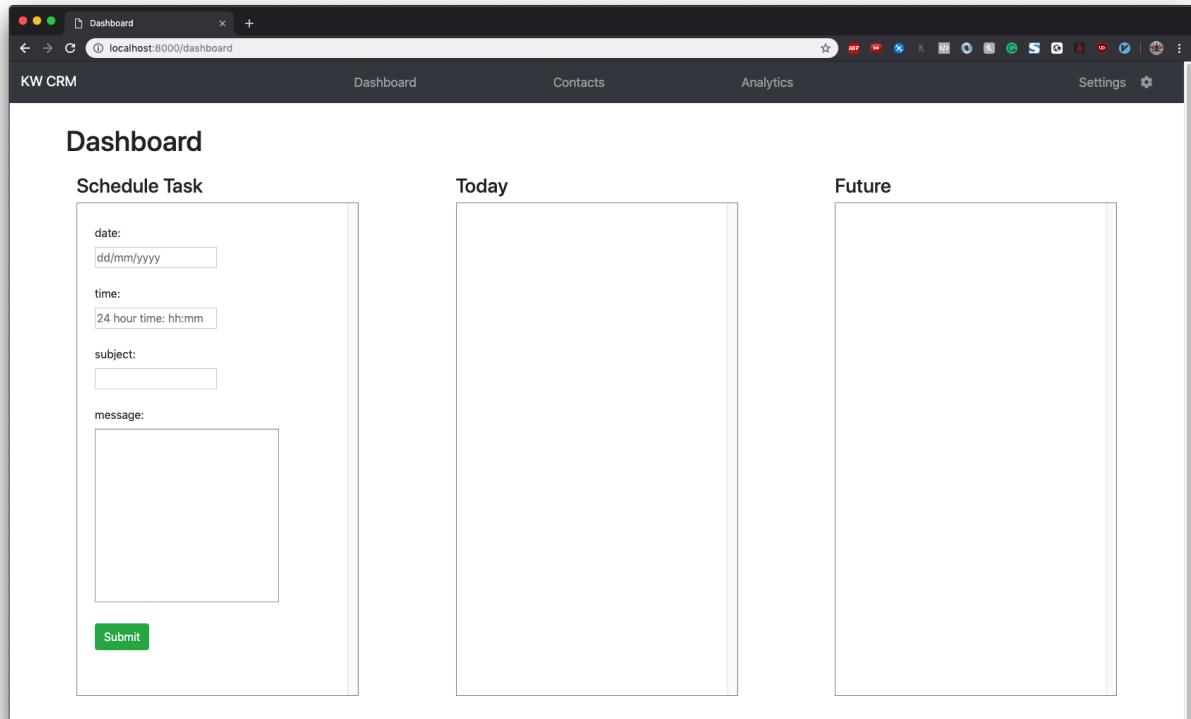
UI And All Validation Messages

Home Page



The home page has no features other than a greeting and description of the system and who it is aimed towards

The Dashboard



The Dashboard has the ability to add, view and delete scheduled tasks

The screenshot shows a web-based CRM application with a dark-themed header. The header includes the logo 'KW CRM', navigation links for 'Dashboard', 'Contacts', 'Analytics', and 'Settings', and a gear icon for settings.

The main area is titled 'Dashboard' and contains three sections:

- Schedule Task**: A form for scheduling a task. It has fields for 'date' (dd/mm/yyyy), 'time' (24 hour time: hh:mm), 'subject', and 'message'. A green 'Submit' button is at the bottom.
- Today**: A list of scheduled tasks for the current day. It includes:
 - 07/04/2019, 10:00, **Birthday**: sallys birthday need to buy her a card before asking if she would like to upgrade the service at a discount
 - 07/04/2019, 19:45, **Meeting With Ajay**: he wants the gold package but wants us to do it for cheaper so prepare a presentation on what he will get for the price to convince him its worth it
 - 07/04/2019, 21:20, **Meeting With James Jones**: to finalise which product design he wants to go ahead with
- Future**: A list of scheduled tasks for future dates. It includes:
 - 08/07/2019, 10:00, **Birthday**: sallys birthday need to buy her a card before asking if she would like to upgrade the service at a discount
 - 28/10/2019, 13:00, **Meeting With Ajay**: he wants the gold package but wants us to do it for cheaper so prepare a presentation on what he will get for the price to convince him its worth it
 - 07/12/2019, 21:00, **Meeting With James Jones**: to finalise which product design he wants to go ahead with

This is a populated dashboard where all the scheduled tasks for both the day and the future are displayed

Please Ensure All Text Fields Have Data Entered _____ Please Ensure The Time Is In The Format hh:mm Use The 24hr Clock _____ Please Ensure The Date Is In The Format dd/mm/yyyy And The Date Is In The Future Or Today's Date _____ Ensure That The Subject Field Is Within Range Of 1 - 40 Characters _____ Ensure That The Message Field Is Within Range Of 1 - 200 Characters _____

Dashboard

These are all the possible validation messages that can get displayed due to invalid data entry in the schedule task form

Successfully Added A Scheduled Task

Dashboard

Schedule Task

Today

This message get's displayed when the user successfully adds a task

Successfully Deleted A Scheduled Task

Dashboard

This message get's displayed when the user successfully deletes a task

Contacts List Page

Name	Business	City	Email	Telephone
Allen Roger	BigMerch	Cardiff	alrog@gmail.com	07496437123
Sally Hughes	Dynasty	Neath	dynastyClothing@gmail.com	07496467098
Paul Smith	Paulsmith Guitars	Swansea	psGuitars@gmail.com	07436890212
Joe Rob	Joe Skate	Port Talbot	joeyskater@gmail.com	01792782635
Jamie Roberts	Big Breakfasts 4 You	London	jamierobs@outlook.com	07496436543
James Brinks	James Media	Swansea	jamesbrinks@hotmail.com	01792946751
Bob Wally	The Network Guys	Swansea	boobywallyT@networkgys.com	01792876784
Rob Sally	Breakfasts U Want	Swansea	robsysally@breakfasts.com	07496436543
Larry Watson	Dig Dig Diggers	Port Talbot	lazzywatz@gmail.com	07456897564

This page displays all the contacts in the data store (to view the contacts individual page, the name of the contact is to be clicked as it is a link to their page). You can search the list with the search bar. You can sort the list in both ascending and descending order. You can go to the add contact page from clicking the add contact button and download all the contact's data by clicking the export contacts button.

The purpose of the page is to display the contacts that are stored to the user and provide ways in which to find or narrow down contacts within that list.

No Matches found

Contacts List

When a search value that has no matches in the contacts datastore is entered, this message is displayed

4 Matches Found

Contacts List - "Searched for: Swansea"

Search

Name	Business Name
Paul Smith	Pauls Restaurant
Dan Sally	Breakfast King

When a search value is entered where there are matches, these messages gets displayed and the contacts list only shows the records that contain the matching search value

These messages get displayed when the user sorts the contacts list and then the contacts list gets sorted in either ascending or descending order in the column that the list is being sorted by

Contacts List - "Sorted By: City"

Search

Name	Business Name
Allen Roger	Eazy Eatzz
Jamie Roberts	Eatery
Sally Hughes	Cafe 28

Contacts List - "Sorted By: Date"

Search

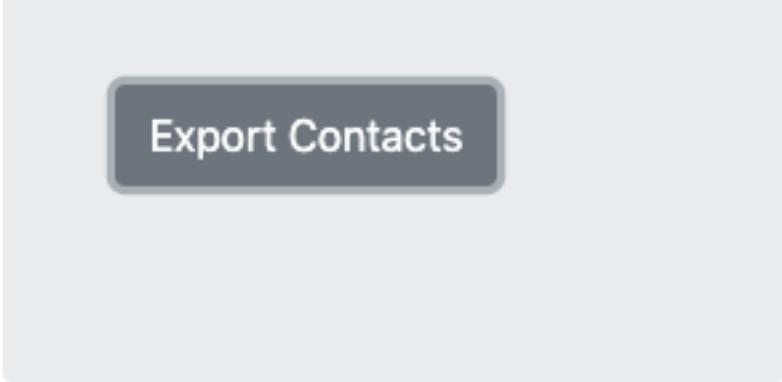
Name
Allen Roge

Contacts List - "Sorted By: Business Name"

Search

Name	Business Name
Jamie Roberts	Big Breakfast

Pressing the export contacts button downloads the contacts data onto the clients computer



Export Contacts



contactRecord....csv



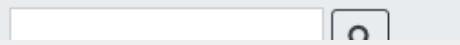
Contact Successfully Edited

Contacts List

When a contact's data get's edited, this message gets displayed after they get redirected back to the contacts list page

Successfully Created A Contact

Contacts List



Search

Name

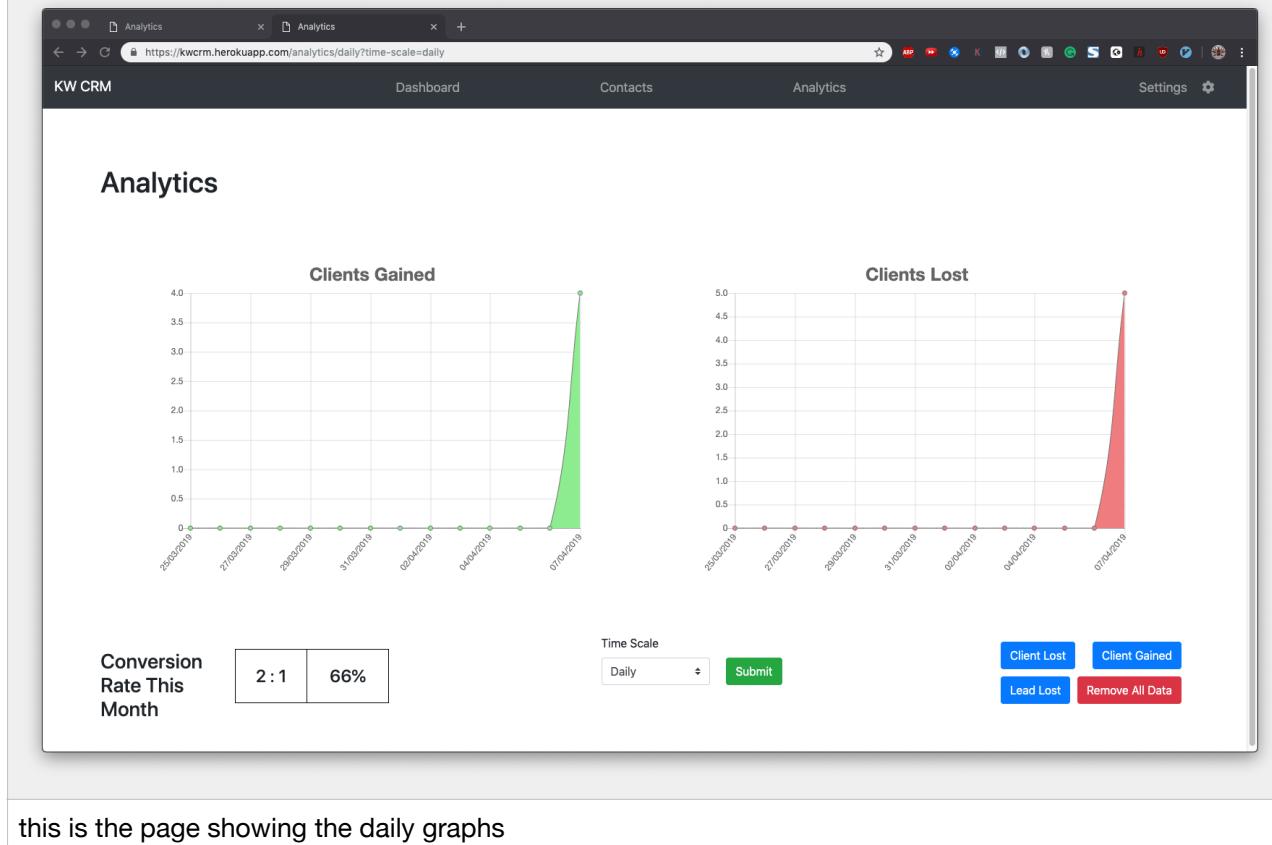
Allen Roger

Sally Hughes

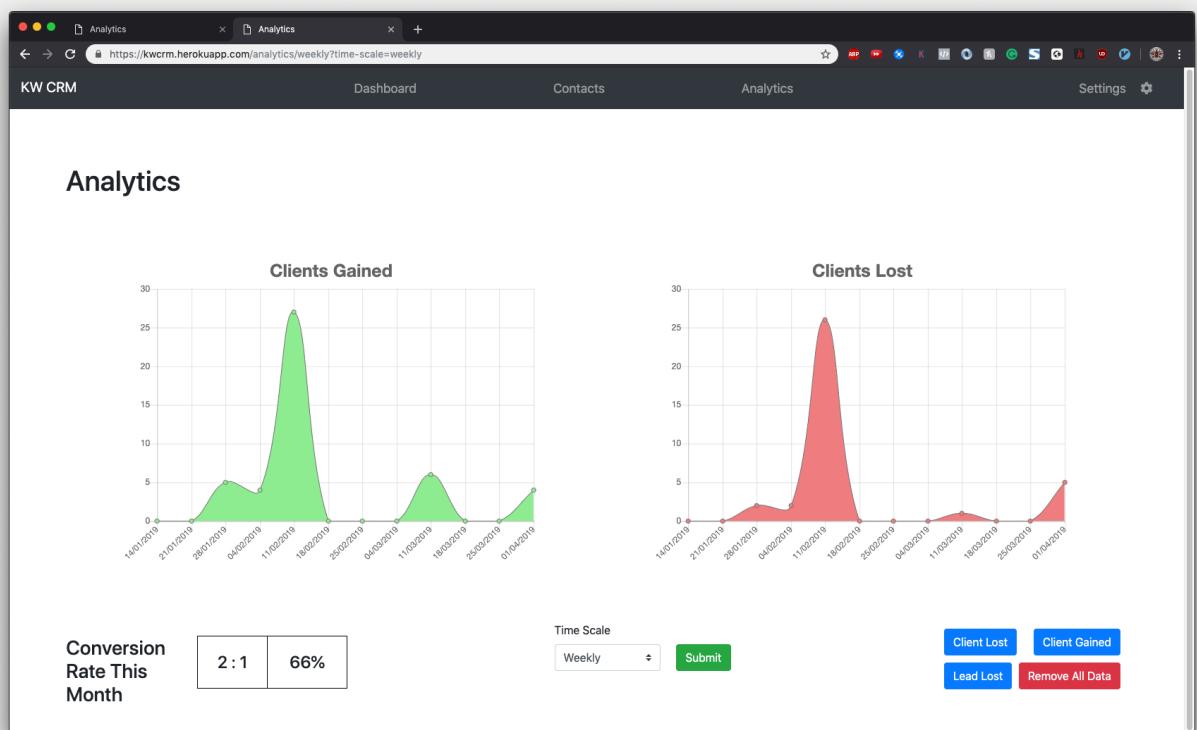
When a contact gets successfully created, this message gets displayed after they get redirected back to the contacts list page

The Analytics Page

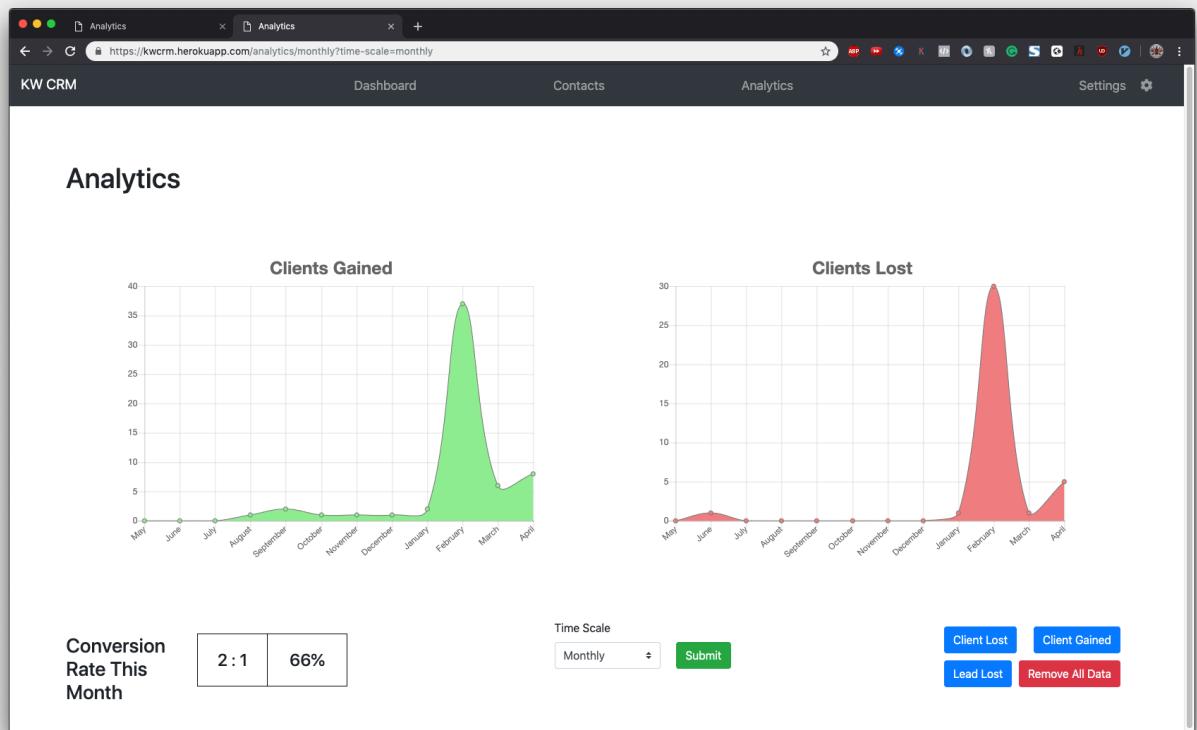
The purpose of this page is to see how the business is doing. It has the input buttons for when a lead is lost, a client is lost and for when a client is gained. When these buttons are pressed, their relevant data stores get a new record appended to the end of the file with the current date. The graphs then display the clients gained and clients lost over a period of time. The period of time that the graphs display data for is determined by the drop down menu that has options for daily (the past 14 days), weekly (the past 12 weeks), and monthly (the past 12 months). There is also the conversion rate for the month which is the clients gained against the leads lost in both a ratio that is in its lowest form and as a percentage.



this is the page showing the daily graphs



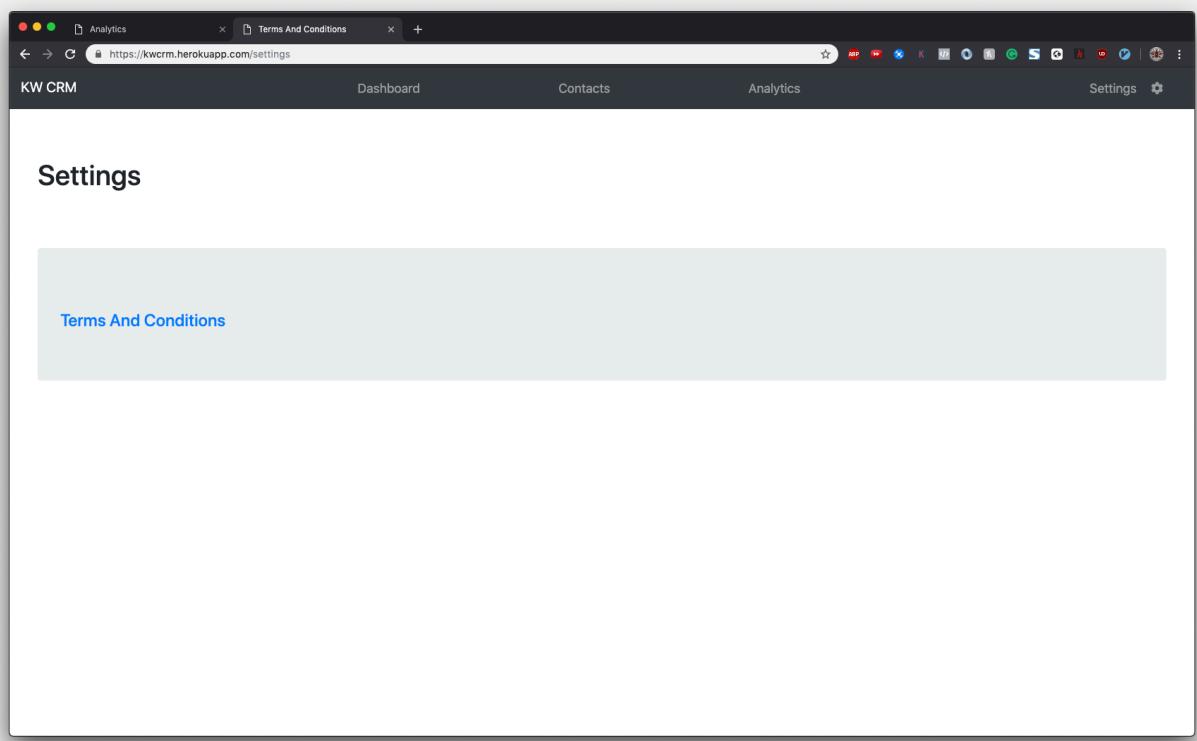
This is showing the weekly graphs



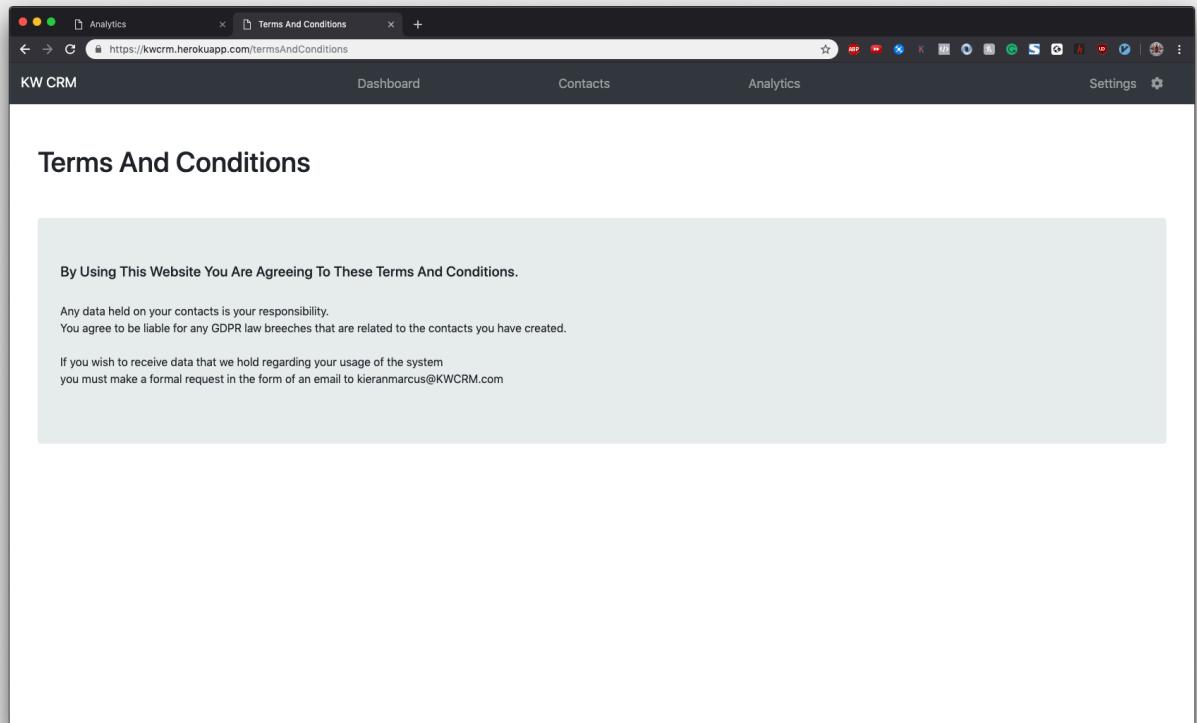
This is showing the monthly graphs

The Settings Page

The purpose of this page is to show any additional links and to alter any settings however there are no settings currently and so there is just one link to the terms and conditions page



The Terms And Conditions Page



This page outlines the terms and conditions the user is agreeing to by using the system in order to be in line with the GDPR legislation

The Individual Contact's Page

The screenshot shows the KW CRM Individual Contact page for a contact named "Allen Roger".

Contact Details:

- Business:** BigMerch
- Email:** alrog@gmail.com
- Phone Number:** 07496437123
- City:** Cardiff
- Postcode:** ca078dw
- Address:** 18 Main Street
- Contacts Status:** Past Client

Extra Notes:

```
54/28/11/9/16/72/30/3/5/66/18/90/50/83/23/9/71/11/5/4/0/17/80/65/31/113/28/84/9/87/7  
90/4/92/94/18/36
```

Note Addition Form:

Add Note

Note History Table:

Date	Notes
07/04/2019	sent allen an offer if he wishes to do business with us again he can have a discount
12/02/2019	big merch were lost as a client today

Action Buttons:

- Edit Contact
- Delete Contact
- Decrypt
- Save & Encrypt

The purpose of this page is to show all the details held on the contact, to input and view all the notes that are related to that contact and to display/ input/ encrypt/ decrypt the extra notes section

When a note gets added the current date also gets added to the record and is displayed along with the note in the table

When there is no data present in the extra notes section there is place holder piece of text that describes how to use the section

You can also edit the contact that is currently selected by pressing the edit contact button that redirects to the edit contact page. Pressing the delete contact button will delete the contact that is currently selected and redirect the user to the contact's list page

Successfully Added A Note

Allen Roger

When a note is added this message gets displayed

____ Please Ensure All Text Fields Have Data Entered ____ Ensure That The Subject Field Is Within Range Of 1 - 250 Characters ____

Allen Roger

These are all the possible validation messages there are for the notes input form

Extra Notes

REMEMBER THE PASSWORD! There Is No Way Of Getting The Data Back If You Forget It. Type Data into text area and press save & encrypt to save the data to the file with xor encryption done to it. this data can be changed by first decrypting it, changing the data and then pressing the save & encrypt button again. When data is decrypted it will not save anything back to the file. If the wrong decrypt password is entered, the decrypted text will not be the original text



Decrypt

Save & Encrypt

This is the place holder text that defines how to use the extra notes section to make it easier for the user to understand how to use it for the first time.

The screenshot shows a web browser window with the URL https://kwcrm.herokuapp.com/contacts/viewContact9rh7j15w2jqyx0m3/encryptNotes_decrypt. The page title is "Encrypt Extra Notes". The main content is a form titled "Encryption Password" with two input fields: "Password" and "Confirm Password". Below the fields is a checkbox labeled "See Password" which is checked. A note below the checkbox states: "Your password must be longer than 8 characters, contain upper and lower case letters, numbers, and must not contain spaces, special characters, or emoji." At the bottom of the form is a blue "Submit" button.

When the user types text into the extra notes section and goes to save it they get redirected to this page where they have to enter there password to encrypt or decrypt their data

Please Ensure All Text Fields Have Data Entered _____ Ensure That The password Field Is Within Range Of 8 - 15 Characters _____ Ensure That Password Consists Of At Least One: Uppercase Character, Lowercase Character And Number _____

Encryption Password

Password

Confirm Password

These are all the possible validation messages that can get displayed to the user

Extra Notes

54/28/11/9/16/72/30/3/5/66/18/90/50/83/23/9/71/11/5/4/0/17/80/65/31/113/28/84/9/87/7/
90/4/92/94/18/36

this is what the encrypted text will look like

Successfully Decrypted The Data

Allen Roger

Extra Notes

bob's personal phone number is 07496 427354

Business: BigMerch
Email: alrog@gmail.com
Phone Number: 07496437123

When decrypted this is what the page will look like with a message that states the data has been decrypted and the decrypted data in the text area which will then go back to being encrypted if the page gets refreshed

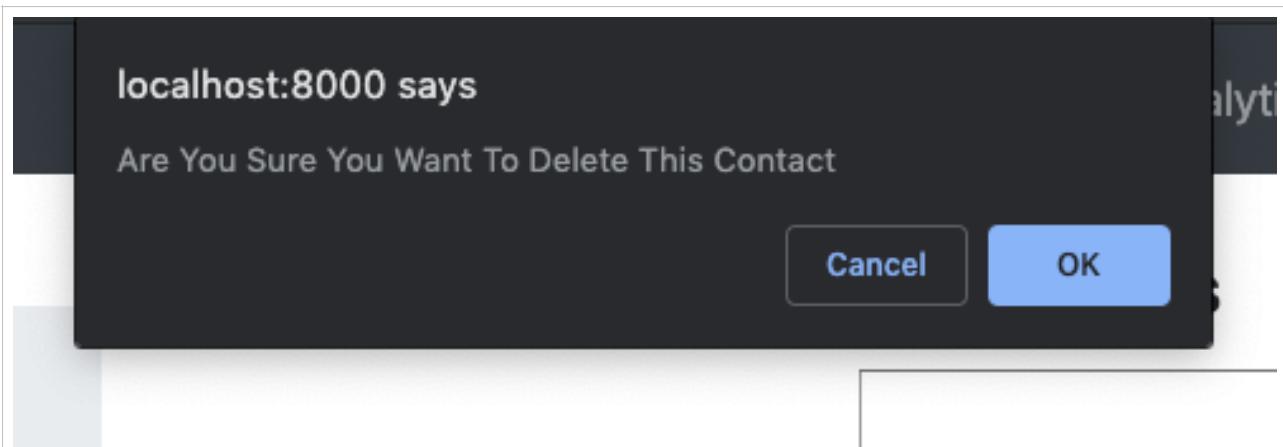
Successfully Changed The Encrypted Data

Allen Roger

Extra Notes

116/71/75/84/95/71/91/90/67/8/29/2/119/92/83/84/72/91/90/68/95/30/7/28/70/119/92/85/
94/72/81/90/68/89/1/29/11/114/92/86/85/72/81/69/92/93/30/3/2/95/69/110/71/75/86/72/
89/93/68/84/1/29/5/116/92/87/87/72/89/91/88/67/3/10/28/72/117/92/93/73/95/95/69/92/
67/8/2/28/117/92/93/84/72/81/94/68/93/9/29/0/70/101

This is the message that gets displayed to the user when they encrypt the data in the extra notes section



When the user goes to delete a contact, this confirmation dialog will get displayed to ensure that the user wants to delete the contact, upon pressing ok the contact will get deleted and the user will get redirected to the contacts list page, upon pressing cancel the delete request will get cancelled and the user will remain on the same page

The Edit Contact Page

when the edit contact button on the individual contact page gets clicked, the user will get redirected to this page where they can edit the contacts data and then press submit when finished. The contact's data will get updated if the input is all valid and then the user will get redirected to the contacts list page

Please Ensure All Text Fields Have Data Entered _____ Please Ensure The Telephone Length Is 11 Characters Long _____ Ensure That The Business Name Field Is Within Range Of 1 - 25 Characters _____
 Ensure That The First Name Field Is Within Range Of 1 - 15 Characters _____ Ensure That The Last Name Field Is Within Range Of 1 - 15 Characters _____ Ensure That The Email Field Is Within Range Of 4 - 30 Characters _____ Ensure That The City Field Is Within Range Of 1 - 15 Characters _____ Ensure That The Post Code Field Is Within Range Of 4 - 12 Characters _____ Ensure That The Address Field Is Within Range Of 1 - 30 Characters _____ Please Ensure The Email Is Valid _____

Edit Contact

These are all the possible validation messages that can get displayed due to invalid data entry in the edit contact form

The Create Contact Page

A screenshot of a web browser window showing the 'Create Contact' page. The browser title bar says 'Create Contact' and the address bar shows 'localhost:8000/contacts/new'. The page has a dark header with 'KW CRM', 'Dashboard', 'Contacts', 'Analytics', and 'Settings' tabs. Below the header, the main content area has a title 'Create Contact' and a sub-section 'Contact Details'. This section contains several input fields: 'Business' (text), 'First Name' (text), 'Last Name' (text), 'Email' (text), 'Telephone' (text), 'City' (text), 'Postcode' (text), 'Address' (text), and a dropdown 'Contact Status' with 'Lead' selected. At the bottom is a blue 'Submit' button.

The purpose of this page is to allow the user to create a new contact and add it to the contacts file store

____ Please Ensure All Text Fields Have Data Entered ____ Please Ensure The Telephone Length Is 11 Characters Long ____ Ensure That The Business Name Field Is Within Range Of 1 - 25 Characters ____
Ensure That The First Name Field Is Within Range Of 1 - 15 Characters ____ Ensure That The Last Name Field Is Within Range Of 1 - 15 Characters ____ Ensure That The Email Field Is Within Range Of 4 - 30
Characters ____ Ensure That The City Field Is Within Range Of 1 - 15 Characters ____ Ensure That The Post Code Field Is Within Range Of 4 - 12 Characters ____ Ensure That The Address Field Is Within
Range Of 1 - 30 Characters ____ Please Ensure The Email Is Valid ____

Create Contact

These are all the possible validation messages that can get displayed due to invalid data entry in the create contact form

Data Storage

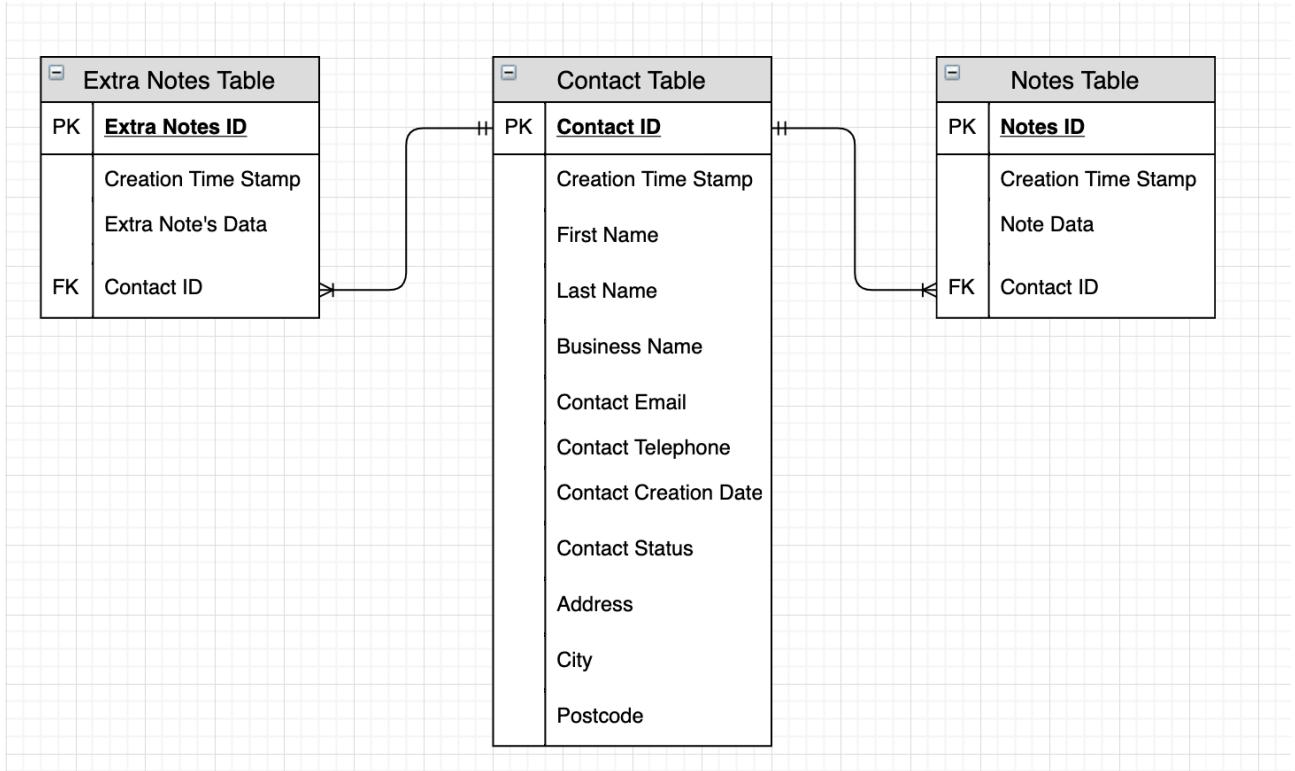
Entity Relationship Diagrams

Client Lost Table	
PK	<u>Client Lost ID</u>
	Creation Time Stamp

Client Gained Table	
PK	<u>Client Gained ID</u>
	Creation Time Stamp

Lead Lost Table	
PK	<u>Lead Lost ID</u>
	Creation Time Stamp

Scheduled Table	
PK	<u>Scheduled ID</u>
	Creation Time Stamp
	Scheduled Date
	Scheduled Time
	Subject
	Message



Data Dictionaries

* = Primary key

// = Foreign Key

Contacts Table

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	String	10	M123hjfdk343dfd
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
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.	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00
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 Table

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Extra Notes ID *	The unique identifying field. Will be used to identify one record in the extra notes file from another	String	10	M123hjfdk343dfd
Extra Notes	Will store the contact's Extra notes can be in plain text or as an encrypted string.	String	10,000	14/19/56/26/00/57/03/07/12/05/04
Extra Note Creation Date	Will store the date of when that extra note was created.	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00
Contact's ID //	The unique identifying field. Will be used to identify a contact	String	10	M123hjfdk343dfd

Notes Table

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
Notes ID *	The unique identifying field for a given note.	String	10	M123hjfdk343dfd
Note Creation Date	Will store the date of when that the note was created.	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00
Contact's ID //	The unique identifying field. Will be used to identify a contact	String	10	M123hjfdk343dfd

Scheduled Tasks Table

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	String	10	M123hjfdk343dfd
Scheduled Task Creation Date	Will store the date of when that the scheduled task was created.	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00
Scheduled Date	Will hold the scheduling's date	String	10	21/10/2019

Name Of Field	Description Of Field	Type Of Field	Size Of Field	Example Data
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.

Client Lost Table

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
Client Lost Date	Will store the date of when that client was lost (the date when the client lost button got pressed)	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00

Client Gained Table

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	245522
Client Gained Date	Will store the date of when that client was gained (the date when the client gained button got pressed)	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00

Lead Lost Table

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
Lead Lost Date	Will store the date of when that lead was lost (the date when the lead lost button got pressed)	ISO 8601 date and time format	70	2014-09-08T08:02:17-05:00

Populated Data Store Files

Contacts Extra Notes File (.CSV)	
1	17gdv3216js262g0b,2019-02-12T19:35:47+00:00,mlwmc41eznjrxy3t15,41/22/8/10/8/72/30/3/9/67/87/19/54/28/22/10/3/27/18,

2 mlwmc41xpnjs0k9qlf,2019-02-11T00:00:00+00:00,9rh7j15w2jrqyx0m3,54/28/11/9/16/72/30/3/5/66/18/90/50/83/23/9/71/11/5/4/0/17/80/65/31/113/28/84/9/87/7/90/4/92/94/18/36,

3 mlwmc41y9ajs0m3o0b,2019-02-11T00:00:00+00:00,mlwmc4139ojrwkirmu,9/22/8/10/8/72/29/4/62/93/86/11,

4 mlwmc42118js0s8jqa,2019-02-11T20:20:51+00:00,9rh7j15xxjrqz103c,41/22/22/70/9/9/30/2/3/95/83/95/97/26/10/21/18/26/11/5/15/84/18/93/5/44/17/1/20/71/1/25/75/66/31/28/29/111/93/74/72/73/70/74/3/9/93/94/92/80/4/41/22/22/3/55,

5

Contacts Notes (.CSV)

1 17gdv3216js260h5v,2019-02-12T19:34:15+00:00,12/02/2019,mlwmc41eznjrxy3t15,he needed
time to think over what we discussed call next week,
2 17gdv33s8js2aejqi,2019-02-12T21:37:10+00:00,12/02/2019,9rh7j15w2jrqyx0m3,big merch
were lost as a client today,
3

Clients Lost (.CSV)

1 mlwmc4222wfs0dtn81,2019-02-02T00:00:00+00:00,2019-02,
2 mlwmc4222wfs0dtn82,2019-02-02T00:00:00+00:00,2019-02,
3 mlwmc4222wfs0xtn83,2019-02-04T00:00:00+00:00,2019-02,
4 mlwmc4222wfs0xan84,2019-02-09T00:00:00+00:00,2019-02,
5 slwmc4222wgs0xsyxb,2019-03-11T00:00:00+00:00,2019-03,
6 flwmc4222dgs0xan8u,2019-05-11T00:00:00+00:00,2019-05,
7 hlwmc4222wjs0xtd8u,2019-07-11T00:00:00+00:00,2019-07,
8 llwmc4222wddtyfgxb,2019-08-11T00:00:00+00:00,2019-08,
9 dlwmc4222sjdfstn8u,2019-08-11T00:00:00+00:00,2019-08,
10 qlwmc4222fjs0xtyfg,2019-09-11T00:00:00+00:00,2019-09,
11 elwmc4222wjggsdxxb,2019-10-11T00:00:00+00:00,2019-10,
12 rlwmc4222wjshxtn8u,2019-11-11T00:00:00+00:00,2019-11,
13 ht1wmc4222gwjkdtxb,2019-12-11T00:00:00+00:00,2019-12,
14 ulwmc4222ssxtntsgu,2019-01-11T00:00:00+00:00,2019-01,
15 olwmc4222wjgaxtyxb,2018-06-11T00:00:00+00:00,2018-06,
16 17gdv314fjs5ebpfy,2019-02-15T00:00:00+00:00,2019-02,
17 17gdv31buj5sedify,2019-02-15T00:00:00+00:00,2019-02,
18 17gdv31buj5edkxs,2019-02-15T00:00:00+00:00,2019-02,
19 17gdv31buj5ee1kd,2019-02-15T00:00:00+00:00,2019-02,
20 17gdv31lfjs5f25ss,2019-02-15T00:00:00+00:00,2019-02,
21 17gdv31lfjs5f275t,2019-02-15T00:00:00+00:00,2019-02,
22 17gdv31aojj57x45n0,2019-02-16T00:00:00+00:00,2019-02,
23 17gdv31aojj57x45rb,2019-02-16T00:00:00+00:00,2019-02,
24 17gdv31aojj57x46aw,2019-02-16T00:00:00+00:00,2019-02,
25 17gdv31aojj57x46ff,2019-02-16T00:00:00+00:00,2019-02,
26 17gdv31aojj57x46ka,2019-02-16T00:00:00+00:00,2019-02,
27 17gdv31aojj57x46os,2019-02-16T00:00:00+00:00,2019-02,
28 17gdv31aojj57x46sy,2019-02-16T00:00:00+00:00,2019-02,
29 17gdv31aojj57x46xf,2019-02-16T00:00:00+00:00,2019-02,
30 17gdv31aojj57x4peg,2019-02-16T00:00:00+00:00,2019-02,
31 17gdv31aojj57x4sgy,2019-02-16T00:00:00+00:00,2019-02,
32 17gdv31aojj57x4s1k,2019-02-16T00:00:00+00:00,2019-02,
33 17gdv31aojj57x4t61,2019-02-16T00:00:00+00:00,2019-02,
34 17gdv31aojj57x4tav,2019-02-16T00:00:00+00:00,2019-02,
35 17gdv31aojj57x4tff,2019-02-16T00:00:00+00:00,2019-02,
36 17gdv31aojj57x4tjl,2019-02-16T00:00:00+00:00,2019-02,
37 17gdv31aojj57x4tok,2019-02-16T00:00:00+00:00,2019-02,
38 17gdv31aojj57x4tt7,2019-02-16T00:00:00+00:00,2019-02,
39 17gdv31aojj57x4txv,2019-02-16T00:00:00+00:00,2019-02,
40 17gdv31aojj57x4u2h,2019-02-16T00:00:00+00:00,2019-02,
41 17gdv31aojj57x4u79,2019-02-16T00:00:00+00:00,2019-02,
42

Contacts (.CSV)

1 9rh7j15w2jrqyx0m3,2019-02-04T00:00:00+00:00,BigMerch,Allen,Roger,alrog@gmail.com,07496436543,Cardiff,ca078dw,18 Main Street,past-client,
 2 9rh7j15xxjrqz103c,2019-02-04T00:00:00+00:00,Dynasty,Sally,Hughes,dynastyClothing@gmail.com,07496467098,Neath,sa95nb,Abbey Road,dead-lead,
 3 9rh7j16i2jrqz125e,2019-02-04T00:00:00+00:00,Paulsmith
 Guitars,Paul,Smith,psGuitars@gmail.com,07436890212,Swansea,sa89op,78 Green
 Street,lead,
 4 mlwmc41eznjrxy2mfj,2019-02-09T00:00:00+00:00,Joe
 Skate,Joe,Rob,joeyskater@gmail.com,01792782635,Port Talbot,sa94bw,32 Blah Blah
 Road,recontact-lead,
 5 mlwmc41eznjrxy3t15,2019-02-09T00:00:00+00:00,Big Breakfasts 4
 You,Jamie,Roberts,jamierobs@outlook.com,07496436543,London,Ln09eb,Queens
 Street,recontact-lead,
 6 mlwmc41eznjrxy48n0,2019-02-09T00:00:00+00:00,James
 Media,James,Brinks,jamesbrinks@hotmail.com,01792946751,Swansea,sa14na,22 Abc Castle
 Road,recontact-lead,
 7 mlwmc41eznjrxy6sm3,2019-02-09T00:00:00+00:00,The Network
 Guys,Bob,Wally,boobywallyIT@networkgys.com,01792876784,Swansea,sa14na,22 Abc Castle
 Road,lead,
 8 mlwmc41z37js0ote8c,2019-02-11T00:00:00+00:00,Breakfasts U
 Want,Rob,Sally,robsysally@breakfasts.com,07496436543,Swansea,sa19op,Kingsway,past-clie
 nt,
 9 mlwmc4a6ojrt8dqpg,2019-02-06T00:00:00+00:00,Dig Dig
 Diggers,Larry,Watson,lazzywatz@gmail.com,07456897564,Port Talbot,sa131dp,Station
 Road,current-client,
 10

Leads Lost (.CSV)

1	mlwmc4222wfs0xtn81, 2019-02-02T00:00:00+00:00, 2019-02,
2	mlwmc4222wfs0xtn82, 2019-02-02T00:00:00+00:00, 2019-02,
3	mlwmc4222was0xtn82, 2019-02-05T00:00:00+00:00, 2019-02,
4	mlwmc4222wfs0xtn83, 2019-02-04T00:00:00+00:00, 2019-02,
5	mlwmc4222wfs0xtn84, 2019-02-09T00:00:00+00:00, 2019-02,
6	mlwmc4222wfs0xtn85, 2019-02-11T00:00:00+00:00, 2019-02,
7	mlwmc4222afs0xtn85, 2019-02-11T00:00:00+00:00, 2019-02,
8	slwmc4222wgs0xtyx, 2019-03-11T00:00:00+00:00, 2019-03,
9	flwmc4222dgs0xtn8u, 2019-05-11T00:00:00+00:00, 2019-05,
10	glwmc4222whg0xtyx, 2019-05-11T00:00:00+00:00, 2019-05,
11	hlwmc4222wjs0xtn8u, 2019-07-11T00:00:00+00:00, 2019-07,
12	jlwmc4222wls0xtyx, 2019-08-11T00:00:00+00:00, 2019-08,
13	klwmc4222dfs0xtn8u, 2019-08-11T00:00:00+00:00, 2019-08,
14	llwmc4222wdg0xtyx, 2019-08-11T00:00:00+00:00, 2019-08,
15	dlwmc4222sjs0xtn8u, 2019-08-11T00:00:00+00:00, 2019-08,
16	qlwmc4222fjs0xtyx, 2019-09-11T00:00:00+00:00, 2019-09,
17	wlwmc4222dfs0xtn8u, 2019-09-11T00:00:00+00:00, 2019-09,
18	elwmc4222wjggxtyx, 2019-10-11T00:00:00+00:00, 2019-10,
19	rlwmc4222wjshxtn8u, 2019-11-11T00:00:00+00:00, 2019-11,
20	htlwmc4222wjk0xtx, 2019-12-11T00:00:00+00:00, 2019-12,
21	ulwmc4222djssxtn8u, 2019-01-11T00:00:00+00:00, 2019-01,
22	olwmc4222wjgaxtyx, 2019-01-11T00:00:00+00:00, 2019-01,
23	17gdv31bujs5ed1z9, 2019-02-15T00:00:00+00:00, 2019-02,
24	17gdv31bujs5edmvy, 2019-02-15T00:00:00+00:00, 2019-02,
25	17gdv31bujs5edo4k, 2019-02-15T00:00:00+00:00, 2019-02,
26	17gdv31bujs5edp5b, 2019-02-15T00:00:00+00:00, 2019-02,
27	17gdv31bujs5edqgz, 2019-02-15T00:00:00+00:00, 2019-02,
28	17gdv31bujs5edr4o, 2019-02-15T00:00:00+00:00, 2019-02,
29	17gdv31bujs5edrhz, 2019-02-15T00:00:00+00:00, 2019-02,
30	17gdv31bujs5edrk2, 2019-02-15T00:00:00+00:00, 2019-02,
31	17gdv31bujs5edrsu, 2019-02-15T00:00:00+00:00, 2019-02,
32	17gdv31bujs5edrx, 2019-02-15T00:00:00+00:00, 2019-02,
33	17gdv31bujs5eds7z, 2019-02-15T00:00:00+00:00, 2019-02,
34	17gdv31bujs5eds15, 2019-02-15T00:00:00+00:00, 2019-02,
35	17gdv31bujs5edtia, 2019-02-15T00:00:00+00:00, 2019-02,
36	17gdv31bujs5educ3, 2019-02-15T00:00:00+00:00, 2019-02,
37	17gdv31bujs5edv96, 2019-02-15T00:00:00+00:00, 2019-02,
38	17gdv31lfjs5f1cas, 2019-02-15T00:00:00+00:00, 2019-02,
39	17gdv31lfjs5f1m0u, 2019-02-15T00:00:00+00:00, 2019-02,
40	17gdv31lfjs5f287o, 2019-02-15T00:00:00+00:00, 2019-02,
41	17gdv31lfjs5f29wj, 2019-02-15T00:00:00+00:00, 2019-02,
42	17gdv31lfjs5f2awb, 2019-02-15T00:00:00+00:00, 2019-02,
43	17gdv31lfjs5f2c1c, 2019-02-15T00:00:00+00:00, 2019-02,
44	17gdv31lfjs5f2d4o, 2019-02-15T00:00:00+00:00, 2019-02,
45	17adv3maris62zbka, 2019-02-15T00:00:00+00:00, 2019-02.

Scheduled (.CSV)

1	mlwmc41hs6jry3i2za,2019-02-09T00:00:00+00:00,2019-02-29T00:00:00+00:00,29/02/2019,00:00,Call Sally,hell yeah,
2	

Clients Gained (.CSV)	
1	mlwmc4222wfs0xtn81, 2019-00-02T00:00:00+00:00, 2019-02,
2	mlwmc4222wfs0xtn82, 2019-02-02T00:00:00+00:00, 2019-02,
3	mlwmc4222was0xtn82, 2019-02-05T00:00:00+00:00, 2019-02,
4	mlwmc4222wfs0xtn83, 2019-02-04T00:00:00+00:00, 2019-02,
5	mlwmc4222wfs0xtn84, 2019-02-09T00:00:00+00:00, 2019-02,
6	mlwmc4222wfs0xtn85, 2019-02-11T00:00:00+00:00, 2019-02,
7	mlwmc4222afs0xtn85, 2019-02-11T00:00:00+00:00, 2019-02,
8	slwmc4222wgs0xtyx, 2019-03-11T00:00:00+00:00, 2019-03,
9	flwmc4222dgs0xtn8u, 2019-05-11T00:00:00+00:00, 2019-05,
10	glwmc4222whg0xtyx, 2019-05-11T00:00:00+00:00, 2019-05,
11	hlwmc4222wjs0xtn8u, 2019-06-11T00:00:00+00:00, 2019-06,
12	jlwmc4222wls0xtyx, 2019-04-11T00:00:00+00:00, 2019-04,
13	klwmc4222dfs0xtn8u, 2019-04-11T00:00:00+00:00, 2019-04,
14	llwmc4222wdg0xtyx, 2019-04-11T00:00:00+00:00, 2019-04,
15	dlwmc4222sjs0xtn8u, 2019-04-11T00:00:00+00:00, 2019-04,
16	qlwmc4222fjs0xtyx, 2019-03-11T00:00:00+00:00, 2019-03,
17	wlwmc4222dfs0xtn8u, 2019-03-11T00:00:00+00:00, 2019-03,
18	elwmc4222wjggxtyx, 2019-03-11T00:00:00+00:00, 2019-03,
19	rlwmc4222wjshxtn8u, 2019-03-11T00:00:00+00:00, 2019-03,
20	htlwmc4222wjk0xtx, 2019-03-11T00:00:00+00:00, 2019-03,
21	ulwmc4222djssxtn8u, 2019-01-11T00:00:00+00:00, 2019-01,
22	olwmc4222wjgaxtyx, 2019-01-11T00:00:00+00:00, 2019-01,
23	17gdv314fjs5eb2hu, 2019-02-15T00:00:00+00:00, 2019-02,
24	17gdv314fjs5ed3bp, 2019-02-15T00:00:00+00:00, 2019-02,
25	17gdv3lbujs5edcue, 2019-02-15T00:00:00+00:00, 2019-02,
26	17gdv3lbujs5ede7z, 2019-02-15T00:00:00+00:00, 2019-02,
27	17gdv3lbujs5edfuf, 2019-02-15T00:00:00+00:00, 2019-02,
28	17gdv3lbujs5eeqyh, 2019-02-13T00:00:00+00:00, 2019-02,
29	17gdv3lbujs5eervx, 2019-02-11T00:00:00+00:00, 2019-02,
30	17gdv311fjs5f118r, 2019-02-09T00:00:00+00:00, 2019-02,
31	17gdv311fjs5f19ew, 2019-02-03T00:00:00+00:00, 2019-02,
32	17gdv311fjs5f19ew, 2019-02-03T00:00:00+00:00, 2019-02,
33	17gdv311fjs5f19ew, 2019-02-03T00:00:00+00:00, 2019-02,
34	17gdv311fjs5f19ew, 2019-02-03T00:00:00+00:00, 2019-02,
35	17gdv311fjs5f1skm, 2019-02-15T00:00:00+00:00, 2019-02,
36	17gdv311fjs5f1vel, 2019-02-15T00:00:00+00:00, 2019-02,
37	17gdv311fjs5f1wiz, 2019-02-15T00:00:00+00:00, 2019-02,
38	17gdv311fjs5f1xxr, 2019-02-15T00:00:00+00:00, 2019-02,
39	17gdv311fjs5f1z61, 2019-02-15T00:00:00+00:00, 2019-02,
40	17gdv3marjs62z8fx, 2019-02-15T00:00:00+00:00, 2019-02,
41	wlwmc4222dfs0xtn8u, 2018-09-11T00:00:00+00:00, 2018-09,
42	elwmc4222wjggxtyx, 2018-10-11T00:00:00+00:00, 2018-10,
43	rlwmc4222wjshxtn8u, 2018-11-11T00:00:00+00:00, 2018-11,
44	htlwmc4222wjk0xtx, 2018-12-11T00:00:00+00:00, 2018-12,

Validation Methods By Type Of Validation

Validation Validation Code Snippet Type

Look Up Check

```

404 // preforms a look up check on the string comparing every character in the argument string
405 // to an array of acceptable characters. true is returned if there is a match found in the string to the list of
406 // acceptable characters, false returned if there is no match.
407 Validation[_stringLookupCheckForLowerCase] = (stringValue) => {
408     let lowerCaseFlag = false;
409     // all possible lower case letters put into an array of characters
410     let lowerCaseArray = "abcdefghijklmnopqrstuvwxyz".split("");
411     // The stringValue argument gets split up into an array of characters
412     let stringArray = stringValue.split("");
413     let currentCharacter = '';
414
415     // check for a lower case character in the stringValue
416     // by iterating through the stringArray and checking to see if a value matches any value in the lowerCaseArray
417     // this is achieved through using a NESTED FOR LOOP
418     for (let i = 0; i < stringArray.length; i++) {
419         currentCharacter = stringArray[i];
420         for (let j = 0; j < lowerCaseArray.length; j++) {
421             // if a value in the stringArray matches the value in the lowerCaseArray, set the lowerCaseFlag to true
422             if (currentCharacter === lowerCaseArray[j] ) {
423                 lowerCaseFlag = true;
424             }
425         }
426     }
427     // if the lowerCaseFlag is still set to false after the loop, then a lowerCase character doesn't exist, false is returned
428     return lowerCaseFlag;
429 };
430
431
432 // preforms a look up check on the string comparing every character in the argument string
433 // to an array of acceptable characters. true is returned if there is a match found in the string to the list of
434 // acceptable characters, false returned if there is no match.
435 Validation[_stringLookupCheckForUpperCase] = (stringValue) => {
436     let upperCaseFlag = false;
437     // all possible upper case letters put into an array of characters
438     let upperCaseArray = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".split("");
439     // The stringValue argument gets split up into an array of characters
440     let stringArray = stringValue.split("");
441     let currentCharacter = '';
442
443     // check for a upper case character in the stringValue
444     // by iterating through the stringArray and checking to see if a value matches any value in the upperCaseArray
445     // this is achieved through using a NESTED FOR LOOP
446     for (let i = 0; i < stringArray.length; i++) {
447         currentCharacter = stringArray[i];
448         for (let j = 0; j < upperCaseArray.length; j++) {
449             // if a value in the stringArray matches the value in the upperCaseArray, set the upperCaseFlag to true
450             if (currentCharacter === upperCaseArray[j] ) {
451                 upperCaseFlag = true;
452             }
453         }
454     }
455     // if the upperCaseFlag is still set to false after the loop, then an upperCase character doesn't exist, false is returned
456     return upperCaseFlag;
457 };
458

```

Validation Validation Code Snippet

Type

```

460 // preforms a look up check on the string comparing every character in the argument string
461 // to an array of acceptable characters. true is returned if there is a match found in the string to the list of
462 // acceptable characters, false returned if there is no match.
463 Validation[_stringLookupCheckForNumbers] = (stringValue) => {
464     let numbersFlag = false;
465     // all possible number characters put into an array of characters
466     let numbersArray = "1234567890".split("");
467     // The stringValue argument gets split up into an array of characters
468     let stringArray = stringValue.split("");
469     let currentCharacter = '';
470
471     // check for a number character in the stringValue
472     // by iterating through the stringArray and checking to see if a value matches any value in the numbersArray
473     // this is achieved through using a NESTED FOR LOOP
474     for (let i = 0; i < stringArray.length; i++) {
475         currentCharacter = stringArray[i];
476         for (let j = 0; j < numbersArray.length; j++) {
477             // if a value in the stringArray matches the value in the numbersArray, set the numbersFlag to true
478             if (currentCharacter === numbersArray[j]) {
479                 numbersFlag = true;
480             }
481         }
482     }
483     // if the numbersFlag is still set to false after the loop, then a number character doesn't exist, false is returned
484     return numbersFlag;
485 };
486
487
488 // preforms a look up check on the string comparing every character in the argument string
489 // to a space character. true is returned if there is a match found in the string, false returned if there is no match.
490 Validation[_stringLookupCheckForSpaces] = (stringValue) => {
491     let spaceFlag = false;
492     // the comparison space character
493     let spaceCharacter = " ";
494     // The stringValue argument gets split up into an array of characters
495     let stringArray = stringValue.split("");
496     let currentCharacter = '';
497
498     // check for a space character in the stringValue
499     // by iterating through the stringArray and checking to see if a value matches any value in the spaceCharacter
500     for (let i = 0; i < stringArray.length; i++) {
501         currentCharacter = stringArray[i];
502         if (currentCharacter === spaceCharacter) {
503             spaceFlag = true;
504         }
505     }
506     // if the spaceFlag is still set to false after the loop, then a space character doesn't exist, false is returned
507     return spaceFlag;
508 };
509

```

Validation Validation Code Snippet**Type**

```
510
511 // preforms a look up check on the string comparing every character in the argument string
512 // to a comma character. true is returned if there is a match found in the string, false returned if there is no match.
513 Validation.stringLookupCheckForCommas = (stringValue) => {
514     let commaFlag = false;
515     // the comparison comma character
516     let commaCharacter = ",";
517     // The stringValue argument gets split up into an array of characters
518     let stringArray = stringValue.split("");
519     let currentCharacter = '';
520
521     // check for a number character in the stringValue
522     // by iterating through the stringArray and checking to see if a value matches any value in the commaCharacter
523     for (let i = 0; i < stringArray.length; i++) {
524         currentCharacter = stringArray[i];
525         if (currentCharacter === commaCharacter) {
526             commaFlag = true;
527         }
528     }
529     // if the commaFlag is still set to false after the loop, then a comma character doesn't exist, false is returned
530     return commaFlag;
531 };
532
```

Validation Validation Code Snippet

Type

```

Validation.stringLookupCheck = (stringValue, objectOfWhatNeedsToBeInTheString) => {

    // if there is no object parameter an error message is thrown to
    // notify that an object parameter is needed
    if (objectOfWhatNeedsToBeInTheString === undefined) {
        throw "ERROR NEED TO HAVE AN OBJECT PARAMETER: "
        + "as the second parameter, set options of:\nlowerCase, upperCase, numbers: " +
        " to true \nif you want to check the string contains one of them types of characters ";
    }

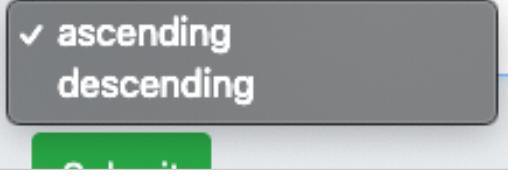
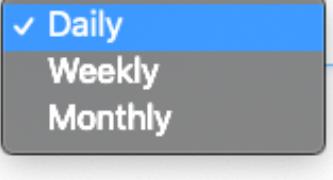
    // checks to see if there is a property in the object passed as argument into the method
    // called lowerCase and checks to see if it's value is set to the boolean value true
    if (objectOfWhatNeedsToBeInTheString.lowerCase === true) {
        // preforms lookup check to see if there is any lowerCase characters in the string
        // if there is not then false gets returned
        if(!Validation[_stringLookupCheckForLowerCase](stringValue)){ return false }
    }

    // checks to see if there is a property in the object passed as argument into the method
    // called upperCase and checks to see if it's value is set to the boolean value true
    if (objectOfWhatNeedsToBeInTheString.upperCase === true) {
        // preforms lookup check to see if there is any upperCase characters in the string
        // if there is not then false gets returned
        if(!Validation[_stringLookupCheckForUpperCase](stringValue)){ return false }
    }

    // checks to see if there is a property in the object passed as argument into the method
    // called numbers and checks to see if it's value is set to the boolean value true
    if (objectOfWhatNeedsToBeInTheString.numbers === true) {
        // preforms lookup check to see if there is any number characters in the string
        // if there is not then false gets returned
        if(!Validation[_stringLookupCheckForNumbers](stringValue)){ return false }
    }

    // if false hasn't been returned yet then the string must contain all criteria
    return true;
};

```

Type	Validation Validation Code Snippet
	<pre> 573 574 // method checks that the string passed in as an argument only contains lower case characters, upper case characters 575 // or number characters. if a character is found in the string that is not one of them, then the method returns false 576 // if all characters in the string that is passed as argument are either upper, lower or numbers then true is returned 577 Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars = (stringValue, allowSpaces) => { 578 // iterate through each character in the string checking if it is a number, lowercase char or an uppercase char 579 let curChar = ''; 580 let charArray = stringValue.split(""); // puts the string into an array of its characters 581 for (let i = 0; i < charArray.length; i++) { 582 583 curChar = charArray[i]; 584 585 if (allowSpaces === true) { 586 // if all of the checks come back false, false is returned 587 if (!Validation[_stringLookupCheckForLowerCase](curChar)) 588 && (!Validation[_stringLookupCheckForUpperCase](curChar)) 589 && (!Validation[_stringLookupCheckForNumbers](curChar)) 590 && (!Validation[_stringLookupCheckForSpaces](curChar))) 591 { 592 return false; 593 } 594 } 595 596 else { 597 // if all of the checks come back false, false is returned 598 if (!Validation[_stringLookupCheckForLowerCase](curChar)) 599 && (!Validation[_stringLookupCheckForUpperCase](curChar)) 600 && (!Validation[_stringLookupCheckForNumbers](curChar)) 601 { 602 return false; 603 } 604 } 605 606 } 607 608 // loop has finished, if false hasn't been returned it means that all the characters 609 // in the string consist of either a number, lowercase char or an uppercase char 610 return true; 611 }; 612 </pre>
	<p>Order</p> 
	<p>Time Scale</p> 

Validation Type	Validation Code Snippet
	<p>Contact Status</p> <ul style="list-style-type: none"> ✓ Lead Recontact Lead Dead Lead Current Client Past Client
	<p>localhost:8000 says</p> <p>Are You Sure You Want To Delete This Contact</p> <p style="text-align: right;">Cancel OK</p>
Type Check	<pre> 37 Validation.isaNumber = (value, stringToNumberConversionNeeded) => { 38 39 if (stringToNumberConversionNeeded === true) { 40 value = Number(value); 41 } 42 43 return typeof(value) === "number" && !isNaN(value); 44 }; 45 </pre> <pre> 47 Validation.isaInt = (value) => { 48 return typeof(value) === "number" && !isNaN(value) && value % 1 === 0; 49 }; </pre> <pre> 51 Validation.isaReal = (value) => { 52 return typeof(value) === "number" && !isNaN(value) && value % 1 !== 0; 53 }; 54 </pre> <pre> 55 Validation.isaString = (value) => { 56 return typeof(value) === "string"; 57 }; 58 </pre>

Validation	Validation Code Snippet
Type	<pre>59 Validation.isaBoolean = (value) => { 60 return typeof(value) === "boolean"; 61 }; 62 63 Validation.isaFunction = (value) => { 64 return typeof(value) === "function"; 65 }; 66</pre>
	<pre>67 Validation.isaObject = (value) => { 68 return typeof(value) === "object"; 69 }; 70</pre>
	<pre>71 Validation.isaArray = (value) => { 72 return Array.isArray(value); 73 }; 74</pre>
	<pre>75 Validation.isaCharacter = (value) => { 76 return typeof(value) === "string" && Validation.lengthCheck(value, 1); 77 }; 78</pre>
Length Check	<pre>12 Validation.lengthCheck = (value, length) => { 13 return value.length === length; 14 }; 15</pre>

Validation	Validation Code Snippet
Type	
Range Check	<pre> 16 Validation.rangeCheck = (value, lower, upper, stringLengthInRange) => { 17 if (typeof value === "string"){ 18 if (stringLengthInRange) { 19 value = value.length; 20 } 21 else { 22 value = Number(value); 23 } 24 } 25 26 if (upper === undefined upper === null) { 27 upper = lower; 28 } 29 30 return (value >= lower && value <= upper); 31 32 }; </pre>
Format Check	<pre> 285 Validation.timeFormat_HHMM = (value) => { 286 287 let valueArray = []; 288 let val = 0; 289 let hour, minute; 290 291 // checks that the input is a string and that it has a length of 5 characters 292 if(Validation.isaString(value) && Validation.lengthCheck(value, 5)) { 293 294 valueArray = value.split(""); // puts the string into an array 295 296 // loops through every element in the array so that checks can be done on the hour and minute parts of the time 297 for (let i = 0; i < valueArray.length; i++) { 298 299 // will be using this a lot so to make it more readable, the current array element is set to the variable val 300 val = valueArray[i]; 301 302 // the 2 index in the array as the element should be a ':' if it is not, false is returned 303 if (i === 2) { 304 if (val !== ":") { 305 return false; 306 } 307 } 308 else { 309 310 // checks that any other character is a number value, if it isn't false is returned 311 if (!Validation.isaNumber(Number(val))) { 312 return false; 313 } 314 } 315 } 316 317 hour = valueArray[0] + valueArray[1]; 318 minute = valueArray[3] + valueArray[4]; 319 320 return ((Number(hour) >= 0 && Number(hour) <= 23 && Number(minute) >= 0 && Number(minute) <= 59)); 321 } 322 else { 323 return false; 324 } 325 }; 326 </pre>

Validation	Validation Code Snippet
Type	<pre> 84 Validation.fileIsTXT = (file) => { 85 let regex = /^[a-zA-Z0-9\s_\\.\-:]*)+\\.txt\$/; 86 return (regex.test(file.value)); 87 }; 88 </pre>
	<pre> 79 Validation.fileIsCSV = (file) => { 80 let regex = /^[a-zA-Z0-9\s_\\.\-:]*)+\\.csv\$/; 81 return (regex.test(file)); 82 }; 83 </pre>
	<pre> 327 Validation.emailFormat = (value) => { 328 let emailRegex = /^(([^<>()\\[\\]\\.,;:\\s@"]+(\.[^<>()\\[\\]\\.,;:\\s@"]+)*) ".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}]) (([a-zA-Z-\w-]+\.)+[a-zA-Z]{2,}))\$/ ; 329 return emailRegex.test(value); 330 }; 331 </pre>
Presence Check	<pre> 8 Validation.presenceCheck = (value) => { 9 return !(value === undefined value === null /\s*\$/.test(value)); 10 }; </pre>
Comparison Check	<pre> 382 // EXAMPLE OF A POLYMORPHIC METHOD 383 // value two can be an array to check valueOne against multiple values in 384 // the valueTwo argument. although valueTwo can be a single value to check equality 385 Validation.comparisonCheck = (valueOne, valueTwo) => { 386 if (typeof valueTwo === "object") { 387 let foundMatchFlag = false; 388 // iterate through the array and compare valueOne to all the values in the array, if one matches then return true 389 // if there's no matches return false 390 for (let i = 0; i < valueTwo.length; i++) { 391 if (valueOne === valueTwo[i]) { 392 foundMatchFlag = true; 393 } 394 } 395 return foundMatchFlag; 396 } else { // if valueTwo is not an array then check for equality on the single value 397 return valueOne === valueTwo; 398 } 399 } 400 401 ; </pre>

Source Code For Final System

Customer Relationship
Management System (CRM)

Kieran Williams

Root Directory Contents

Index.js

```
1 const express = require("express");
2 const app = express();
3 const methodOverride = require("method-override");
4 const bodyParser = require("body-parser");
5 const session = require('express-session');
6
7 //requiring routes
8 const
9     indexRoutes = require("./routes/index"),
10    analyticsRoutes = require("./routes/analyticsRoutes"),
11    dashboardRoutes = require("./routes/dashboardRoutes"),
12    contactsRoutes = require("./routes/contactsRoutes");
13
14
15
16
17 // so that I can pass data between the backend and front end without just xhr requests
18 // .urlencoded({extended: true}) allows me to pass nested objects to the front end
19 // and from the front end to the server
20 app.use(bodyParser.urlencoded({extended: true}));
21 app.set("view engine", "ejs"); //sets my view engine to the ejs templating language
22 which will allow me to use js in the mark up
23 app.use('/public', express.static(__dirname + '/public'));
24 app.use(express.static(__dirname + '/public')); // express middleware to serve the
25 static files such as css and js to the client using the express HTTP framework
26 app.use(methodOverride("_method")); // methodOverride will allow me to use more http
27 verbs than html5 forms support
28 app.use(session({secret: 'mySecret', resave: false, saveUninitialized: false})); // will
29 allow me to store data to the session property
30
31 // setting my express http framework to use these url routes as the route paths for
32 // each
33 // of the different types of routes to aid readability with RESTFUL ROUTES for each
34 // of the routes
35 // the routes that extend from their base
36 app.use("/", indexRoutes);
37 app.use("/contacts", contactsRoutes);
38 app.use("/analytics", analyticsRoutes);
39 app.use("/dashboard", dashboardRoutes);
40
41 app.listen(process.env.PORT || 8000, (err) => {
42     if (err) {
43         console.log(err);
44     }
45     else {
46         console.log("server started on port 8000");
47     }
48 });
49
```

Package-lock.json

```
1  {
2    "name": "kwcrmCourseworkProject",
3    "version": "1.0.0",
4    "lockfileVersion": 1,
5    "requires": true,
6    "dependencies": {
7      "accepts": {
8        "version": "1.3.5",
9        "resolved": "https://registry.npmjs.org/accepts/-/accepts-1.3.5.tgz",
10       "integrity": "sha1-63d99gEXI6OxTopywIBcjoZ0a9I=",
11       "requires": {
12         "mime-types": "2.1.20",
13         "negotiator": "0.6.1"
14       }
15     },
16     "array-flatten": {
17       "version": "1.1.1",
18       "resolved": "https://registry.npmjs.org/array-flatten/-/array-flatten-1.1.1.tgz",
19       "integrity": "sha1-ml9pkFGx5wczKPKgCJaLZOpVdI="
20     },
21     "body-parser": {
22       "version": "1.18.3",
23       "resolved": "https://registry.npmjs.org/body-parser/-/body-parser-1.18.3.tgz",
24       "integrity": "sha1-WykhmP/dVTs6DyDe0FkrlWlVyLQ=",
25       "requires": {
26         "bytes": "3.0.0",
27         "content-type": "1.0.4",
28         "debug": "2.6.9",
29         "depd": "1.1.2",
30         "http-errors": "1.6.3",
31         "iconv-lite": "0.4.23",
32         "on-finished": "2.3.0",
33         "qs": "6.5.2",
34         "raw-body": "2.3.3",
35         "type-is": "1.6.16"
36       }
37     },
38     "bytes": {
39       "version": "3.0.0",
40       "resolved": "https://registry.npmjs.org/bytes/-/bytes-3.0.0.tgz",
41       "integrity": "sha1-0ygVQE1olpn4Wk6k+odV3ROpYEg="
42     },
43     "content-disposition": {
44       "version": "0.5.2",
45       "resolved": "https://registry.npmjs.org/content-disposition/-/content-disposition-0.5.2.tgz"
46       ,
47       "integrity": "sha1-DPaLud318r55YcOoUXjLhdunjLQ="
48     },
49     "content-type": {
50       "version": "1.0.4",
51       "resolved": "https://registry.npmjs.org/content-type/-/content-type-1.0.4.tgz",
52       "integrity": "sha1-512-hIP3EEPs8tB9AT1L+NUqtwoAps4mk2Zob89MWXMHjHWg9milF/j4osnnQLXBCFBk/tvIG/tUc9mOUJiPBhPXAA=="
53     },
54     "cookie": {
55       "version": "0.3.1",
56       "resolved": "https://registry.npmjs.org/cookie/-/cookie-0.3.1.tgz",
57       "integrity": "sha1-5+Ch+e9DtMi6k1xcWpb0BtFoc7s="
58     },
59     "cookie-signature": {
60       "version": "1.0.6",
61       "resolved": "https://registry.npmjs.org/cookie-signature/-/cookie-signature-1.0.6.tgz",
62       "integrity": "sha1-4wOogrNCzD7oylE6eZmXNNqzriw="
63     },
64     "crc": {
65       "version": "3.4.4",
66       "resolved": "https://registry.npmjs.org/crc/-/crc-3.4.4.tgz",
67       "integrity": "sha1-naHpgOO9RPxck79as9ozeNheRms="
68   },
```

```
68 "debug": {
69   "version": "2.6.9",
70   "resolved": "https://registry.npmjs.org/debug/-/debug-2.6.9.tgz",
71   "integrity": "sha512-bC7ElrdJaJnPbAP+1EotYvqZsb3ec15wi6Bfi6BJTUCNowp6cvspg0jXznRTKDjm/E7AdgFBVeAPVMNcKGsHMA==",
72   "requires": {
73     "ms": "2.0.0"
74   }
75 },
76 "depd": {
77   "version": "1.1.2",
78   "resolved": "https://registry.npmjs.org/depd/-/depd-1.1.2.tgz",
79   "integrity": "sha1-m81S4UwJd2PnSbJ0xDRu0uVgtak="
80 },
81 "destroy": {
82   "version": "1.0.4",
83   "resolved": "https://registry.npmjs.org/destroy/-/destroy-1.0.4.tgz",
84   "integrity": "sha1-14hXRCxEJ5CBmE+N5RiBYJqvYA=="
85 },
86 "ee-first": {
87   "version": "1.1.1",
88   "resolved": "https://registry.npmjs.org/ee-first/-/ee-first-1.1.1.tgz",
89   "integrity": "sha1-WQxhFWsK4vTwJVcyoViyZrxWsh0="
90 },
91 "ejs": {
92   "version": "2.6.1",
93   "resolved": "https://registry.npmjs.org/ejs/-/ejs-2.6.1.tgz",
94   "integrity": "sha512-0xy4A/twfrRCnkhfk8ErDi5DqdAsAqeGxht4xkCUsvhbQNs7E+4jV0CN7+NKIY0aHE72+XvqtBIXzD31ZbXQ=="
95 },
96 "encodeurl": {
97   "version": "1.0.2",
98   "resolved": "https://registry.npmjs.org/encodeurl/-/encodeurl-1.0.2.tgz",
99   "integrity": "sha1-rT/0yG7C0CkyL1oCw6mmBs1bP1k="
100 },
101 "escape-html": {
102   "version": "1.0.3",
103   "resolved": "https://registry.npmjs.org/escape-html/-/escape-html-1.0.3.tgz",
104   "integrity": "sha1-Aljq5NPQwJdN4cFpGI7wBR0dGYg=="
105 },
106 "etag": {
107   "version": "1.8.1",
108   "resolved": "https://registry.npmjs.org/etag/-/etag-1.8.1.tgz",
109   "integrity": "sha1-Qa4u62XvpiJorr/qg6x9eSmbCIC="
110 },
111 "express": {
112   "version": "4.16.3",
113   "resolved": "http://registry.npmjs.org/express/-/express-4.16.3.tgz",
114   "integrity": "sha1-avilAjUNsyRuzEvs9rWjTSL37VM=",
115   "requires": {
116     "accepts": "1.3.5",
117     "array-flatten": "1.1.1",
118     "body-parser": "1.18.2",
119     "content-disposition": "0.5.2",
120     "content-type": "1.0.4",
121     "cookie": "0.3.1",
122     "cookie-signature": "1.0.6",
123     "debug": "2.6.9",
124     "depd": "1.1.2",
125     "encodeurl": "1.0.2",
126     "escape-html": "1.0.3",
127     "etag": "1.8.1",
128     "finalhandler": "1.1.1",
129     "fresh": "0.5.2",
130     "merge-descriptors": "1.0.1",
131     "methods": "1.1.2",
132     "on-finished": "2.3.0",
133     "parseurl": "1.3.2",
134     "path-to-regexp": "0.1.7",
135     "proxy-addr": "2.0.4",
136     "qs": "6.5.1",
137   }
138 }
```

```
137     "range-parser": "1.2.0",
138     "safe-buffer": "5.1.1",
139     "send": "0.16.2",
140     "serve-static": "1.13.2",
141     "setprototypeof": "1.1.0",
142     "statuses": "1.4.0",
143     "type-is": "1.6.16",
144     "utils-merge": "1.0.1",
145     "vary": "1.1.2"
146   },
147   "dependencies": {
148     "body-parser": {
149       "version": "1.18.2",
150       "resolved": "https://registry.npmjs.org/body-parser/-/body-parser-1.18.2.tgz",
151       "integrity": "sha1-h2eKGdhLR9hZuDGZvVm84iKxBFQ=",
152       "requires": {
153         "bytes": "3.0.0",
154         "content-type": "1.0.4",
155         "debug": "2.6.9",
156         "depd": "1.1.2",
157         "http-errors": "1.6.3",
158         "iconv-lite": "0.4.19",
159         "on-finished": "2.3.0",
160         "qs": "6.5.1",
161         "raw-body": "2.3.2",
162         "type-is": "1.6.16"
163       }
164     },
165     "iconv-lite": {
166       "version": "0.4.19",
167       "resolved": "https://registry.npmjs.org/iconv-lite/-/iconv-lite-0.4.19.tgz",
168       "integrity": "sha512-oTZqweIP51xaGPI4uPa56/Pri/480R+mo7SeU+YETByQNhDG55ycFyNLigta9vXhILrxXdmF7ZGhqZIcuN0gJQ=="
169     },
170     "qs": {
171       "version": "6.5.1",
172       "resolved": "https://registry.npmjs.org/qs/-/qs-6.5.1.tgz",
173       "integrity": "sha512-eRzhrN1WSINYCDCbrz796z37LOe3m5tmW7RQf6oBntukAG1nmovJvhnwHHRMAfeoItclm2Hk02WER2aQ/iqs+A=="
174   },
175   "raw-body": {
176     "version": "2.3.2",
177     "resolved": "https://registry.npmjs.org/raw-body/-/raw-body-2.3.2.tgz",
178     "integrity": "sha1-vNYMd9Prk83gBQKVw/N5OJvIj4k=",
179     "requires": {
180       "bytes": "3.0.0",
181       "http-errors": "1.6.2",
182       "iconv-lite": "0.4.19",
183       "unpipe": "1.0.0"
184     },
185     "dependencies": {
186       "depd": {
187         "version": "1.1.1",
188         "resolved": "https://registry.npmjs.org/depd/-/depd-1.1.1.tgz",
189         "integrity": "sha1-V4004cRZ8G+lyif5kfPQbnoxA1k="
190     },
191     "http-errors": {
192       "version": "1.6.2",
193       "resolved": "https://registry.npmjs.org/http-errors/-/http-errors-1.6.2.tgz",
194       "integrity": "sha1-CgAsyFcHGSp+eUb07cERVfYOxzY=",
195       "requires": {
196         "depd": "1.1.1",
197         "inherits": "2.0.3",
198         "setprototypeof": "1.0.3",
199         "statuses": "1.4.0"
200       }
201     },
202     "setprototypeof": {
203       "version": "1.0.3",
```

```
204         "resolved":  
205         "https://registry.npmjs.org/setprototypeof/-/setprototypeof-1.0.3.tgz",  
206         "integrity": "sha1-ZlZ+NwQ+608E2RvWWMDL77VbjgQ="  
207     }  
208 },  
209     "statuses": {  
210         "version": "1.4.0",  
211         "resolved": "https://registry.npmjs.org/statuses/-/statuses-1.4.0.tgz",  
212         "integrity":  
213         "sha512-zhSCtt8v2NDrR1PQpCNtw/heZLtfUDqxBMludqikb/Hbk52LK4nQSwr10u77iopCW5LsyHpuXS0GnEc48mLeew=="  
214     }  
215 },  
216     "express-session": {  
217         "version": "1.15.6",  
218         "resolved":  
219         "https://registry.npmjs.org/express-session/-/express-session-1.15.6.tgz",  
220         "integrity":  
221         "sha512-r0nrHTCYtAMrFwZ0kBzZEa1vtPVrw0dKvGSrKP4dahwBQ1BJpF2/y1Pp4sCD/0kvxV4zzec1yvfmw0B4RMJQA==",  
222         "requires": {  
223             "cookie": "0.3.1",  
224             "cookie-signature": "1.0.6",  
225             "crc": "3.4.4",  
226             "debug": "2.6.9",  
227             "depd": "1.1.2",  
228             "on-headers": "1.0.1",  
229             "parseurl": "1.3.2",  
230             "uid-safe": "2.1.5",  
231             "utils-merge": "1.0.1"  
232         }  
233     },  
234     "filereader": {  
235         "version": "0.10.3",  
236         "resolved": "https://registry.npmjs.org/filereader/-/filereader-0.10.3.tgz",  
237         "integrity": "sha1-x0fUos2PYeVBinwH/hJXpD8KzbE="  
238 },  
239     "finalhandler": {  
240         "version": "1.1.1",  
241         "resolved": "https://registry.npmjs.org/finalhandler/-/finalhandler-1.1.1.tgz",  
242         "integrity":  
243         "sha512-Y1GUDo39ez4aH Aw7MysnUD5JzYX+Waij8I57kO3aEPT1fFRL4sr7mjei97FgnwhAyyzRYmQZaTHb2+9uZ1dPtg==",  
244         "requires": {  
245             "debug": "2.6.9",  
246             "encodeurl": "1.0.2",  
247             "escape-html": "1.0.3",  
248             "on-finished": "2.3.0",  
249             "parseurl": "1.3.2",  
250             "statuses": "1.4.0",  
251             "unpipe": "1.0.0"  
252         }  
253     },  
254     "dependencies": {  
255         "statuses": {  
256             "version": "1.4.0",  
257             "resolved": "https://registry.npmjs.org/statuses/-/statuses-1.4.0.tgz",  
258             "integrity":  
259             "sha512-zhSCtt8v2NDrR1PQpCNtw/heZLtfUDqxBMludqikb/Hbk52LK4nQSwr10u77iopCW5LsyHpuXS0GnEc48mLeew=="  
260         }  
261     },  
262     "forwarded": {  
263         "version": "0.1.2",  
264         "resolved": "https://registry.npmjs.org/forwarded/-/forwarded-0.1.2.tgz",  
265         "integrity": "sha1-mMI9qxFlZXuMBXPozszZGw/xjIQ="  
266 },  
267     "fresh": {  
268         "version": "0.5.2",  
269         "resolved": "https://registry.npmjs.org/fresh/-/fresh-0.5.2.tgz",  
270         "integrity": "sha1-PYyt2Q2XZWn6glqx+OSyOhBWBac="
```

```
267 },
268 "http-errors": {
269   "version": "1.6.3",
270   "resolved": "http://registry.npmjs.org/http-errors/-/http-errors-1.6.3.tgz",
271   "integrity": "sha1-i1VoC7S+KDoLW/TqLjhYC+HZMg0=",
272   "requires": {
273     "depd": "1.1.2",
274     "inherits": "2.0.3",
275     "setprototypeof": "1.1.0",
276     "statuses": "1.5.0"
277   }
278 },
279 "iconv-lite": {
280   "version": "0.4.23",
281   "resolved": "https://registry.npmjs.org/iconv-lite/-/iconv-lite-0.4.23.tgz",
282   "integrity": "sha512-neyTUVFtahjf0mB3dZT77u+800QB89jFdNBkd5P1JgYPbPaia3gXXOVL2fq8VyU2gMMD7SaN7QukTB/pmXYvDA==",
283   "requires": {
284     "safer-buffer": "2.1.2"
285   }
286 },
287 "inherits": {
288   "version": "2.0.3",
289   "resolved": "https://registry.npmjs.org/inherits/-/inherits-2.0.3.tgz",
290   "integrity": "sha1-Yzwsg+PaQqUC9SRmAisA9CCYd4="
291 },
292 "ipaddr.js": {
293   "version": "1.8.0",
294   "resolved": "https://registry.npmjs.org/ipaddr.js/-/ipaddr.js-1.8.0.tgz",
295   "integrity": "sha1-6qM9bd16zo9/b+DJygRA5wZzix4="
296 },
297 "media-typer": {
298   "version": "0.3.0",
299   "resolved": "https://registry.npmjs.org/media-typer/-/media-typer-0.3.0.tgz",
300   "integrity": "sha1-hxDXrwqmJvj/+hzgAWhUUmMlV0g="
301 },
302 "merge-descriptors": {
303   "version": "1.0.1",
304   "resolved": "https://registry.npmjs.org/merge-descriptors/-/merge-descriptors-1.0.1.tgz",
305   "integrity": "sha1-sAqqVW3YtEVoFQ7J0blT8/kMu2E="
306 },
307 "method-override": {
308   "version": "3.0.0",
309   "resolved": "https://registry.npmjs.org/method-override/-/method-override-3.0.0.tgz",
310   "integrity": "sha1-IJ2NNN/mSl9w3kzWB92rcdHpz+HjkxhDJWNDBqSlas+zQdP8wBiJzITPg08M/k2uVvMow7Sk4latndNtt/PHSA==",
311   "requires": {
312     "debug": "3.1.0",
313     "methods": "1.1.2",
314     "parseurl": "1.3.2",
315     "vary": "1.1.2"
316   },
317   "dependencies": {
318     "debug": {
319       "version": "3.1.0",
320       "resolved": "https://registry.npmjs.org/debug/-/debug-3.1.0.tgz",
321       "integrity": "sha1-OX8XqP7/1a9cqkxYw2yXss15f26NKWBpDXQd0/uK/KPqdQhxbPa994hnzjcE2VqQpDslf55723cKPUOGSmMY3g==",
322       "requires": {
323         "ms": "2.0.0"
324       }
325     }
326   }
327 },
328 "methods": {
329   "version": "1.1.2",
330   "resolved": "https://registry.npmjs.org/methods/-/methods-1.1.2.tgz",
331   "integrity": "sha1-VSmklNZUE07cxSZmVoNbD4Ua/O4="
```

```
332 },
333   "mime": {
334     "version": "1.4.1",
335     "resolved": "https://registry.npmjs.org/mime/-/mime-1.4.1.tgz",
336     "integrity":
337       "sha512-KI1+qOZu5DcW6wayYHSzR/tXKCDC5Om4s1z2QJjDULzLcmf3DvzS7oluY4HCTrc+9FiKmWUgeNLg7W3uIQvxtQ=="
338   },
339   "mime-db": {
340     "version": "1.36.0",
341     "resolved": "https://registry.npmjs.org/mime-db/-/mime-db-1.36.0.tgz",
342     "integrity":
343       "sha512-L+xvyD9MkoYMXb1jAmzI/lWYAxAMCPvIBSWur0PZ5nOf5euahRLVqH//FKW9mWp21kqUgYixPgkzfMUfi4zVDw=="
344   },
345   "mime-types": {
346     "version": "2.1.20",
347     "resolved": "https://registry.npmjs.org/mime-types/-/mime-types-2.1.20.tgz",
348     "integrity":
349       "sha512-HrkrPaP9vGuWbLK1B1FfgAkbqNjIuy4eHlIYnFi7kamZyLLrGlo2mpcx0bBmNpKqBtYtAfGbodDddIgddSJC2A==",
350     "requires": {
351       "mime-db": "1.36.0"
352     }
353   },
354   "moment": {
355     "version": "2.24.0",
356     "resolved": "https://registry.npmjs.org/moment/-/moment-2.24.0.tgz",
357     "integrity":
358       "sha512-bv7f+6l2QigeBBZSM/6yTNq4P2fNpSWj/0e7jQcy87A8e7o2nAfP/34/2ky5Vw4B9$446EtIhodAzkFCcR4dQg=="
359   },
360   "ms": {
361     "version": "2.0.0",
362     "resolved": "https://registry.npmjs.org/ms/-/ms-2.0.0.tgz",
363     "integrity": "sha1-VgiurfwAvmwpAd9fmGF4jeDVl8g="
364   },
365   "negotiator": {
366     "version": "0.6.1",
367     "resolved": "https://registry.npmjs.org/negotiator/-/negotiator-0.6.1.tgz",
368     "integrity": "sha1-KzJxhOizIQExeyhWP7XnECrNDKk="
369   },
370   "on-finished": {
371     "version": "2.3.0",
372     "resolved": "https://registry.npmjs.org/on-finished/-/on-finished-2.3.0.tgz",
373     "integrity": "sha1-IPEzzIGwg811M3mSoWlxqi2QaUc=",
374     "requires": {
375       "ee-first": "1.1.1"
376     }
377   },
378   "on-headers": {
379     "version": "1.0.1",
380     "resolved": "https://registry.npmjs.org/on-headers/-/on-headers-1.0.1.tgz",
381     "integrity": "sha1-ko9dD0cNSTQmUepn1LCFFBAGk/c="
382   },
383   "parseurl": {
384     "version": "1.3.2",
385     "resolved": "https://registry.npmjs.org/parseurl/-/parseurl-1.3.2.tgz",
386     "integrity": "sha1-CidTtiZMR1GDBViUyYs3I3mW/M="
387   },
388   "path-to-regexp": {
389     "version": "0.1.7",
390     "resolved": "https://registry.npmjs.org/path-to-regexp/-/path-to-regexp-0.1.7.tgz",
391     "integrity": "sha1-32BBeABfUi8V60SQ5yR6G/qmf4w="
392   },
393   "proxy-addr": {
394     "version": "2.0.4",
395     "resolved": "https://registry.npmjs.org/proxy-addr/-/proxy-addr-2.0.4.tgz",
396     "integrity":
397       "sha512-5erio2h9jp5CHGwcybmxmVqHmnCBZeewlfJ0pex+UW7Qny7OOZXTth56TGNyBizkgiOwhJtMKrVzDTeKcySzWa==",
398     "requires": {
```

```
394     "forwarded": "0.1.2",
395     "ipaddr.js": "1.8.0"
396   },
397   "qs": {
398     "version": "6.5.2",
399     "resolved": "https://registry.npmjs.org/qs/-/qs-6.5.2.tgz",
400     "integrity":
401       "sha512-N5ZAX4/LxJmF+7wN74pUD6qAh9/wnvdQcj9TZjevvXzSUo7bfmw91saqMjzGS2xq91/odN2dW/WO17qQHNDGA=="
402   },
403   "random-bytes": {
404     "version": "1.0.0",
405     "resolved": "https://registry.npmjs.org/random-bytes/-/random-bytes-1.0.0.tgz",
406     "integrity": "sha1-T2ih3Arli9P7lYSMMJDNt11kNgs="
407   },
408   "range-parser": {
409     "version": "1.2.0",
410     "resolved": "https://registry.npmjs.org/range-parser/-/range-parser-1.2.0.tgz",
411     "integrity": "sha1-9JvmtIeJTxA3M1KMi9hEJLgDV4="
412   },
413   "raw-body": {
414     "version": "2.3.3",
415     "resolved": "https://registry.npmjs.org/raw-body/-/raw-body-2.3.3.tgz",
416     "integrity":
417       "sha512-9esiElv1BrZoI3rCDuOuKCBRbuApGGaDPQfjSf1Gxdy4oyzqghxu6k1EkkVIvBje+FF0BX9coEv8KqW6X/7njw==",
418     "requires": {
419       "bytes": "3.0.0",
420       "http-errors": "1.6.3",
421       "iconv-lite": "0.4.23",
422       "unpipe": "1.0.0"
423     }
424   },
425   "safe-buffer": {
426     "version": "5.1.1",
427     "resolved": "https://registry.npmjs.org/safe-buffer/-/safe-buffer-5.1.1.tgz",
428     "integrity":
429       "sha512-kKvNJn6Mm93gAczWVJg7wH+wGYWNrDHdWvpUmHyEsgCtIwoo3bqPtV4tR5tuPaUhTOo/kvhVwd8XwwOllGYkbg=="
430   },
431   "safer-buffer": {
432     "version": "2.1.2",
433     "resolved": "https://registry.npmjs.org/safer-buffer/-/safer-buffer-2.1.2.tgz",
434     "integrity":
435       "sha512-YZo3K82SD7Riyi0E1EQPojLz7kpepnSQI9IyPbHHg1XXXevb5dJI7tpyN2ADxGcQbHG7vcyRHk0cbwqcQriUtg=="
436   },
437   "send": {
438     "version": "0.16.2",
439     "resolved": "https://registry.npmjs.org/send/-/send-0.16.2.tgz",
440     "integrity":
441       "sha512-E64YFPUsxFHEFBvpbbjr44NCLtI1AohxQ8ZSiJjQLskAdKuriYEP6VyGEsRDH8ScozGpkaX1BGvhanqCwkcEZw==",
442     "requires": {
443       "debug": "2.6.9",
444       "depd": "1.1.2",
445       "destroy": "1.0.4",
446       "encodeurl": "1.0.2",
447       "escape-html": "1.0.3",
448       "etag": "1.8.1",
449       "fresh": "0.5.2",
450       "http-errors": "1.6.3",
451       "mime": "1.4.1",
452       "ms": "2.0.0",
453       "on-finished": "2.3.0",
454       "range-parser": "1.2.0",
455       "statuses": "1.4.0"
456     },
457     "dependencies": {
458       "statuses": {
459         "version": "1.4.0",
460         "resolved": "https://registry.npmjs.org/statuses/-/statuses-1.4.0.tgz",
461       }
462     }
463   }
464 }
```

```
457     "integrity":  
458     "sha512-zhSCtt8v2NDrRlPQpCNtw/heZLtfUDqxBMludqikb/Hbk52LK4nQSwr10u77iopCW5Ls  
459     }  
460   },  
461   "serve-static": {  
462     "version": "1.13.2",  
463     "resolved": "https://registry.npmjs.org/serve-static/-/serve-static-1.13.2.tgz",  
464     "integrity":  
465     "sha512-p/tdJrO4U387R9oMjb1oj7qSMaMfmOyd4j9hOFoxZe2baQszgHcSWjuya/CiT5kgZZKRudHN  
466     OA0pYX018rQ5nw==",  
467     "requires": {  
468       "encodeurl": "1.0.2",  
469       "escape-html": "1.0.3",  
470       "parseurl": "1.3.2",  
471       "send": "0.16.2"  
472     }  
473   },  
474   "setprototypeof": {  
475     "version": "1.1.0",  
476     "resolved": "https://registry.npmjs.org/setprototypeof/-/setprototypeof-1.1.0.tgz",  
477     "integrity":  
478     "sha512-BvE/TwpZX4FXExxOxZyRGQQv651MSwmWKZGqvmPcRIjDqWub67kTKuIMx43cZZrs/cBBzwBc  
479     NDWoFxt2XEFlpQ=="  
480   },  
481   "statuses": {  
482     "version": "1.5.0",  
483     "resolved": "https://registry.npmjs.org/statuses/-/statuses-1.5.0.tgz",  
484     "integrity": "sha1-Fhx9rBd2Wf2YEFQ3cfqZOBR4Yow="  
485   },  
486   "type-is": {  
487     "version": "1.6.16",  
488     "resolved": "https://registry.npmjs.org/type-is/-/type-is-1.6.16.tgz",  
489     "integrity":  
490     "sha512-HRkVv/5qY2G6I8iab9cI7v1b0Idhm94dVjQCPFE1W9W+3GeDOSHmy2EBYe4VTApuzolPcmgF  
491     TN3ftVJRKR2J9Q==",  
492     "requires": {  
493       "media-typer": "0.3.0",  
494       "mime-types": "2.1.20"  
495     }  
496   },  
497   "uid-safe": {  
498     "version": "2.1.5",  
499     "resolved": "https://registry.npmjs.org/uid-safe/-/uid-safe-2.1.5.tgz",  
500     "integrity":  
501     "sha512-KPHm4VL5dDXKz01UuEd88Df+KzynaohSL9fBh096KWAxSKZQDI2uBrVqtvRM4rwrIrRRKsdL  
502     NML/lnaaVSRioA==",  
503     "requires": {  
504       "random-bytes": "1.0.0"  
505     }  
506   },  
507   "uniqid": {  
508     "version": "5.0.3",  
509     "resolved": "https://registry.npmjs.org/uniqid/-/uniqid-5.0.3.tgz",  
510     "integrity":  
511     "sha512-R2qx3X/LYWSdGRalui04dYrPXAJACTqyUjuyXHoJLBuOIfmMcnyOyY2d6Y4clZcIz5lK6Zai  
512     OZzmm0cPfsIqzQ=="  
513   },  
514   "unpipe": {  
515     "version": "1.0.0",  
516     "resolved": "https://registry.npmjs.org/unpipe/-/unpipe-1.0.0.tgz",  
517     "integrity": "sha1-sr906FFKrmFltIF4KdIBLvSZB0w="  
518   },  
519   "utils-merge": {  
520     "version": "1.0.1",  
521     "resolved": "https://registry.npmjs.org/utils-merge/-/utils-merge-1.0.1.tgz",  
522     "integrity": "sha1-n5VxD1CiZ5R7LMwSR0HBaoQn5xM="  
523   },  
524   "vary": {  
525     "version": "1.1.2",  
526     "resolved": "https://registry.npmjs.org/vary/-/vary-1.1.2.tgz",  
527   }
```

```
517     "integrity": "sha1-IpnwLG3tMNS1lhsLn3RSShj2NPw="
518 }
519 }
520 }
521 }
```

Package.json

```
1  {
2      "name": "kwcrmCourseworkProject",
3      "version": "1.0.0",
4      "description": "",
5      "main": "index.js",
6      "scripts": {
7          "test": "echo \\\"Error: no test specified\\\" && exit 1",
8          "start": "node index.js"
9      },
10     "author": "kieran williams",
11     "license": "ISC",
12     "dependencies": {
13         "body-parser": "^1.18.3",
14         "ejs": "^2.6.1",
15         "express": "^4.16.3",
16         "express-session": "^1.15.6",
17         "filereader": "^0.10.3",
18         "method-override": "^3.0.0",
19         "moment": "^2.24.0",
20         "uniqid": "^5.0.3"
21     }
22 }
23
```

Classes

Directory

Contents

Analytics.js

Kieran Marcus Williams

```

1  const DateHandler = require('./DateHandler');
2  const ConversionRate = require('./ConversionRate');
3  const GenerateDataPoints = require('./GenerateDataPoints');
4
5
6  const _clientsGainedFilePath = Symbol("clientsGainedFilePath");
7  const _clientsLostFilePath = Symbol("clientsLostFilePath");
8  const _leadsLostFilePath = Symbol("leadsLostFilePath");
9
10 // The class that will be used to handle all back-end tasks of the Analyticals page.
11 class Analyticals {
12
13     constructor(clientsGainedCSVHandlerObject, clientsLostCSVHandlerObject,
14                 leadsLostCSVHandlerObject) {
15
16         this[_clientsGainedFilePath] = clientsGainedCSVHandlerObject;
17         this[_clientsLostFilePath] = clientsLostCSVHandlerObject;
18         this[_leadsLostFilePath] = leadsLostCSVHandlerObject;
19     }
20
21     // this static method will return an object for the conversion rate of
22     // the month as both a percentage and as a ratio in it's lowest form
23     GetConversionRateForTheMonth () {
24
25         let clientsGainedFile = this[_clientsGainedFilePath];
26         let leadsLostFile = this[_leadsLostFilePath];
27
28         let currentDate = DateHandler.getCurrentDate();
29         let numberOfClientsGainedThisMonth;
30         let numberOfLeadsLostThisMonth;
31
32         // to find data for clients gained and leads lost this month I will have to
33         // find all the matches
34         // in the second index against the current date in standard time with the
35         // last 18 characters truncated
36         // as in the 2nd index of all the analytics files there is the month and
37         // year stored as yyyy-mm
38
39         currentDate = currentDate.split(""); // turns the date into an array of
40         // characters
41         currentDate.splice(7,18); // truncates the last 18 characters from the array
42         currentDate = currentDate.join(""); // joins the array together into a
43         // string and stores the value into the variable
44
45         numberOfClientsGainedThisMonth = clientsGainedFile.countMatches([2],
46                           currentDate);
47         numberOfLeadsLostThisMonth = leadsLostFile.countMatches([2], currentDate);
48
49         return ConversionRate.conversionRate(numberOfClientsGainedThisMonth,
50                                               numberOfLeadsLostThisMonth);
51     }
52
53     // will return the data points and labels for both clients gained and lost
54     // done for the past 14 days
55
56     getDataPointsAndLabelsForThePast14Days () {
57
58         let clientsLost = new GenerateDataPoints("clientsLost",
59                                         this[_clientsGainedFilePath],
60                                         this[_clientsLostFilePath],
61                                         this[_leadsLostFilePath]);
62
63         let clientsGained = new GenerateDataPoints("clientsGained",
64                                         this[_clientsGainedFilePath],
65                                         this[_clientsLostFilePath],
66                                         this[_leadsLostFilePath]);
67
68         // returns an object containing data for the clients gained and lost with
69         // their relative
70         // data points and labels to go along with both

```

```

65     return {
66
67         clientsGained: {
68             dataPoints: clientsGained.getPast14DaysOfDataPoints().dataPoints,
69             labels: clientsGained.getPast14DaysOfDataPoints().labels
70         },
71
72         clientsLost: {
73             dataPoints: clientsLost.getPast14DaysOfDataPoints().dataPoints,
74             labels: clientsLost.getPast14DaysOfDataPoints().labels
75         }
76     }
77
78 }
79
80
81 // will return the data points and labels for both clients gained and lost
82 // done for the past 12 weeks
83
84 getDataPointsAndLabelsForThePast12Weeks () {
85
86     let clientsLost = new GenerateDataPoints("clientsLost",
87         this[_clientsGainedFilePath],
88         this[_clientsLostFilePath],
89         this[_leadsLostFilePath]);
90
91     let clientsGained = new GenerateDataPoints("clientsGained",
92         this[_clientsGainedFilePath],
93         this[_clientsLostFilePath],
94         this[_leadsLostFilePath]);
95
96     // returns an object containing data for the clients gained and lost with
97     // their relative
98     // data points and labels to go along with both
99
100    return {
101        clientsGained: {
102            dataPoints: clientsGained.getDataPointsForLast12Weeks().dataPoints,
103            labels: clientsGained.getDataPointsForLast12Weeks().labels
104        },
105
106        clientsLost: {
107            dataPoints: clientsLost.getDataPointsForLast12Weeks().dataPoints,
108            labels: clientsLost.getDataPointsForLast12Weeks().labels
109        }
110    }
111
112 }
113
114 // will return the data points and labels for both clients gained and lost
115 // done for the past 12 months
116
117 getDataPointsAndLabelsForThePast12Months () {
118
119     let clientsLost = new GenerateDataPoints("clientsLost",
120         this[_clientsGainedFilePath],
121         this[_clientsLostFilePath],
122         this[_leadsLostFilePath]);
123
124     let clientsGained = new GenerateDataPoints("clientsGained",
125         this[_clientsGainedFilePath],
126         this[_clientsLostFilePath],
127         this[_leadsLostFilePath]);
128
129     // returns an object containing data for the clients gained and lost with
130     // their relative
131     // data points and labels to go along with both
132
133     return {
134         clientsGained: {
135             dataPoints: clientsGained.getDataPointsForLast12Months().dataPoints,

```

```
136         labels: clientsGained.getDataPointsForLast12Months().labels
137     },
138
139     clientsLost: {
140         dataPoints: clientsLost.getDataPointsForLast12Months().dataPoints,
141         labels: clientsLost.getDataPointsForLast12Months().labels
142     }
143 }
144
145 }
146
147 }
148
149
150 module.exports = Analyticals;
151
```

ConversionRate.js

Kieran Marcus Williams

```

1 // declaring the symbols so that I can imitate private variables and methods
2 // will prevent the methods and variables appearing as suggestions when I type
3 // 'Analytics.'
4 // will only show when I type 'Analytics[_]' this will make the class easier to use
5 // else where when
6 // an instance of this class is made and help ensure that everything works as intended
7 const _greatestCommonDivisor = Symbol('greatestCommonDivisor');
8 const _getConversionRateRatioAsString = Symbol('getConversionRateRatioAsString');
9 const _getConversionRatePercentageAsString =
10 Symbol('getConversionRatePercentageAsString');

11 // The class that will be used to create a conversion rate
12 class ConversionRate {
13
14     // uses A RECURSIVE ALGORITHM to find the greatest common divisor
15     // uses the basis of the Euclidean algorithm
16     static [_greatestCommonDivisor] (valueOne, valueTwo) {
17         let remainder;
18         if (!valueTwo) { // conditional gets met when valueTwo ('remainder' after
19             first call) == 0 (terminating condition)
20             return valueOne; // valueOne is now our greatest common divisor
21         }
22         remainder = valueOne % valueTwo; // calculates the remainder
23         // calls its self again with the valueOne parameter passing in the valueTwo
24         // as an argument
25         // and in the place of the valueTwo parameter the remainder is passed as an
26         // argument
27         return this[_greatestCommonDivisor](valueTwo, remainder);
28     };
29
30
31     // method outputs a ratio as type string of the conversion rate
32     static [_getConversionRateRatioAsString] (numberOfClientsGained,
33     numberOfLeadsLost) {
34
35         // if both the numberOfClientsGained and the numberOfLeadsLost is == 0,
36         // output a dash so that the output is not 'NaN : NaN'
37         if (numberOfClientsGained === 0 && numberOfLeadsLost === 0) {
38             return '-'
39         }
40         // stores the greatest common divisor in the gcd variable so that the ratio
41         // can be
42         // shown in it's lowest form
43         let gcd = this[_greatestCommonDivisor](numberOfClientsGained,
44         numberOfLeadsLost);
45         // the ratio values are calculated by dividing the inputs by the greatest
46         // common divisor
47         // the output is a string such as "2:3" if the inputs for example were 20
48         // and 30
49         return `${numberOfClientsGained/gcd} : ${numberOfLeadsLost/gcd}`;
50     }
51
52     // method outputs a percentage value as type string of the conversion rate
53     static [_getConversionRatePercentageAsString] (numberOfClientsGained,
54     numberOfLeadsLost) {
55         // if both the numberOfClientsGained and the numberOfLeadsLost is == 0,
56         // output a dash so that the output is not 'NaN%'
57         if (numberOfClientsGained === 0 && numberOfLeadsLost === 0) {
58             return '-'
59         }
60         // total is calculated to get a divisor
61         let total = numberOfClientsGained + numberOfLeadsLost;
62         // percentage is calculated by dividing the numberOfClients gained by the
63         // total variable
64         // then multiplying by 100 to get the number as a value between 0 and 100
65         // Math.floor() method is used to truncate the value or any decimal values
66         // it may have
67         let percentage = Math.floor((numberOfClientsGained / total) * 100);
68         // the return is the percentage
69         return `${percentage}%`;
70     }

```

```
60
61
62
63
64 // public static method that can be called anywhere without instantiating the
65 // class
66 // used to get an object containing the conversion rate as a ratio and as a
67 // percentage. arguments passed should be of type number
68 static conversionRate (numberOfClientsGained, numberOfLeadsLost) {
69
70     let conversionRateRatio =
71     this[_getConversionRateRatioAsString](numberOfClientsGained,
72     numberOfLeadsLost);
73     let conversionRatePercentage =
74     this[_getConversionRatePercentageAsString](numberOfClientsGained,
75     numberOfLeadsLost);
76
77     return {
78         ratio: conversionRateRatio,
79         percentage: conversionRatePercentage
80     }
81 }
82
83 module.exports = ConversionRate;
```

CSVHandler.js

```

1 "use strict";
2
3 const
4   fs = require("fs"),
5   path = require('path'),
6   uniqueID = require('uniqid'),
7   Sort = require("./Sort"),
8   Search = require("./Search"),
9   DateHandler = require("./DateHandler");
10
11
12 // way of imitating private methods and variables as the JS does not have a
13 // a way to define them yet. Done by differentiating the way
14 // they are accessed. It isn't, accessible through the object model
15 // of the class like you would expect the method or variable to be.
16 // This is done so that it is clear what is meant to be accessible and
17 // what is not, from outside the class. Even tho the IDE won't
18 // suggest the methods or variables, it is still possible to
19 // access them. And so, this is not perfect but better than nothing.
20
21 const _csvToTxt = Symbol('csvToTxt');
22 const _txtToCsv = Symbol('txtToCsv');
23 const _csvToArray = Symbol('txtInCsvFormatToArray');
24 const _FILE_PATH = Symbol('filePath');
25 const _rewriteDataToFileWithChanges = Symbol('rewriteDataToFileWithChanges');
26 const _generateID = Symbol('generateID');
27 const _makeChangesToRecord = Symbol('makeChangesToRecord');
28 const _getLastRecordItem = Symbol('_getLastRecordItem');
29
30
31 class CSVHandler {
32
33   constructor(filePath) {
34     // private property of the class
35     this[_FILE_PATH] = filePath;
36   }
37
38   // these functions are used to change the
39   // file from a csv file to a txt file or vice versa
40   // these are private methods
41   [_csvToTxt]() {
42     return path.basename(this[_FILE_PATH], '.txt');
43   }
44
45   [_txtToCsv]() {
46     return path.basename(this[_FILE_PATH], '.csv');
47   }
48
49   // will put a txt file that is in a csv format into a 2d array and return it
50   [_csvToArray]() {
51     let file = fs.readFileSync(this[_FILE_PATH], 'utf8');
52     let recordsArray = file.split("\n");
53     let csvInArray = [];
54
55     // there is an empty array element at the end of each record
56     // in this loop every record has the empty element removed and the rest of the
57     // array gets put back into the csvInArray array.
58     for (let i = 0; i < recordsArray.length; i++) {
59       let singleRecordArray = recordsArray[i].split(",");
60       singleRecordArray.pop(); // gets rid of the empty last element in the
61       // array
62       csvInArray.push(singleRecordArray);
63     }
64     return csvInArray;
65   }
66
67   // returns all records stored in the file
68   getAllRecords () {
69     let arrayOfRecords = this[_csvToArray]();
70     arrayOfRecords.pop();
71     return arrayOfRecords;
72   }

```

```

73
74
75 // used to find a record by it's id
76 // requires only the id to retrieve the record
77 selectRecordByID(id) {
78     let array = this[_csvToArray]();
79     array.pop();
80
81     let recordFoundByID = Search.binarySearch2D(array, [0], id); // Stores the
82     // record found.
83
84     // function done on recordFoundByID in the return is done to flatten the array
85     // Output would be a multi dimensional array otherwise containing just one
86     // record.
87     return [].concat.apply([], recordFoundByID)
88 }
89
90 // searches for a record to return based off of a search value
91 // will check all columns entered to see if there is a match in any of them
92 // columnsToSearch parameter should be an array of numbers that represent the
93 // index's of the columns to be searched
94 // the search value should be a string
95 selectRecordByField(columnsToSearch, searchValue) {
96     let arrayOfRecords = this[_csvToArray]();
97
98     arrayOfRecords.splice(arrayOfRecords.length - 1, 1);
99     return Search.binarySearch2D(arrayOfRecords, columnsToSearch, searchValue)
100 }
101
102 // will return a number for how many matches were found in that column for the
103 // searchValue
104 // columnsToSearch parameter should be an array of numbers that represent the
105 // index's of the columns
106 // to be searched. The search value should be a string.
107 countMatches (columnsToSearched, searchValue) {
108     return this.selectRecordByField(columnsToSearched, searchValue).length;
109 }
110
111 // returns only the records found that match the searchValue
112 // unlike the selectRecordByField which returns an array of objects
113 getRecordsOnlyByField (columnsToSearch, searchValue) {
114     let arrayOfRecordsAndIndexesObj = this.selectRecordByField(columnsToSearch,
115     searchValue);
116     let arrayOfRecordsOnly = [];
117     for (let i = 0; i < arrayOfRecordsAndIndexesObj.length; i++) {
118         arrayOfRecordsOnly.push(arrayOfRecordsAndIndexesObj[i].record)
119     }
120     return arrayOfRecordsOnly;
121 }
122
123 // The generateUniqueID variable will make this a polymorphic method as it will
124 // run differently dependent on
125 // the argument that gets passed for it.
126 // the data argument should be an array of all the data contained in a record
127 addRecord(data, generateUniqueID, noTimeInTimeStamp) { // no time in time stamp
128     allows the user to make a time stamp with no time elements in it
129
130     let arrayOfRecords = this[_csvToArray]();
131     let arrayHolder = [];
132     let currentDateTime;
133
134
135     // if there is no argument passed in the generateUniqueID parameter
136     // the it's default value will be set to undefined which would be
137     // interpreted as
138     // false by the if statements. Because this method is more likely to be used
139     // to create a unique id when a record is being added, it makes sense for it
140     // to be true when
141     // the user is creating a record and leaves out that argument. Also done to
142     // aid readability as
143     // I could have named it doNotGenerateId but parameters aren't typically
144     // named like that

```

```

134
135     if (generateUniqueID === undefined) {
136         generateUniqueID = true;
137     }
138
139     if (generateUniqueID) {
140         // generate the unique id, has to be declared down here so it does not
141         // interfere with the UniqueIdGenerator class when it accesses the
142         // addRecord method
143         const ID = this[_generateID](); // generates a unique id value
144
145         if (noTimeInTimeStamp === true) { // sets appropriate date time stamp
146             dependent on argument passed in
147             currentDateTime = DateHandler.getCurrentDate();
148         } else {
149             currentDateTime = DateHandler.getCurrentDateTime();
150         }
151
152         let recordWithDateStampAndID = [ID, currentDateTime];
153         // add record
154         data.forEach((field) => {
155             recordWithDateStampAndID.push(field);
156         });
157         // console.log(recordWithDateStampAndID);
158         // removes the empty last element in the array caused by the empty new
159         // line
160         // .pop() resulted in the entire array got cleared for some reason so
161         // this is the
162         // work around
163         arrayOfRecords.splice(arrayOfRecords.length - 1);
164         arrayOfRecords.push(recordWithDateStampAndID);
165         // console.log( arrayOfRecords );
166         this[_rewriteDataToFileWithChanges](arrayOfRecords);
167     }
168
169     // used by the unique id generator private method
170     else {
171         arrayOfRecords.pop();
172         arrayHolder.push(data);
173         // takes the 2D array 'arrayOfRecords' and appends the new record to the
174         // end of it
175         arrayOfRecords.push(arrayHolder);
176         // rewrite the array of records with the appended record to the file
177         this[_rewriteDataToFileWithChanges](arrayOfRecords);
178     }
179
180     deleteRecord(id) {
181         let arrayOfRecords = Sort.bubbleSort2D(this[_csvToArray](), 0);
182         arrayOfRecords.splice(arrayOfRecords.length - 1);
183         //exception handling so program doesn't crash if the delete was not possible
184         try{
185             arrayOfRecords.splice(Search.binarySearch2D(arrayOfRecords, [0],
186             id)[0].index, 1); // deletes
187         }
188         catch(e) {
189             console.error(`record was not able to be deleted: check record with that
190             id ${id} exists\n`);
191             console.error(e)
192         }
193         // arrayOfRecords is parsed back into the file without
194         // the record with the id passed as argument
195         // unless the record was not able to be deleted, then all
196         // records will get parsed back into the original csv file
197         this[_rewriteDataToFileWithChanges](arrayOfRecords);
198     }
199
200     // arrayOfEdits format: [[ "col that needs to be edited", "new data"], ... ,

```

```

["col that needs to be edited", "new data"]]
201 // arrayOfEdits = [["1", "james"], ["2", "doug"]];
202 // example of method calling code: users.editRecord("32", [[{"id": "1", "name": "james"}, {"id": "2", "name": "doug"}]]);
203 // id should be of string type
204 editRecord(id, arrayOfEdits) {
205     // file gets turned to 2d array ad then sorted by the id column
206     let arrayOfRecords = Sort.bubbleSort2D(this[_csvToArray](), 0);
207     arrayOfRecords.splice(arrayOfRecords.length - 1); // removes last element in
208     // array because it is empty
209     // find record by id and returns a javascript object: {record: [record
210     // data], index: index record is in 2d array}
211     let recordObject = Search.binarySearch2D(arrayOfRecords, [0], id)[0];
212     // puts the object values into variables
213     let index = recordObject.index;
214     let record = recordObject.record;
215
216     // calls the makeChangesToRecord private method and then
217     // stores that new record in the place of the old record
218     arrayOfRecords[index] = this[_makeChangesToRecord](record, arrayOfEdits);
219     // rewrites the 2D array to a the csv file.
220     this[_rewriteDataToFileWithChanges](arrayOfRecords);
221 }
222
223 // method that will return the last record in the file
224 [_getLastRecordItem]() {
225     let arrayOfRecords = this[_csvToArray]();
226     let arrLength = arrayOfRecords.length;
227     // to access last record, the element has to be length - 2 because
228     // arrays start from 0 and then the other -1 comes form the fact that it
229     // counts the new line.
230     return arrayOfRecords[arrLength - 2];
231 }
232
233 // takes a 2d array as a argument, will write the 2D Array to the file.
234 // works by deleting all data contained in the file and then writing all the
235 // data in the
236 // 2D array to the text file.
237 [_rewriteDataToFileWithChanges](array2D) {
238
239     // remove all data stored in the file
240     // call back to carry out further tasks after the data has been erased
241     fs.writeFile(this[_FILE_PATH], '', (err) => {
242         if (err) {
243             console.log(err);
244         }
245         else {
246             // rewrite data to file
247             let record = [];
248             //iterates through the length of the array (number of records in the
249             //array)
250             for (let i = 0; i < array2D.length; i++) {
251                 record = array2D[i].join(","); // joins the record into a
252                 // string separated with a comma
253                 // writes a record to the file in format of:
254                 // 'comma separated record data' + "," + new line
255                 // call back to output that the data has been re-written
256                 // successfully or an error
257                 // message outlining the error
258                 fs.appendFileSync(this[_FILE_PATH], `${record}\n`);
259             }
260         }
261     });
262
263     deleteAllRecords () {
264         fs.writeFile(this[_FILE_PATH], '', (err) => {
265             console.log(err);

```

```

265     });
266 }
267
268 // column to sort by should be a number of the index of the field that will get
269 // sorted
270 // ascending should be set to true or false depending you want the 2d array to
271 // be in
272 // ascending order: true or descending order: false
273 // if the ascending parameter has no value passed as argument, it will sort it
274 // in ascending order
275 // as that is how I defined the method in the sort class.
276 sortBy (columnToSortBy, ascending) {
277     let sortedArray = Sort.bubbleSort2D(this[_csvToArray](), columnToSortBy,
278         ascending);
279     sortedArray.pop(); // gets rid of the empty element at the end of the array
280     return sortedArray;
281 }
282
283
284 [_generateID]() {
285     // could use this but there is no point of taking up extra server space so I
286     // am using a trusted unique Id generator package
287     // let idFile = new CSVHandler(_dirname + "/dataStoreFiles/IDs.csv"); // create an instance of the CSVHandler class
288     //         // let lastID = idFile.getLastRecordItem()[0]; // get last id
289     //         // number stored in the csv file
290     //         // let newID = parseInt(lastID) + 1; // set id to that value + 1
291     //         // // if the new id comes back as not a number due to there being
292     //         // no number returned from the file
293     //         // // then make new id = 1;
294     //         // if (isNaN(newID)) {
295     //             // newID = 1;
296     //         // } // only time this module wont work is when the IDs.csv file
297     //         // has no lines in it at all
298     //         // // to fix - press the return key in the csv file to add a
299     //         // blank line
300     //         // idFile.addRecord(newID, false); // add id just generated to
301     //         // the csv file
302     //         // return newID.toString();
303     return uniqueID();
304 }
305
306
307 // this private method makes changes to a single record
308 // record should be an array of a single record
309 // arrayOfEdits should be a 2D array containing arrays of changes
310 // arrayOfEdits format: [[["col that needs to be edited", "new data"], ... ,
311 // ["col that needs to be edited", "new data"]]
312 // example data: record = ["101","kieran","williams"]; edits = [[["1",
313 // "james"], ["2", "doug"]]];
314
315 [_makeChangesToRecord](record, arrayOfEdits) {
316     //initialize the variables
317     let col = "";
318     let newData = "";
319
320     arrayOfEdits.forEach((edit) => {
321         col = parseInt(edit[0]);
322         // makes sure that the column is not 0 because if it is, it will be
323         // possible
324         // to change the id field of a record which is something that should not
325         // ever happen
326         if (col !== 0) {
327             newData = edit[1];
328             record[col] = newData;
329         }
330     });
331     return record;
332 }

```

```
323
324 }
325
326 module.exports = CSVHandler;
327
```

DateHandler.js

```

1  const moment = require('moment');
2
3  const _ddmmyyyyToStandardDateTimeCheck = Symbol('ddmmyyyyToStandardDateTimeCheck');
4
5  class DateHandler {
6
7      // creates a time stamp from a date and a time
8      // returns a time stamp that moment.js or the
9      // Date class js has can work with. arguments to be type string
10     // date => dd/mm/yyyy
11     // time is optional, and the seconds value is also optional
12     // time => hh:mm:ss || hh:mm
13     static createTimeStamp (date, time) {
14         let day, month, year, hour, minute, second; // declares variables
15         if (time === undefined) {time = "00:00:00"} // if time is passed as an
16         argument, time is set to "00:00:00"
17         if (time.length === 5) {time = `${time}:00`} // if time is only hh:mm => the
18         ss is set to 00
19         date = date.split("/"); // splits the date string up into a string array
20         ["dd", "mm", "yyyy"]
21         time = time.split(":"); // splits the time string up into a string array
22         ["hh", "mm", "ss"]
23         day = date[0];
24         month = date[1];
25         year = date[2];
26         hour = time[0];
27         minute = time[1];
28         second = time[2];
29         return `${year}-${month}-${day}T${hour}:${minute}:${second}+00:00`;
30     }
31
32     // uses the moment library to get the current date and time
33     // outputs a string in the standard date time stamp
34     static getCurrentDateTime () {
35         return moment().format();
36     }
37
38     // uses the moment library to get the current date and time
39     // outputs a string in the standard date time stamp and then
40     // turns the dateTimeStamp into the standard date time stamp but with
41     // only the date information and the time data will all be set to 0
42     static getCurrentDate() {
43         // returns the result of the current time stamp turned into dd/mm/yyyy,
44         // turned into
45         // the standard date time stamp but with not time information
46         return
47             this.createTimeStamp(this.standardDateTo_Date_DDMMYYYY(this.getCurrentDateTime
48                 ()));
49     }
50
51     // pass a date time stamp as argument format: YYYY-MM-DD'T'HH:MM:SS+MS:mS
52     // eg: '2019-02-02T22:52:49+00:00'
53     // will output the date as dd/mm/yyyy
54     static standardDateTo_Date_DDMMYYYY(dateTimeStamp) {
55         // could use moment(dateTimeStamp).format("DD/MM/YYYY");
56         // but will use my own algorithm to extract the date instead
57
58         // splits argument into an array containing the date at index 0 & the time
59         // at index 1
60         // [0] at the end gets us just the date element, split the date up into an
61         // array ["yyyy", "mm", "dd"]
62         // could use .reverse and then .join("/") which would give me "dd/mm/yyyy"
63         // but am writing them process myself for algorithm marks
64         let date = timeStamp.split("T")[0].split("-");
65
66         // self invoking function that reverses the date and then stores the result in
67         // the dateReversed variable
68         let dateReversed = () => {
69             let temp = "";
70             let j = date.length -1;
71             for (let i = 0; i < date.length / 2; i++, j--) {
72                 temp = date[i];
73                 date[i] = date[j];
74                 date[j] = temp;
75             }
76         }
77
78         dateReversed();
79
80         return date.join("/");
81     }

```

```

65         date[j] = temp;
66     }
67     return date;
68 })();
69
70     return dateReversed.join("/");
71 }
72
73 // pass a date time stamp as argument format: YYYY-MM-DD'T'HH:MM:SS+MS:mS
74 // eg: '2019-02-02T22:52:49+00:00'
75 // will output the time as hh:mm:ss if the withSeconds argument is == true
76 // if it is not then the time will be outputted as hh:mm
77 static standardDateTo_Time_HHMMSS(dateTimeStamp, withSeconds) {
78
79     // splits argument into an array containing the date at index 0 & the time
80     // at index 1
81     // [1] at the end gets us just the time element. split the time element up
82     // into an array of characters,
83     // the array of characters will be:
84     // ["h","h",":","m","m",":","s","s","+","m","s",":","m","s"]
85     // in terms of format and length. if withSeconds option parameter is set to
86     // true, then all the characters
87     // in the array after the 7th index will get removed. if withSeconds is not
88     // true, elements after 4th index will get removed
89     let time = timeStamp.split("T")[1].split("");
90     let indexToRemoveFrom;
91     if (withSeconds !== true) {indexToRemoveFrom = 5}
92     else { indexToRemoveFrom = 8 }
93     //self invoking function that truncates the array of unwanted data will be
94     // returned
95     return () => {
96         let outputArr = []; // declare output array
97         // append elements from time to outputArray up to the indexToRemoveFrom
98         for (let i = 0; i < indexToRemoveFrom; i++) { outputArr.push(time[i]) }
99         return outputArr.join(""); //joins the array of characters together into
100        // a string
101    }();
102
103 // checks if the dateToCheck is equal to the current date
104 // dateToCheck should be type string in either dd/mm/yyyy format or standard
105 // date time format
106 static isToday (dateToCheck) {
107     // gets the users current date in a time stamp. Then, that time stamp
108     // gets turned into a dd/mm/yyyy format
109     let currentDate =
110         this.standardDateTo_Date_DDMMYYYY(this.getCurrentDateTime());
111     // checks to see if the date is in standard time or dd/mm/yyyy format
112     // if it is in standard time format then the length of the string will not
113     // be = to 10
114     // in that case the dateToCheck is converted into dd/mm/yyyy format
115     if (dateToCheck.length !== 10) { dateToCheck =
116         this.standardDateTo_Date_DDMMYYYY(dateToCheck) }
117     // compares the 2 strings, if the dates are equal true is returned.
118     // Otherwise, false is returned
119     return (dateToCheck === currentDate);
120 }
121
122 // checks if the dateToCheck's month or year is equal to the current month or year
123 // dateToCheck should be type string in either dd/mm/yyyy format or standard
124 // date time format
125 // monthOrYear option should be set to 'month' or 'year'
126 static isThisMonthOrYear (dateToCheck, monthOrYearOption) {
127     // gets the users current date in a time stamp. Then, that time stamp
128     // gets turned into a dd/mm/yyyy format
129     let currentDate =
130         this.standardDateTo_Date_DDMMYYYY(this.getCurrentDateTime());
131     // checks to see if the date is in standard time or dd/mm/yyyy format
132     // if it is in standard time format then the length of the string will not
133     // be = to 10

```

```

123     // in that case the dateToCheck is converted into dd/mm/yyyy format
124     dateToCheck = this[_ddmmyyyyToStandardDateTimeCheck] (dateToCheck);
125
126     // splits the datetime by "T" and then take index 0 of the array to get
127     // yyyy-mm-dd
128     // split the yyyy-mm-dd by "-" to get array of ["yyyy", "mm", "dd"]
129     dateToCheck = dateToCheck.split("T") [0].split("-");
130     currentDate = currentDate.split("T") [0].split("-");
131
132     // if the the option 'year' is not passed in as an argument the
133     // months will be compared by default
134     if (monthOrYearOption !== 'year') {
135         // returns the evaluation result of the equality between the month data
136         // from both arrays
137         return dateToCheck[1] === currentDate[1];
138     }
139     else {
140         // returns the evaluation result of the equality between the year data
141         // from both arrays
142         return dateToCheck[2] === currentDate[2];
143     }
144
145
146
147     // can pass in date as standard date time format or as dd/mm/yyyy as type string
148     // objectContainingTheAddingToMake should be type object can be a negative
149     // number to subtract days
150     // objectContainingTheAddingToMake: examples: {days : 6} or {months: 9} or
151     // {years : 9}
152     // can even do any combination: examples: {days : 7, months : 9} or {days : 8,
153     // months: 9, years : 2}
154     static addToDate (date, objectContainingTheAddingToMake) {
155         // if date is in dd/mm/yyyy format it gets changed into the standard date
156         // time format the moment.js library can work with
157         date = this[_ddmmyyyyToStandardDateTimeCheck] (date);
158         // uses the moment library to make relevant changes to the date. The result
159         // then gets formatted to the standard
160         // date format by using the .format()
161         return moment(date).add(objectContainingTheAddingToMake).format();
162     }
163
164
165
166     // can pass in date as standard date time format or as dd/mm/yyyy as type string
167     // objectContainingTheAddingToMake should be type object can be a negative
168     // number to subtract days
169     // objectContainingTheAddingToMake: examples: {days : 6} or {months: 9} or
170     // {years : 9}
171     // can even do any combination: examples: {days : 7, months : 9} or {days : 8,
172     // months: 9, years : 2}
173     static subtractFromDate (date, objectContainingTheSubtractionsToMake) {
174         // if date is in dd/mm/yyyy format it gets changed into the standard date
175         // time format the moment.js library can work with
176         date = this[_ddmmyyyyToStandardDateTimeCheck] (date);
177         // uses the moment library to make relevant changes to the date. The result
178         // then gets formatted to the standard
179         // date format by using the .format()
180         return moment(date).subtract(objectContainingTheSubtractionsToMake).format();
181     }

```

```
182
183
184
185 // method checks to see if the dateToBeChecked is in the past or in the future.
186 // dateToBeChecked should type string
187 // The argument should be in the standard date format for greater
188 // accuracy as time will also be accounted for then but dd/mm/yyyy format will
189 // work
190 // the method will return true if the date has passed and false if the date is
191 // in the future
192 static hasDatePassed (dateToBeChecked) {
193     let currentDateTime = this.getCurrentDateTime();
194     // if date is in dd/mm/yyyy format it gets changed into the standard date
195     // time format the moment.js library can work with
196     dateToBeChecked = this[_ddmmyyyyToStandardDateTimeCheck] (dateToBeChecked);
197
198     // returns the evaluation of the dates
199     return currentDateTime > dateToBeChecked;
200 }
201
202 // private method to check if a date is dd/mm/yyyy or standard date time stamp
203 // if the date is dd/mm/yyyy format then the date gets changed to the equivalent
204 // in
205 // the standard date time stamp
206 static [_ddmmyyyyToStandardDateTimeCheck] (date) {
207     if (date.length === 10) { date = DateHandler.createTimeStamp(date) }
208     return date;
209 }
210
module.exports = DateHandler;
```

Encryption.js

Kieran Marcus Williams

```

1  function Encryption() {}
2
3  const _generateKey = Symbol('generateKey');
4
5  // change all characters from input and key to character codes to store in separate
6  // arrays
7  // xor all input characters by key characters, store xor characters in new array
8
9  Encryption[_generateKey] = (key, inputLength) => {
10    let keyArray; // used to store key / password as an array of characters
11    let difference;
12    while (key.length <= inputLength) { // loop to make the key longer or equal to
13      the length of the input text
14      key = key + key + "p";
15    }
16    keyArray = key.split("");
17
18    // stores how much longer the length of the key is to the input
19    difference = keyArray.length - inputLength;
20
21    // loop to get rid of last element in the array each time to make length of
22    keyArray == inputArray
23    for (let i = 0; i < difference; i++) {
24      keyArray.pop();
25    }
26    return keyArray;
27  };
28
29
30  Encryption.xorEncrypt = (input, key) => {
31    let inputArray = input.split(""); // puts input into an array of characters
32    let keyArray = []; // used to store key / password as an array of characters
33    let xorValues = [];
34    keyArray = Encryption[_generateKey](key, input.length);
35
36    // replace all characters in their element with a character code for inputArray
37    // and keyArray
38    // preform xor on the elements, storing the answers in the xorValues Array
39    for (let i = 0; i < keyArray.length; i++) {
40      keyArray[i] = keyArray[i].charCodeAt(0).toString();
41      inputArray[i] = inputArray[i].charCodeAt(0).toString();
42      xorValues[i] = keyArray[i] ^ inputArray[i]; // preforms the xor on both
43      // characters from the key and the input
44    }
45
46
47    //makes the last number of the coma separated string, the length of original
48    // message
49    // and puts double quotes around the string so that it can be saved to the
50    // csv file without causing disruption to the rest of the file
51    return xorValues.join("/") + "/" + (input.length).toString();
52  };
53
54
55  // Static method that can be called anywhere in my code
56  Encryption.xorDecrypt = (input, key) => { // input needs to be comma separated
57  // integers as one string.
58    let inputArray = input.split("/"); // splits the input array up into an array of
59    // the numbers
60    let keyArray = [];
61    let xorValues = [];
62    let decryptedTextArray = [];
63
64    // gets length of input before it was encrypted and stores it in a constant
65    const INPUT_LENGTH_BEFORE_ENCRYPTION = inputArray[(inputArray.length-1)];
66    inputArray.pop(); // gets rid of the length of original text element
67    keyArray = Encryption[_generateKey](key, INPUT_LENGTH_BEFORE_ENCRYPTION);
68
69    for (let i = 0; i < keyArray.length; i++) {
70      keyArray[i] = keyArray[i].charCodeAt(0).toString();
71      xorValues[i] = keyArray[i] ^ inputArray[i]; // preforms the xor on both
72      // characters from the key and the input
73
74    // gets the character code from the xor result, turns it into a string and

```

```
65     then puts it into the decryptedTextArray
66     decryptedTextArray[i] = String.fromCharCode(xorValues[i]);
67   }
68
69   return (decryptedTextArray.join(""))); // decrypted text gets returned
70
71 }
72
73 module.exports = Encryption;
74
75
76
77 // // Test code
78 //
79 // let text = "hey";
80 // let password = "wowww";
81 // let fakePass = "wowx";
82 //
83 //
84 // let encryptedText = Encryption.xorEncrypt(text, password);
85 // let decryptedText = Encryption.xorDecrypt(encryptedText, fakePass);
86 //
87 // console.log(`text: ${text} \npassword: ${password} \nencrypted text:
$encryptedText} \ndecrypted text: ${decryptedText}`);
```

GenerateDataPoints.js

```

1  const CSVHandler = require('./CSVHandler');
2  const DateHandler = require('./DateHandler');
3
4  const _getDataPointsFromLast7Days = Symbol("getDataPointsFromLast7Days");
5  const _fileToLookThrough = Symbol("fileToLookThrough");
6  const _clientsGainedCSVHandlerObject = Symbol("clientsGainedCSVHandlerObject");
7  const _clientsLostCSVHandlerObject = Symbol("clientsLostCSVHandlerObject");
8  const _leadsLostCSVHandlerObject = Symbol("leadsLostCSVHandlerObject");
9
10 class GenerateDataPoints{
11
12     constructor (fileToLookThrough, clientsGainedCSVHandlerObject,
13     clientsLostCSVHandlerObject, leadsLostCSVHandlerObject) {
14         this[_fileToLookThrough] = fileToLookThrough;
15         this[_clientsGainedCSVHandlerObject] = clientsGainedCSVHandlerObject;
16         this[_clientsLostCSVHandlerObject] = clientsLostCSVHandlerObject;
17         this[_leadsLostCSVHandlerObject] = leadsLostCSVHandlerObject;
18     }
19
20     // will get the data points and the labels for the past 7 days over however many
21     // weeks and return the labels as dd/mm/yyyy
22     // and will also output the data points which will be the number of matches
23     // found on the date being checked
24     [_getDataPointsFromLast7Days] (numOfWeeks) {
25
26         let clientsGained = this[_clientsGainedCSVHandlerObject];
27         let clientsLost = this[_clientsLostCSVHandlerObject];
28         let leadsLost = this[_leadsLostCSVHandlerObject];
29
30         let dataPointsArray = [];
31         let datesArray = []; // will be used to store the dates so that they can be
32         // used as labels for the graph
33
34         // uses the getCurrentDate() method instead of getCurrentDateTime() method,
35         // as the dates stored in the database wont contain time information. This
36         // will allow for an easy check on the dates
37         const DATE = DateHandler.getCurrentDate(); // date is a constant value as it
38         // will not get changed at all
39         let date = DATE; // date is used as a temp variable that will get
40         // manipulated in the loop
41         let noDays = numOfWeeks * 7;
42         let numberOfDateMatches = 0;
43         // variables below used to get the date into dd/mm/yyyy for the labels
44         let dateDDMMYY_to_standardDateTime;
45         let standardDateToDDMMYY;
46
47         // console.log(`current date: ${DATE}`); // will output the current date for
48         // debugging
49
50         for (let i = noDays - 1; i >= 0; i--) {
51             date = DateHandler.subtractFromDate(DATE, {days: i});
52             // console.log(`${i} days ago: ${date}`); // can be used to check the
53             // dates I am getting back
54
55             switch (this[_fileToLookThrough]){
56                 case "clientsGained":
57                     numberOfDateMatches = clientsGained.countMatches([1],date);
58                     break;
59                 case "clientsLost":
60                     numberOfDateMatches = clientsLost.countMatches([1],date);
61                     break;
62                 case "leadsLost":
63                     numberOfDateMatches = leadsLost.countMatches([1],date);
64                     break;
65                 default:
66                     console.error("ensure the fileToLook through constructor method
67                     argument is: 'clientsGained', 'clientsLost', or 'leadsLost' \n" +
68                         "You entered: " + this[_fileToLookThrough]);
69                     break;
70             }
71         }
72     }

```

```

64 // done to get the date into dd/mm/yyyy for the labels
65 dateDDMMYY_to_standardDateTime =
66 DateHandler.standardDateTo_Date_DDMMYYYY(date);
67 standardDateToDDMMYY =
68 DateHandler.standardDateTo_Date_DDMMYYYY(DateHandler.createTimeStamp(dateD
69 DMMYY_to_standardDateTime));
70
71 ///////////////////////////////////////////////////
72 ///////////////////////////////////////////////////
73
74 dataPointsArray.push(numberOfDateMatches); // puts the number of matches
75 per date into an array
76 }
77 // method returns an object containing the labels and the corresponding
78 // dataPoints
79
80
81 // returns the last 14 days of data points for the files: clientsGained,
82 // clientsLost leadsLost
83 // them above strings are the accepted parameter values in the string type
84 getPast14DaysOfDataPoints() {
85
86     return this[_getDataPointsFromLast7Days] (2, this[_fileToLookThrough]);
87 }
88
89
90
91 // returns data points for 12 weeks of matches from the files: clientsGained,
92 // clientsLost leadsLost
93 // them above strings are the accepted parameter values in the string type
94 getDataPointsForLast12Weeks () {
95     // get the past 12 weeks of data as individual days and then go through the
96     // array
97     // add 7 of the match numbers at a time append that value to a new array for
98     // data points
99     // and the first date of the 7 dates to a different array for labels
100
101    let allLabels = this[_getDataPointsFromLast7Days](12).labels;
102    let allDataPoints = this[_getDataPointsFromLast7Days](12).dataPoints;
103    let usedLabels = [];
104    let usedDataPoints = [];
105    let totalMatchesInTheWeek;
106
107    // goes through all 12 weeks
108    for (let i = 0; i < 12; i++) {
109
110        totalMatchesInTheWeek = 0; // initializes the variable / resets the
111        // variable back to 0
112
113        usedLabels.push(allLabels[i*7]); // appends the 1st date of each week
114        // into an array for the labels
115
116        // goes through all 7 days in the week
117        for (let j = 0; j < 7; j++) {
118            totalMatchesInTheWeek += allDataPoints[j]; // gets total matches for
119            // the week
120        }
121
122        usedDataPoints.push(totalMatchesInTheWeek); // appends the average
123        // number of matches within the week to an array
124        allDataPoints.splice(0,7); // deletes the first week (7 days)/ 7
125        // elements of the array as they are no longer needed
126    }
127
128
129
130
131
132
133
134
135
136
137
138
139

```

```

120     return {labels: usedLabels, dataPoints: usedDataPoints}
121 }
122
123
124
125     getDataPointsForLast12Months () {
126
127         // get the dates of the past 12 months
128         // turn them dates into yyyy-mm and store in an array
129         // count matches in the 2nd index of the files for each of the 12 dates
130
131         let clientsGained = this[_clientsGainedCSVHandlerObject];
132         let clientsLost = this[_clientsLostCSVHandlerObject];
133         let leadsLost = this[_leadsLostCSVHandlerObject];
134
135         const CURRENT_DATE = DateHandler.getCurrentDateTime(); // gets the current
136         time
137         let date = CURRENT_DATE; // will be the variable getting changed
138         let numberOfDateMatches;
139         let dataPointsArray = [];
140         let labelsArray = [];
141
142         for (let i = 11; i >= 0; i--) {
143
144             date = DateHandler.subtractFromDate(date, {month: i}); // subtracts i
145             months from the current date
146
147             labelsArray.push(DateHandler.dateToWordedMonth(date)); // appends the
148             month as it's name to the csv file
149
150             date = date.split(""); // puts the date into an array of characters
151             date.splice(7,18); // gets rid of the last 18 characters
152             date = date.join("");
153
154             switch (this[_fileToLookThrough]) {
155                 case "clientsGained":
156                     numberOfDateMatches = clientsGained.countMatches([2],date);
157                     break;
158                 case "clientsLost":
159                     numberOfDateMatches = clientsLost.countMatches([2],date);
160                     break;
161                 case "leadsLost":
162                     numberOfDateMatches = leadsLost.countMatches([2],date);
163                     break;
164                 default:
165                     console.error("ensure the fileToLook through constructor method
166                     argument is: 'clientsGained', 'clientsLost', or 'leadsLost' \n" +
167                     "You entered: " + this[_fileToLookThrough]); // outputs an
168                     error message of what is most likely the cause of any
169                     problems in this case
170                     break;
171             }
172
173             dataPointsArray.push(numberOfDateMatches); // appends the number of
174             matches found for that month to the data points array
175
176         date = CURRENT_DATE; // sets the date back to the current date so that
177         it can be
178     }
179
180     return {labels: labelsArray, dataPoints: dataPointsArray}
181 }
182
183 module.exports = GenerateDataPoints;

```

Search.js

```

1  const Sort = require("./Sort"); // allows the use of sort methods in this file
2
3  function Search(){}
4
5
6  // creates private method
7  const _binarySearch2DFinder = Symbol('binarySearch2DFinder');
8  const _checkBoundsForMatches = Symbol('checkBoundsForMatches');
9
10
11
12 // searches for a value and returns the index the element was found at
13 // or returns -1 if the search value was not found in the array
14 // ONLY WORKS ON SORTED ARRAYS
15 Search.binarySearch = (array, searchValue) => {
16     // makes sure that the array is sorted before the search is done
17     array = Sort.bubbleSort(array);
18     let first = 0;
19     let last = array.length;
20     let middle = Math.floor((first + last)/2);
21     let found = false;
22     while(!found || (first <= last)){
23         if(array[middle] === searchValue){
24             found = true;
25             return middle;
26         }
27         if(searchValue < array[middle]){
28             last = middle - 1;
29             middle = Math.floor((first + last)/2);
30         }
31         if(searchValue > array[middle]){
32             first = middle + 1;
33             middle = Math.floor((first + last)/2);
34         }
35     }
36     return -1;
37 };
38
39 Search[_checkBoundsForMatches] = (array, middle, col, searchValue) => {
40     let matchAtBoundsFoundFlag = true; // set to true so that the loop will start
41     let recordsFound = [];
42     let upperBounds = middle;
43     let lowerBounds = middle;
44
45     // the algorithm works in the binary search as any other matches will be found
46     // either side of the initial match
47     // due to a binary search only working on sorted arrays which will result in the
48     // matches in conjunction in the array
49
50     // loop through upper values
51     while (matchAtBoundsFoundFlag === true){
52
53         // set to false so that if no matches are found on the upper bounds from the
54         // first match
55         // the loop will end.
56         matchAtBoundsFoundFlag = false;
57
58         // checks that if the upperBounds was to get increased by one,
59         // the program will not crash due to the record not being in the bounds of
60         // the array
61         if ((upperBounds + 1) > (array.length - 1)) {
62             break; // terminate the loop
63         }
64
65         if (array[upperBounds + 1][col] === searchValue) { // checks to see if the
66             array element above is a match to the search value
67             matchAtBoundsFoundFlag = true; // set back to true so that the loop will
68             run again to check for more matches.
69             upperBounds++; // increase value stored in upperBounds variable by 1
70
71             // match to the search value was found so the recordsFound
72             // array gets the record and index of record in a javascript object
73             // appended to the end of it

```

```

67         recordsFound.push({ record : array[upperBounds], index : upperBounds });
68     }
69 }
70
71 matchAtBoundsFoundFlag = true; // reset the flag value so the next loop can run
72
73 // loop through lower values
74 while (matchAtBoundsFoundFlag === true) {
75
76     // set to false so that if no matches are found on the lower bounds of the
77     // first match
78     // the loop will end.
79     matchAtBoundsFoundFlag = false;
80
81     // checks that if the lowerBounds was to get decreased by one,
82     // the program will not crash due to the record not being in the bounds of
83     // the array
84     if ((lowerBounds - 1) < (0)) {
85         break; // terminate the loop
86     }
87
88     if (array[lowerBounds - 1][col] === searchValue) {
89         matchAtBoundsFoundFlag = true; // set back to true so that the loop will
90         // run again to check for more matches.
91         lowerBounds--; // increase value stored in lowerBounds variable by 1
92
93     }
94 }
95
96 return recordsFound;
97 };
98
99 // purpose of method is to find a match with the searchValue in a set column
100 Search[_binarySearch2DFinder] = (array, col, searchValue) => {
101     let first = 0;
102     let last = array.length;
103     let middle = Math.floor((first + last)/2);
104     let upperAndLowerBoundMatches = [];
105     let recordsFound = [];
106
107
108
109     // loop that will keep going until last > first or the matching records are found
110     // the array[middle][col] ---> middle is the row that is currently under
111     // inspection, col is a constant
112     while (first <= last) {
113
114         // to avoid a crash, the program will check to see if middle is greater than
115         // the index of the last
116         // record and that if it is a negative value. In either instance the loop
117         // will be terminated
118         if (middle > array.length - 1 || middle < 0){
119             break;
120         }
121
122         if(array[middle][col] === searchValue){
123             //pushes the array and an index contained in a javascript object type
124             recordsFound.push({ record : array[middle], index : middle }); // adds
125             the record into records found array
126
127             // finds any matches in the upper and lower bounds of the
128             upperAndLowerBoundMatches = Search[_checkBoundsForMatches](array,
129             middle, col, searchValue);
130
131             upperAndLowerBoundMatches.forEach((match) => {
132                 recordsFound.push(match);
133             });
134             break; // gets out of loop
135         }
136     }
137 }
```

```

131
132     // terminates the loop if the middle value is 0 as it will cause a crash
133     // crash is due to the fact that middle will turn into a negative value
134     if (middle === 0) {
135         break;
136     }
137
138     if(searchValue < array[middle][col]){
139         last = middle - 1;
140         middle = Math.floor((first + last)/2);
141     }
142     if(searchValue > array[middle][col]){
143         first = middle + 1;
144         middle = Math.floor((first + last)/2);
145     }
146 }
147
148 if (recordsFound.length > 0){
149     return recordsFound;
150 }
151 };
152
153
154
155
156 // returns a 2D array of matching records
157 // columnsToBeSearched should be an array of integers
158 // that determines what columns are to be searched
159 Search.binarySearch2D = (array, columnsToBeSearched, searchValue) => {
160
161     let recordsFoundInAllColumnsSearched = [];
162     let recordsFoundInOneColumn = [];
163     let col = 0;
164
165     // if there is only one record in the array then there is no point in running
166     // any more code
167
168     for (let i = 0; i < columnsToBeSearched.length; i++) { // goes through all
169     // columns that need to be searched
170
171         col = columnsToBeSearched[i]; // col is made equal to the i index of the
172         // columnsToBeSearched array
173
174         // sorts the array with regard to the column that is being searched on
175         array = Sort.bubbleSort2D(array, col);
176
177         recordsFoundInOneColumn = Search[_binarySearch2DFinder](array, col,
178         searchValue);
179
180         // if statement to check that what got returned is an
181         // array of records found and not just undefined
182         if (recordsFoundInOneColumn !== undefined){
183             // goes through every record in the recordsFoundInOneColumnArray
184             recordsFoundInOneColumn.forEach((currentRecord) => {
185                 recordsFoundInAllColumnsSearched.push(currentRecord);
186             });
187         }
188     }
189 }
190
191 return recordsFoundInAllColumnsSearched;
192
193
194 //searches for a value and returns the index the element was found at
195 //or returns -1 if the search value was not found in the array
196
197 Search.linearSearch = (array, searchValue) => {
198     let found = false;
199     let i = 0;

```

```
200 let length = array.length;
201 while((!found) || (i < length)){
202     if(array[i] === searchValue){
203         found = true;
204         return i;
205     } else {
206         i++;
207     }
208 }
209 return -1;
210 };
211
212 Search.getRecords = (ArrayOfObjectsContainingRecordsWithIndexes) => {
213     let arrayOfRecords = [];
214     ArrayOfObjectsContainingRecordsWithIndexes.forEach((recordWithIndex) => {
215         arrayOfRecords.push(recordWithIndex.record);
216     });
217     return arrayOfRecords;
218 };
219
220 module.exports = Search;
221
222
223
```

Sort.js

```

1
2
3 function Sort (){}
4
5 const _comparisonOperator = Symbol('comparisonOperator');
6
7
8 // private function used so that dynamic comparisons can be made
9 // params should take a < or > type string in the comparissonOp
10 // value1 and value2 should be the two values that are going to be compared.
11 Sort[_comparisonOperator] = (comparisonOp, value1, value2) => {
12   if(comparisonOp === "<") { return value1 < value2 }
13   if(comparisonOp === ">") { return value1 > value2 }
14 };
15
16 // these methods can be considered an imitation of static methods
17
18 // sorts the array in ascending or descending order
19 // depending on if the ascending variable is set to true
20 // or false, and returns the sorted array
21 // THE ASCENDING PARAMETER COULD BE CONSIDERED POLYMORPHISM AS THE METHODS
22 // FUNCTIONALITY DIFFERS DEPENDENT ON
23 // WHETHER THE ARGUMENT PASSED SETS IT TO FALSE OR IS LEFT ALONE/ SET TO TRUE
24 Sort.bubbleSort = (array, ascending) => {
25
26   if (ascending === undefined) ascending = true;
27   // if no ascending argument is passed, method sorts array in ascending order
28
29   let n = array.length;
30   let sorted = false;
31   let temp = 0;
32   let comparisonOp = "";
33
34   if (ascending) comparisonOp = ">";
35   if (!ascending) comparisonOp = "<";
36
37   while(!sorted){
38
39     sorted = true;
40     for(let i = 0; i < n; i++){
41
42       // if(array[i] > array[i+1]) is for ascending
43
44       // uses the comparisonOperator method to make this
45       // method dynamic in sorting both ascending and descending
46       if (Sort[_comparisonOperator](comparisonOp,
47         array[i], array[i+1])) {
48
49         temp = array[i];
50         array[i] = array[i+1];
51         array[i+1] = temp;
52         sorted = false;
53       }
54     }
55
56   return array;
57 };
58
59 // works by comparing a set lot of columns from different
60 // records against each other. Swaps the records if the condition
61 // for a swap to be made is met (condition is determined by the boolean
62 // argument "ascending")
63 // column to sort by should be type number
64 // ascending should be boolean or not entered
65 Sort.bubbleSort2D = (array2D, columnToSortBy, ascending) => {
66
67   if (ascending === undefined) ascending = true;
68   // if no ascending argument is passed, method sorts array in ascending order
69
70   let n = array2D.length;
71   let sorted = false;
72   let temp = [];

```

```
73 let comparisonOp = "";
74
75 if (ascending) comparisonOp = ">";
76 if (!ascending) comparisonOp = "<";
77
78
79 while(!sorted){
80     sorted = true;
81     for(let i = 0; i < n -1 ; i++){
82
83         // if(array[i] > array[i+1]) is for ascending
84
85         // uses the comparisonOperator method to make this
86         // method dynamic in sorting both ascending and descending
87         if (Sort[_comparisonOperator](comparisonOp,
88             array2D[i][columnToSortBy], array2D[i+1][columnToSortBy])) {
89
90
91             temp = array2D[i];
92             array2D[i] = array2D[i+1];
93             array2D[i+1] = temp;
94             sorted = false;
95
96
97         }
98     }
99 }
100
101 return array2D;
102 };
103
104
105
106 module.exports = Sort;
107
108
109
```

Validation.js

```

1  function Validation () {}
2
3  const _stringLookupCheckForLowerCase = Symbol('stringLookupCheckForLowerCase');
4  const _stringLookupCheckForUpperCase = Symbol('stringLookupCheckForUpperCase');
5  const _stringLookupCheckForNumbers = Symbol('stringLookupCheckForNumbers');
6  const _stringLookupCheckForSpaces = Symbol('stringLookupCheckForSpaces');
7
8  Validation.presenceCheck = (value) => {
9    return !(value === undefined || value === null || /^[^\s]*$/.test(value));
10 }
11
12 Validation.lengthCheck = (value, length) => {
13   return value.length === length;
14 }
15
16 Validation.rangeCheck = (value, lower, upper, stringLengthInRange) => {
17   if (typeof value === "string") {
18     if (stringLengthInRange) {
19       value = value.length;
20     }
21     else {
22       value = Number(value);
23     }
24   }
25
26   if (upper === undefined || upper === null) {
27     upper = lower;
28   }
29
30   return (value >= lower && value <= upper );
31 }
32
33 // checks if the value type is number
34 // the stringToNumberConversionNeeded parameter should take a boolean value
35 // if it is set to true then the string will get converted into a number before
36 // being checked
37 Validation.isaNumber = (value, stringToNumberConversionNeeded) => {
38
39   if (stringToNumberConversionNeeded === true) {
40     value = Number(value);
41   }
42
43   return typeof(value) === "number" && !isNaN(value);
44 };
45
46
47 Validation.isaInt = (value) => {
48   return typeof(value) === "number" && !isNaN(value) && value % 1 === 0;
49 };
50
51 Validation.isaReal = (value) => {
52   return typeof(value) === "number" && !isNaN(value) && value % 1 !== 0;
53 };
54
55 Validation.isaString = (value) => {
56   return typeof(value) === "string";
57 };
58
59 Validation.isaBoolean = (value) => {
60   return typeof(value) === "boolean";
61 };
62
63 Validation.isaFunction = (value) => {
64   return typeof(value) === "function";
65 };
66
67 Validation.isaObject = (value) => {
68   return typeof(value) === "object";
69 };
70
71 Validation.isArray = (value) => {
72   return Array.isArray(value);

```

```

73 } ;
74
75 Validation.isaCharacter = (value) => {
76   return typeof(value) === "string" && Validation.lengthCheck(value, 1);
77 };
78
79 Validation.fileIsCSV = (file) => {
80   let regex = /^[a-zA-Z0-9\s\-\_\.]+(\.csv)$/;
81   return (regex.test(file));
82 };
83
84 Validation.fileIsTXT = (file) => {
85   let regex = /^[a-zA-Z0-9\s\-\_\.]+(\.txt)$/;
86   return (regex.test(file.value));
87 };
88
89
90 // value should be the date entry, futureDate parameter is for a boolean argument
91 // if futureDate has a true value passed then it will check that the value is
92 // a date into the future
93 Validation.dateFormat = (value, futureDate) => {
94
95   let valueArray = [];
96   let val = 0;
97   let day = "";
98   let month = "";
99   let year = "";
100  let usersCurrentDate = new Date();
101  let usersDay = "";
102  let usersMonth = "";
103  let usersYear = "";
104  // an object that stores the maximum number of days
105  // any month could have
106  let monthAndMaxDay = {
107    jan: 31,
108    feb: 28 || 29,
109    mar: 31,
110    apr: 30,
111    may: 31,
112    jun: 30,
113    jul: 31,
114    aug: 31,
115    sep: 30,
116    oct: 31,
117    nov: 30,
118    dec: 31
119  };
120
121  // function inside the static method to check if the day value is within the
122  // bounds for that month
123  let monthDayChecker = () => {
124    let maxDay = 0;
125
126    // uses a switch statement to check all possible cases for the months then
127    // gets the max
128    // day possible for that month from the monthAndMaxDay object
129    // a range check is then done using the maxDay as the inclusive upper bound
130    // with 1 being the inclusive lower bound
131    switch (month){
132      case "01":
133        maxDay = monthAndMaxDay.jan;
134        if (!Validation.rangeCheck(day, 1, maxDay)){
135          return false;
136        }
137        break;
138      case "02":
139        maxDay = monthAndMaxDay.feb;
140        if (!Validation.rangeCheck(day, 1, maxDay)){
141          return false;
142        }
143        break;
144      case "03":
145        maxDay = monthAndMaxDay.mar;

```

```

145         if (!Validation.rangeCheck(day, 1, maxDay)) {
146             return false;
147         }
148         break;
149     case "04":
150         maxDay = monthAndMaxDay.apr;
151         if (!Validation.rangeCheck(day, 1, maxDay)) {
152             return false;
153         }
154         break;
155     case "05":
156         maxDay = monthAndMaxDay.may;
157         if (!Validation.rangeCheck(day, 1, maxDay)) {
158             return false;
159         }
160         break;
161     case "06":
162         maxDay = monthAndMaxDay.jun;
163         if (!Validation.rangeCheck(day, 1, maxDay)) {
164             return false;
165         }
166         break;
167     case "07":
168         maxDay = monthAndMaxDay.jul;
169         if (!Validation.rangeCheck(day, 1, maxDay)) {
170             return false;
171         }
172         break;
173     case "08":
174         maxDay = monthAndMaxDay.aug;
175         if (!Validation.rangeCheck(day, 1, maxDay)) {
176             return false;
177         }
178         break;
179     case "09":
180         maxDay = monthAndMaxDay.sep;
181         if (!Validation.rangeCheck(day, 1, maxDay)) {
182             return false;
183         }
184         break;
185     case "10":
186         maxDay = monthAndMaxDay.oct;
187         if (!Validation.rangeCheck(day, 1, maxDay)) {
188             return false;
189         }
190         break;
191     case "11":
192         maxDay = monthAndMaxDay.nov;
193         if (!Validation.rangeCheck(day, 1, maxDay)) {
194             return false;
195         }
196         break;
197     case "12":
198         maxDay = monthAndMaxDay.dec;
199         if (!Validation.rangeCheck(day, 1, maxDay)) {
200             return false;
201         }
202         break;
203     default:
204         return false;
205     }
206 };
207
208 // checks that the input is a string and that it has a length of 10 characters
209 if(Validation.isaString(value) && Validation.lengthCheck(value, 10)){
210
211     valueArray = value.split(""); // puts the string into an array
212
213     // loops through every element in the array so that checks can be done on
214     // date, month and the year part of the date
215     for(let i = 0; i < valueArray.length; i++){
216
217         // will be using this a lot so to make it more readable, the current

```

```

array element is set to the variable val
217    val = valueArray[i];
218
219    // the 2 an 5 indexes in the array should be a '/' if they aren't false
220    // is returned
221    if(i === 2 || i === 5){
222        if(val !== "/"){
223            return false;
224        }
225    } else {
226
227        // checks that any other character is a number value, if it isn't
228        // false is returned
229        if (!Validation.isaNumber(Number(val))){ 
230            return false;
231        }
232    }
233
234    // sets the day, month and year variables to the relevant characters
235    day = valueArray[0] + valueArray[1];
236    month = valueArray[3] + valueArray[4];
237    year = valueArray[6] + valueArray[7] + valueArray[8] + valueArray[9];
238
239    // check that the day is between 1 and 31 inclusive, return false if it is not
240    if (!Validation.rangeCheck(day, 1, 31)){
241        return false;
242    }
243
244    // check that the month is between 1 and 12 inclusive, if it is not then
245    // return false
246    if (!Validation.rangeCheck(month, 1, 12)){
247        return false;
248    }
249
250    // check that the day is in the range for that month
251    monthDayChecker();
252
253    if (futureDate !== true) {
254        return true;
255    } else { // if the future date is needed
256
257        // set variables fot the users current system date
258        usersDay = usersCurrentDate.getDate();
259        usersMonth = usersCurrentDate.getMonth() + 1;
260        usersYear = usersCurrentDate.getFullYear();
261
262        // check that the date entered is one for the future
263        if (Number(year) > usersYear){ return true; }
264        if (Number(year) === usersYear){
265            if (Number(month) > usersMonth){ return true; }
266            if (Number(month) === usersMonth) {
267                return (Number(day) >= usersDay); // returns true if the entered
268                // date is in the future or is the current day
269            }
270        } else {
271            return false;
272        }
273    } else {
274        return false;
275    }
276
277 }
278
279 } else {
280     return false;
281 }
282
283 };
284

```

```

285 Validation.timeFormat_HHMM = (value) => {
286
287     let valueArray = [];
288     let val = 0;
289     let hour, minute;
290
291     // checks that the input is a string and that it has a length of 5 characters
292     if(Validation.isaString(value) && Validation.lengthCheck(value, 5)) {
293
294         valueArray = value.split(""); // puts the string into an array
295
296         // loops through every element in the array so that checks can be done on
297         // the hour and minute parts of the time
298         for (let i = 0; i < valueArray.length; i++) {
299
300             // will be using this a lot so to make it more readable, the current
301             // array element is set to the variable val
302             val = valueArray[i];
303
304             // the 2 index in the array as the element should be a ':' if it is not,
305             // false is returned
306             if (i === 2) {
307                 if (val !== ":") {
308                     return false;
309                 }
310             } else {
311
312                 // checks that any other character is a number value, if it isn't
313                 // false is returned
314                 if (!Validation.isaNumber(Number(val))) {
315                     return false;
316                 }
317             }
318
319             hour = valueArray[0] + valueArray[1];
320             minute = valueArray[3] + valueArray[4];
321
322             return ((Number(hour) >= 0 && Number(hour) <= 23 && Number(minute) >= 0 &&
323             Number(minute) <= 59));
324         }
325     } else {
326         return false;
327     }
328 };
329
330 Validation.emailFormat = (value) => {
331     let emailRegex =
332         /^(([^\<\>()\\[\]\\.,,:\\s@"]+(\\\.[^\<\>()\\[\]\\.,,:\\s@"]+)*|(.+"))@((\\[[0-9]{1,3}\\.[0-
333         9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}])|(([a-zA-Z\\-0-9]+\\.)+[a-zA-Z]{2,}))$/ ;
334
335     return emailRegex.test(value);
336 };
337
338
339 Validation.timeFormat_HHMMSS = (value) => {
340
341     let valueArray = [];
342     let val = 0;
343     let hour, minute, second;
344
345     // checks that the input is a string and that it has a length of 5 characters
346     if(Validation.isaString(value) && Validation.lengthCheck(value, 8)) {
347
348         valueArray = value.split(""); // puts the string into an array
349
350         // loops through every element in the array so that checks can be done on
351         // the hour and minute parts of the time
352         for (let i = 0; i < valueArray.length; i++) {
353
354             // will be using this a lot so to make it more readable, the current
355             // array element is set to the variable val
356             val = valueArray[i];

```

```

349
350     // the 2 index in the array as the element should be a ':' if it is not,
351     // false is returned
352     if (i === 2 || i === 5) {
353         if (val !== ":") {
354             return false;
355         }
356     } else {
357
358         // checks that any other character is a number value, if it isn't
359         // false is returned
360         if (!Validation.isaNumber(Number(val))) {
361             return false;
362         }
363     }
364
365     hour = valueArray[0] + valueArray[1];
366     minute = valueArray[3] + valueArray[4];
367     second = valueArray[6] + valueArray[7];
368
369     if (hour === "00" && minute === "00") {
370
371     }
372
373     return ((Number(hour) >= 0 && Number(hour) <= 23 && Number(minute) >= 0 &&
374     Number(minute) <= 59
375         && Number(second) >= 0 && Number(second) <= 59));
376 } else {
377     return false;
378 }
379 };
380
381
382 // EXAMPLE OF A POLYMORPHIC METHOD
383 // value two can be an array to check valueOne against multiple values in
384 // the valueTwo argument. although valueTwo can be a single value to check equality
385 Validation.comparisonCheck = (valueOne, valueTwo) => {
386     if (typeof valueTwo === "object") {
387         let foundMatchFlag = false;
388         // iterate through the array and compare valueOne to all the values in the
389         // array, if one matches then return true
390         // if there's no matches return false
391         for (let i = 0; i < valueTwo.length; i++) {
392             if (valueOne === valueTwo[i]) {
393                 foundMatchFlag = true;
394             }
395         }
396         return foundMatchFlag;
397     } else { // if valueTwo is not an array then check for equality on the single
398     // value
399         return valueOne === valueTwo;
400     }
401 };
402
403
404 // preforms a look up check on the string comparing every character in the argument
405 // string
406 // to an array of acceptable characters. true is returned if there is a match found
407 // in the string to the list of
408 // acceptable characters, false returned if there is no match.
409 Validation[_stringLookupCheckForLowerCase] = (stringValue) => {
410     let lowerCaseFlag = false;
411     // all possible lower case letters put into an array of characters
412     let lowerCaseArray = "abcdefghijklmnopqrstuvwxyz".split("");
413     // The stringValue argument gets split up into an array of characters
414     let stringArray = stringValue.split("");
415     let currentCharacter = '';

```

```

415     // check for a lower case character in the stringValue
416     // by iterating through the stringArray and checking to see if a value
417     // matches any value in the lowerCaseArray
418     // this is achieved through using a NESTED FOR LOOP
419     for (let i = 0; i < stringArray.length; i++) {
420         currentCharacter = stringArray[i];
421         for (let j = 0; j < lowerCaseArray.length; j++) {
422             // if a value in the stringArray matches the value in the
423             // lowerCaseArray, set the lowerCaseFlag to true
424             if ( currentCharacter === lowerCaseArray[j] ) {
425                 lowerCaseFlag = true;
426             }
427         }
428     }
429 }
430
431
432 // preforms a look up check on the string comparing every character in the argument
433 // string
434 // to an array of acceptable characters. true is returned if there is a match found
435 // in the string to the list of
436 // acceptable characters, false returned if there is no match.
437 Validation[_stringLookupCheckForUpperCase] = (stringValue) => {
438     let upperCaseFlag = false;
439     // all possible upper case letters put into an array of characters
440     let upperCaseArray = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".split("");
441     // The stringValue argument gets split up into an array of characters
442     let stringArray = stringValue.split("");
443     let currentCharacter = '';
444
445     // check for a upper case character in the stringValue
446     // by iterating through the stringArray and checking to see if a value matches
447     // any value in the upperCaseArray
448     // this is achieved through using a NESTED FOR LOOP
449     for (let i = 0; i < stringArray.length; i++) {
450         currentCharacter = stringArray[i];
451         for (let j = 0; j < upperCaseArray.length; j++) {
452             // if a value in the stringArray matches the value in the
453             // upperCaseArray, set the upperCaseFlag to true
454             if ( currentCharacter === upperCaseArray[j] ) {
455                 upperCaseFlag = true;
456             }
457         }
458     }
459
460     // if the upperCaseFlag is still set to false after the loop, then an upperCase
461     // character doesn't exist, false is returned
462     return upperCaseFlag;
463 };
464
465
466 // preforms a look up check on the string comparing every character in the argument
467 // string
468 // to an array of acceptable characters. true is returned if there is a match found
469 // in the string to the list of
470 // acceptable characters, false returned if there is no match.
471 Validation[_stringLookupCheckForNumbers] = (stringValue) => {
472     let numbersFlag = false;
473     // all possible number characters put into an array of characters
474     let numbersArray = "1234567890".split("");
475     // The stringValue argument gets split up into an array of characters
476     let stringArray = stringValue.split("");
477     let currentCharacter = '';
478
479     // check for a number character in the stringValue
480     // by iterating through the stringArray and checking to see if a value matches
481     // any value in the numbersArray
482     // this is achieved through using a NESTED FOR LOOP
483     for (let i = 0; i < stringArray.length; i++) {
484         currentCharacter = stringArray[i];
485         for (let j = 0; j < numbersArray.length; j++) {

```

```

477     // if a value in the stringArray matches the value in the numbersArray,
478     // set the numbersFlag to true
479     if ( currentCharacter === numbersArray[j] ) {
480         numbersFlag = true;
481     }
482 }
483 // if the numbersFlag is still set to false after the loop, then a number
484 character doesn't exist, false is returned
485 return numbersFlag;
486 };
487
488 // preforms a look up check on the string comparing every character in the argument
489 // string
490 // to a space character. true is returned if there is a match found in the string,
491 // false returned if there is no match.
492 Validation[_stringLookupCheckForSpaces] = (stringValue) => {
493     let spaceFlag = false;
494     // the comparison space character
495     let spaceCharacter = " ";
496     // The stringValue argument gets split up into an array of characters
497     let stringArray = stringValue.split("");
498     let currentCharacter = '';
499
500     // check for a space character in the stringValue
501     // by iterating through the stringArray and checking to see if a value matches
502     // any value in the spaceCharacter
503     for (let i = 0; i < stringArray.length; i++) {
504         currentCharacter = stringArray[i];
505         if (currentCharacter === spaceCharacter) {
506             spaceFlag = true;
507         }
508     }
509
510     // if the spaceFlag is still set to false after the loop, then a space character
511     // doesn't exist, false is returned
512     return spaceFlag;
513 };
514
515 // preforms a look up check on the string comparing every character in the argument
516 // string
517 // to a comma character. true is returned if there is a match found in the string,
518 // false returned if there is no match.
519 Validation.stringLookupCheckForCommas = (stringValue) => {
520     let commaFlag = false;
521     // the comparison comma character
522     let commaCharacter = ",";
523     // The stringValue argument gets split up into an array of characters
524     let stringArray = stringValue.split("");
525     let currentCharacter = '';
526
527     // check for a number character in the stringValue
528     // by iterating through the stringArray and checking to see if a value matches
529     // any value in the commaCharacter
530     for (let i = 0; i < stringArray.length; i++) {
531         currentCharacter = stringArray[i];
532         if (currentCharacter === commaCharacter) {
533             commaFlag = true;
534         }
535     }
536     // if the commaFlag is still set to false after the loop, then a comma character
537     // doesn't exist, false is returned
538     return commaFlag;
539 };
540
541 Validation.stringLookupCheck = (stringValue, objectOfWhatNeedsToBeInTheString) => {
542     // if there is no object parameter an error message is thrown to
543     // notify that an object parameter is needed
544     if (objectOfWhatNeedsToBeInTheString === undefined) {
545         throw "ERROR NEED TO HAVE AN OBJECT PARAMETER: "

```

```

540         + "as the second parameter, set options of:\nlowerCase, upperCase, numbers:
541         "
542     " to true \nif you want to check the string contains one of them types of
543     characters ";
544 }
545
546 // checks to see if there is a property in the object passed as argument into
547 // the method
548 // called lowerCase and checks to see if it's value is set to the boolean value
549 // true
550 if (objectOfWhatNeedsToBeInTheString.lowerCase === true) {
551     // preforms lookup check to see if there is any lowerCase characters in the
552     // string
553     // if there is not then false gets returned
554     if(!Validation[_stringLookupCheckForLowerCase](stringValue)){ return false }
555 }
556
557 // checks to see if there is a property in the object passed as argument into
558 // the method
559 // called upperCase and checks to see if it's value is set to the boolean value
560 // true
561 if (objectOfWhatNeedsToBeInTheString.upperCase === true) {
562     // preforms lookup check to see if there is any upperCase characters in the
563     // string
564     // if there is not then false gets returned
565     if(!Validation[_stringLookupCheckForUpperCase](stringValue)){ return false }
566 }
567
568 // checks to see if there is a property in the object passed as argument into
569 // the method
570 // called numbers and checks to see if it's value is set to the boolean value true
571 if (objectOfWhatNeedsToBeInTheString.numbers === true) {
572     // preforms lookup check to see if there is any number characters in the
573     // string
574     // if there is not then false gets returned
575     if(!Validation[_stringLookupCheckForNumbers](stringValue)){ return false }
576 }
577
578 // if false hasn't been returned yet then the string must contain all criteria
579 return true;
580 }
581
582 // method checks that the string passed in as an argument only contains lower case
583 // characters, upper case characters
584 // or number characters. if a character is found in the string that is not one of
585 // them, then the method returns false
586 // if all characters in the string that is passed as argument are either upper,
587 // lower or numbers then true is returned
588 Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars = (stringValue,
589 allowSpaces) => {
590     // iterate through each character in the string checking if it is a number,
591     // lowercase char or an uppercase char
592     let curChar = '';
593     let charArray = stringValue.split(""); // puts the string into an array of its
594     // characters
595     for (let i = 0; i < charArray.length; i++) {
596
597         curChar = charArray[i];
598
599         if (allowSpaces === true) {
600             // if all of the checks come back false, false is returned
601             if ((!Validation[_stringLookupCheckForLowerCase](curChar))
602                 && (!Validation[_stringLookupCheckForUpperCase](curChar))
603                 && (!Validation[_stringLookupCheckForNumbers](curChar))
604                 && (!Validation[_stringLookupCheckForSpaces](curChar)))
605             {
606                 return false;
607             }
608         }
609
610         else {
611
612             if ((curChar >= 'A' && curChar <= 'Z') || (curChar >= 'a' && curChar <= 'z')) {
613                 return false;
614             }
615         }
616     }
617
618     return true;
619 }

```

```
597     // if all of the checks come back false, false is returned
598     if ((!Validation[_stringLookupCheckForLowerCase](curChar))
599         && (!Validation[_stringLookupCheckForUpperCase](curChar))
600         && (!Validation[_stringLookupCheckForNumbers](curChar)))
601     {
602         return false;
603     }
604 }
605
606 }
607
608 // loop has finished, if false hasn't been returned it means that all the
609 // characters
610 // in the string consist of either a number, lowercase char or an uppercase char
611 return true;
612 };
613
614
615
616
617 module.exports = Validation;
618
619
```

Validation

Sub Classes

Directory

Contents

AddNoteValidation.js

```

1  const Validation = require("./Validation");
2
3
4 // creating "private" properties for the class by utilising the Symbol function
5 const _note = Symbol("note");
6
7
8
9 // creating "private" method names for the class by utilising the Symbol function
10 const _presenceCheck = Symbol("presenceCheck");
11 const _stringTypeCheck = Symbol("stringTypeCheck");
12 const _rangeCheck = Symbol("rangeCheck");
13 const _basicCharactersOnlyLookUpCheck = Symbol("basicCharactersOnlyLookUpCheck");
14
15
16 class DashboardValidation {
17
18     constructor (note) {
19         this[_note] = note;
20     }
21
22     // presence check gets done on all the fields
23     [_presenceCheck] () {
24
25         let message = "      Please Ensure All Text Fields Have Data
26             Entered      ";
27
28         if (!Validation.presenceCheck(this[_note])) {
29             return message;
30         }
31
32         return '';
33     }
34
35
36
37     // type check gets done on all the fields to check that the value is of type
38     // string
39     [_stringTypeCheck] () {
40
41         let message = "      Please Ensure All Text Fields Are Of Type
42             String      ";
43
44         // if any of the text fields are not of type string then the message will
45         // get returned
46         if (!Validation.isaString(this[_note])) {
47             return message;
48         }
49         // if no messages have been returned, the method will return an empty string
50         return '';
51     }
52
53     // range check done on
54     [_rangeCheck] () {
55
56         let message = '';
57
58         // if text field does not contain a set of data within the string length
59         // range a message will get returned
60         if (!Validation.rangeCheck(this[_note], 1, 250, true)) {
61             message += "      Ensure That The Subject Field Is Within Range Of
62                 1 - 250 Characters      ";
63
64         }
65
66
67         // if no messages have been returned, the method will return an empty string
68         return message;
69     }
70
71
72     // Only allows basic characters to be accepted validation done to help

```

```
circumvent XSS and the user typing commas
68 [_basicCharactersOnlyLookUpCheck] () {
69
70     let message = "      Ensure That The Note Only Contain a-z, A-Z , And
71     0-9 Characters      ";
72
73     if (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_note],
74     true)){
75         return message;
76     }
77
78     return '';
79
80
81
82
83
84 // the public method that will be called on the instance of the class to
85 validate it and
86 // output any necessary messages if it failed the validation
87 checkFieldsAreValid() {
88
89     let outputMessage = '';
90
91     // appends the return message from private methods to the outputMessage
92     // variable in teh public method
93     outputMessage += this[_presenceCheck]();
94     outputMessage += this[_basicCharactersOnlyLookUpCheck] ();
95     outputMessage += this[_stringTypeCheck]();
96     outputMessage += this[_rangeCheck]();
97
98     // if the output message is comprised of just empty strings then the return
99     // value will be true
100    if (!Validation.presenceCheck(outputMessage)){
101        return true;
102    }
103
104    // if there is more than an empty string in the output message, the message
105    // will be returned
106    return outputMessage;
107
108
109 module.exports = DashboardValidation;
```

ContactsValidation.js

```

1 const Validation = require("./Validation");
2
3
4 // creating "private" properties for the class by utilising the Symbol function
5 const _businessName = Symbol("businessName");
6 const _firstName = Symbol("firstName");
7 const _lastName = Symbol("lastName");
8 const _email = Symbol("email");
9 const _telephone = Symbol("telephone");
10 const _city = Symbol("city");
11 const _postcode = Symbol("postcode");
12 const _address = Symbol("address");
13 const _status = Symbol("status");
14
15
16 // creating "private" method names for the class by utilising the Symbol function
17 const _presenceCheck = Symbol("presenceCheck");
18 const _lengthCheck = Symbol("lengthCheck");
19 const _stringTypeCheck = Symbol("stringTypeCheck");
20 const _numberTypeCheck = Symbol("numberTypeCheck");
21 const _rangeCheck = Symbol("rangeCheck");
22 const _emailFormatCheck = Symbol("emailFormatCheck");
23 const _lookUpCheckForStatus = Symbol("lookUpCheckForStatus");
24 const _basicCharactersOnlyLookUpCheck = Symbol("basicCharactersOnlyLookUpCheck");
25
26
27 class ContactsValidation {
28
29     constructor (businessName, firstName, lastName, email, telephone, city,
30     postcode, address, status) {
31         this[_businessName] = businessName;
32         this[_firstName] = firstName;
33         this[_lastName] = lastName;
34         this[_email] = email;
35         this[_telephone] = telephone;
36         this[_city] = city;
37         this[_postcode] = postcode;
38         this[_address] = address;
39         this[_status] = status;
40     }
41
42     // presence check gets done on all the fields
43     [_presenceCheck] () {
44
45         let message = " _____ Please Ensure All Text Fields Have Data
46         Entered _____ ";
47
48         if (!Validation.presenceCheck(this[_businessName])) {
49             return message;
50         }
51         if (!Validation.presenceCheck(this[_firstName])) {
52             return message;
53         }
54         if (!Validation.presenceCheck(this[_lastName])) {
55             return message;
56         }
57         if (!Validation.presenceCheck(this[_email])) {
58             return message;
59         }
60         if (!Validation.presenceCheck(this[_telephone])) {
61             return message;
62         }
63         if (!Validation.presenceCheck(this[_city])) {
64             return message;
65         }
66         if (!Validation.presenceCheck(this[_postcode])) {
67             return message;
68         }
69         if (!Validation.presenceCheck(this[_address])) {
70             return message;
71         }
72         if (!Validation.presenceCheck(this[_status])) {

```

```

72         return message;
73     }
74
75     return '';
76 }
77
78
79
80 // checks that the length of the telephone attribute of the class has a length
81 // equal to 11 characters
82 [_lengthCheck] () {
83     if (!Validation.lengthCheck(this[_telephone], 11)){
84         return " _____ Please Ensure The Telephone Length Is 11 Characters
85         Long _____";
86     }
87     else {
88         return '';
89     }
90 }
91
92 // checks to see if the telephone is comprised of only numbers, if it contains
93 // other characters then a message of
94 // what is wrong will be returned
95 [_numberTypeCheck] () {
96     if (!Validation.isaNumber(this[_telephone], true)){
97         return " _____ Please Ensure The Telephone Is Comprised Only Of
98         Number Characters _____";
99     }
100 }
101
102
103 // type check gets done on all the fields to check that the value is of type
104 // string
105 [_stringTypeCheck] () {
106     let message = " _____ Please Ensure All Text Fields Are Of Type
107     String _____";
108
109     // if any of the text fields are not of type string then the message will
110     // get returned
111     if (!Validation isaString(this[_businessName])) {
112         return message;
113     }
114     if (!Validation isaString(this[_firstName])) {
115         return message;
116     }
117     if (!Validation isaString(this[_lastName])) {
118         return message;
119     }
120     if (!Validation isaString(this[_email])) {
121         return message;
122     }
123     if (!Validation isaString(this[_telephone])) {
124         return message;
125     }
126     if (!Validation isaString(this[_city])) {
127         return message;
128     }
129     if (!Validation isaString(this[_postcode])) {
130         return message;
131     }
132     if (!Validation isaString(this[_address])) {
133         return message;
134     }
135     if (!Validation isaString(this[_status])) {
136         return message;
137     }
138
139     // if no messages have been returned, the method will return an empty string

```

```

138     return '';
139 }
140
141 // range check done on
142 [_rangeCheck] () {
143
144     let message = '';
145
146     // if text field does not contain a set of data within the string length
147     // range a message will get returned
148     if (!Validation.rangeCheck(this[_businessName], 1, 25,true)) {
149         message += " _____ Ensure That The Business Name Field Is Within
150         Range Of 1 - 25 Characters _____ ";
151     }
152     if (!Validation.rangeCheck(this[_firstName], 1, 15,true)) {
153         message += " _____ Ensure That The First Name Field Is Within Range
154         Of 1 - 15 Characters _____ ";
155     }
156     if (!Validation.rangeCheck(this[_lastName], 1, 15,true)) {
157         message += " _____ Ensure That The Last Name Field Is Within Range
158         Of 1 - 15 Characters _____ ";
159     }
160     if (!Validation.rangeCheck(this[_email], 4, 30,true)) {
161         message += " _____ Ensure That The Email Field Is Within Range Of 4
162         - 30 Characters _____ ";
163     }
164     if (!Validation.rangeCheck(this[_city], 1, 15,true)) {
165         message += " _____ Ensure That The City Field Is Within Range Of 1
166         - 15 Characters _____ ";
167     }
168
169     // if no messages have been returned, the method will return an empty string
170     return message;
171 }
172
173
174 [_emailFormatCheck] () {
175     if (!Validation.emailFormat(this[_email])){
176         return " _____ Please Ensure The Email Is Valid _____ ";
177     }
178     else {
179         return '';
180     }
181 }
182
183 // does a look up check to ensure that the contact status is one of the
184 // acceptable values
185 [_lookUpCheckForStatus] () {
186     let acceptableValues = ['lead', 'dead-lead', 'recontact-lead', 'client',
187     'past-client'];
188     if (!Validation.comparisonCheck(this[_status], acceptableValues)) {
189         return ' _____ Ensure That The Contact Status Is One Of The
190         Acceptable Values _____ ';
191     }
192     else {
193         return '';
194     }
195
196 // Only allows basic characters to be accepted validation done to help
197 // circumvent XSS and the user typing commas
198 [_basicCharactersOnlyLookUpCheck] () {
199
200     let message = " _____ Ensure That All Fields Other Than Email Only

```

```

Contain a-z, A-Z , Spaces, And 0-9 Characters _____";  

199  

200  

201     if  

202         (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_businessName], true)) {  

203             return message;  

204         }  

205         if  

206             (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_firstName],  

207                 true)) {  

208                 return message;  

209             }  

210             if  

211                 (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_lastName],  

212                     true)) {  

213                     return message;  

214                 }  

215                 if  

216                     (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_telephone],  

217                         true)) {  

218                         return message;  

219                     }  

220                     if  

221                         (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_city],  

222                             true)) {  

223                             return message;  

224                         }  

225                         return '';
226
227
228
229
230
231 // the public method that will be called on the instance of the class to  

232 // validate it and  

233 // output any necessary messages if it failed the validation  

234 checkFieldsAreValid() {
235
236     let outputMessage = '';
237
238     // appends the return message from private methods to the outputMessage  

239     // variable in teh public method
240     outputMessage += this[_presenceCheck]();
241     outputMessage += this[_lengthCheck]();
242     outputMessage += this[_numberTypeCheck]();
243     outputMessage += this[_stringTypeCheck]();
244     outputMessage += this[_lookUpCheckForStatus]();
245     outputMessage += this[_rangeCheck]();
246     outputMessage += this[_emailFormatCheck]();
247     outputMessage += this[_basicCharactersOnlyLookUpCheck]();
248
249     // if the output message is comprised of just empty strings then the return  

250     // value will be true
251     if (!Validation.presenceCheck(outputMessage)) {
252         return true;
253     }
254
255     // if there is more than an empty string in the output message the message  

256     // will be returned

```

```
254     return outputMessage;
255 }
256
257
258
259
260
261
262
263
264 }
265
266 module.exports = ContactsValidation;
267
```

DashboardValidation.js

```

1  const Validation = require("./Validation");
2
3
4 // creating "private" properties for the class by utilising the Symbol function
5 const _date = Symbol("date");
6 const _time = Symbol("time");
7 const _subject = Symbol("subject");
8 const _message = Symbol("message");
9
10
11 // creating "private" method names for the class by utilising the Symbol function
12 const _presenceCheck = Symbol("presenceCheck");
13 const _stringTypeCheck = Symbol("stringTypeCheck");
14 const _rangeCheck = Symbol("rangeCheck");
15 const _timeFormatCheck = Symbol("timeFormatCheck");
16 const _dateFormatCheck = Symbol("dateFormatCheck");
17 const _basicCharactersOnlyLookUpCheck = Symbol("basicCharactersOnlyLookUpCheck");
18
19
20 class DashboardValidation {
21
22     constructor (date, time, subject, message) {
23         this[_date] = date;
24         this[_time] = time;
25         this[_subject] = subject;
26         this[_message] = message;
27     }
28
29     // presence check gets done on all the fields
30     [_presenceCheck] () {
31
32         let message = " _____ Please Ensure All Text Fields Have Data
33             Entered _____ ";
34
35         if (!Validation.presenceCheck(this[_date])) {
36             return message;
37         }
38         if (!Validation.presenceCheck(this[_time])) {
39             return message;
40         }
41         if (!Validation.presenceCheck(this[_subject])) {
42             return message;
43         }
44         if (!Validation.presenceCheck(this[_message])) {
45             return message;
46         }
47
48         return '';
49     }
50
51
52     // checks that the date is in the format dd/mm/yyyy and that the date entered is
53     // either the current date or a date in the future
54     [_dateFormatCheck] () {
55         if (!Validation.dateFormat(this[_date], true)) {
56             return " _____ Please Ensure The Date Is In The Format dd/mm/yyyy
57             And The Date Is In The Future Or Today's Date _____ "
58         } else {
59             return '';
60         }
61     }
62
63     // checks that the time is in the format hh:mm
64     [_timeFormatCheck] () {
65         if (!Validation.timeFormat_HHMM(this[_time])) {
66             return " _____ Please Ensure The Time Is In The Format hh:mm Use
67             The 24hr Clock _____ "
68         } else {
69             return '';
70         }
71     }

```

```

70
71
72 // type check gets done on all the fields to check that the value is of type
73 string
74 [_stringTypeCheck] () {
75
76     let message = " _____ Please Ensure All Text Fields Are Of Type
77     String _____ ";
78
79     // if any of the text fields are not of type string then the message will
80     // get returned
81     if (!Validation.isaString(this[_date])) {
82         return message;
83     }
84     if (!Validation.isaString(this[_time])) {
85         return message;
86     }
87     if (!Validation.isaString(this[_subject])) {
88         return message;
89     }
90
91     if (!Validation.isaString(this[_message])) {
92         return message;
93     }
94
95
96 // range check done on
97 [_rangeCheck] () {
98
99     let message = '';
100
101    // if text field does not contain a set of data within the string length
102    // range a message will get returned
103    if (!Validation.rangeCheck(this[_subject], 1, 40, true)) {
104        message += " _____ Ensure That The Subject Field Is Within Range Of
105        1 - 40 Characters _____ ";
106    }
107    if (!Validation.rangeCheck(this[_message], 1, 200, true)) {
108        message += " _____ Ensure That The Message Field Is Within Range Of
109        1 - 200 Characters _____ ";
110    }
111
112
113
114
115 // Only allows basic characters to be accepted validation done to help
116 // circumvent XSS and the user typing commas
117 [_basicCharactersOnlyLookUpCheck] () {
118
119     let message = " _____ Ensure That The Subject And Message Only Contain
120     a-z, A-Z , And 0-9 Characters _____ ";
121
122     if
123     (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_subject],
124     true)){
125         return message;
126     }
127     if
128     (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_message],
129     true)){
130         return message;
131     }
132
133
134     return '';
135 }
```

```
131
132
133
134
135 // the public method that will be called on the instance of the class to
136 // validate it and
137 // output any necessary messages if it failed the validation
138 checkFieldsAreValid() {
139
140     let outputMessage = '';
141
142     // appends the return message from private methods to the outputMessage
143     // variable in teh public method
144     outputMessage += this[_presenceCheck]();
145     outputMessage += this[_basicCharactersOnlyLookUpCheck]();
146     outputMessage += this[_timeFormatCheck]();
147     outputMessage += this[_stringTypeCheck]();
148     outputMessage += this[_dateFormatCheck]();
149     outputMessage += this[_rangeCheck]();
150
151     // if the output message is comprised of just empty strings then the return
152     // value will be true
153     if (!Validation.presenceCheck(outputMessage)) {
154         return true;
155     }
156
157     // if there is more than an empty string in the output message, the message
158     // will be returned
159     return outputMessage;
160 }
161
162 module.exports = DashboardValidation;
```

PasswordValidation.js

```

1  const Validation = require("./Validation");
2
3
4 // creating "private" properties for the class by utilising the Symbol function
5 const _password = Symbol("password");
6 const _confirmPassword = Symbol("confirmPassword");
7
8
9 // creating "private" method names for the class by utilising the Symbol function
10 const _presenceCheck = Symbol("presenceCheck");
11 const _stringTypeCheck = Symbol("stringTypeCheck");
12 const _rangeCheck = Symbol("rangeCheck");
13 const _basicCharactersOnlyLookUpCheck = Symbol("basicCharactersOnlyLookUpCheck");
14 const _stringLookUpCheck = Symbol("stringLookUpCheck");
15 const _comparisonCheck = Symbol("comparisonCheck");
16
17
18
19 class PasswordValidation {
20
21     constructor (password, confirmPassword) {
22         this[_password] = password;
23         this[_confirmPassword] = confirmPassword;
24     }
25
26     // presence check gets done on all the fields
27     [_presenceCheck] () {
28
29         let message = "      Please Ensure All Text Fields Have Data
30             Entered      ";
31
32         if (!Validation.presenceCheck(this[_password])) {
33             return message;
34         }
35         if (!Validation.presenceCheck(this[_confirmPassword])) {
36             return message;
37         }
38
39         return '';
40     }
41
42
43
44     // type check gets done on all the fields to check that the value is of type
45     // string
46     [_stringTypeCheck] () {
47
48         let message = "      Please Ensure All Text Fields Are Of Type
49             String      ";
50
51         // if any of the text fields are not of type string then the message will
52         // get returned
53         if (!Validation.isaString(this[_password])) {
54             return message;
55         }
56
57         // if no messages have been returned, the method will return an empty string
58         return '';
59     }
60
61
62     // range check done on
63     [_rangeCheck] () {
64
65         let message = "      Ensure That The password Field Is Within Range Of
66             8 - 15 Characters      ";
67
68         // if text field does not contain a set of data within the string length
69         // range a message will get returned
70         if (!Validation.rangeCheck(this[_password], 8, 15, true)) {
71             return message;
72         }

```

```

68
69
70     // if no messages have been returned, the method will return an empty string
71     return '';
72 }
73
74
75 // Only allows basic characters to be accepted validation done to help
76 // circumvent XSS and the user typing commas
77 [_basicCharactersOnlyLookUpCheck] () {
78
79     let message = "      _____ Ensure That The Note Only Contain a-z, A-Z , And
80     0-9 Characters      _____ ";
81
82     if
83     (!Validation.checkStringIsMadeOfOnlyLowerUpperAndNumberChars(this[_password],
84     true)){
85         return message;
86     }
87
88     return '';
89 }
90
91
92 // checks that the password has at least one: uppercase character, lowercase
93 // character and number
94 [_stringLookUpCheck] () {
95
96     let message = "      _____ Ensure That Password Consists Of At Least One:
97     Uppercase Character, Lowercase Character And Number      _____ ";
98
99     if (!Validation.stringLookupCheck(this[_password], {lowercase: true,
100    uppercase: true, numbers: true})){
101         return message;
102     }
103
104     return '';
105 }
106
107
108
109
110
111
112 // will validate a the password to meet the specified criteria:
113 // greater than 8 characters and less than 20 characters , consisting of only
114 // numbers,
115 // uppercase letters and lowercase letters, and it contains at least one number,
116 // at least one number, at least one uppercase letter and at least one lowercase
117 // letter
118 // a comparison check is done lastly to ensure equality between both fields
119 // relevant messages will be returned if the password is not valid
120 // true is returned if the password is valid. can do this because JS is a
121 // dynamic type language
122
123
124 // the public method that will be called on the instance of the class to
125 // validate it and
126 // output any necessary messages if it failed the validation
127 checkFieldsAreValid() {
128
129     let outputMessage = '';
130
131     // appends the return message from private methods to the outputMessage
132     // variable in teh public method
133     outputMessage += this[_presenceCheck]();
134     outputMessage += this[_basicCharactersOnlyLookUpCheck]();

```

```
129     outputMessage += this[_stringTypeCheck]();
130     outputMessage += this[_rangeCheck]();
131     outputMessage += this[_stringLookUpCheck]();
132     outputMessage += this[_comparisonCheck]();
133 
134     // if the output message is comprised of just empty strings then the return
135     // value will be true
136     if (!Validation.presenceCheck(outputMessage)) {
137         return true;
138     }
139 
140     // if there is more than an empty string in the output message, the message
141     // will be returned
142     return outputMessage;
143 }
144 
145 
146 module.exports = PasswordValidation;
147 
148 
```

Public Directory Contents

Scripts

Directory

Contents

AnalyticsScripts.js

```

1 // store the graph canvas elements into variables
2 let clientsGained = document.querySelector('#clients-gained').getContext('2d');
3 let clientsLost = document.querySelector('#clients-lost').getContext('2d');
4
5
6 // // dummy test data for the graphs
7 // let clientsGainedLabels = ['January', 'February', 'March', 'April', 'May',
8 // 'June', 'July', 'August', 'September', 'October', 'November', 'December'];
9 // let clientsGainedDataPoints = [20, 32, 34, 45, 43, 35, 24, 21, 14, 12, 19, 16];
10 // //
11 // let clientsLostLabels = ['January', 'February', 'March', 'April', 'May', 'June',
12 // 'July', 'August', 'September', 'October', 'November', 'December'];
13 // let clientsLostDataPoints = [20, 32, 45, 23, 54, 21, 25, 31, 45, 17, 11, 8];
14
15
16 // when time scale option gets selected the form's action gets changed to the
17 // selected option's id
18 function changeAction() {
19     let timeScale = $('#time-scale').val();
20     console.log(timeScale);
21     document.querySelector("#time-scale-form").action = "/analytics/" + timeScale;
22     // line that changes the action attribute of the time scale form
23 }
24
25
26
27 // creates the graph for clients gained using the chart.js library
28 let clientsGainedGraph = new Chart(clientsGained, {
29     type: 'line',
30     data: {
31         labels: clientsGainedLabels,
32         datasets: [{
33             label: 'Number Of Clients Gained',
34             data: clientsGainedDataPoints,
35             backgroundColor: 'lightgreen',
36             borderWidth: 1,
37             borderColor: '#777',
38         }]
39     },
40     options: {
41         responsive: true,
42         maintainAspectRatio: false,
43         title: {
44             display: true,
45             text: 'Clients Gained',
46             fontSize: 25,
47         },
48         legend: {
49             display: false
50         },
51     },
52 });
53
54 // creates the graph for clients lost using the chart.js library
55 let clientsLostGraph = new Chart(clientsLost, {
56     type: 'line',
57     data: {
58         labels: clientsLostLabels,
59         datasets: [{
60             label: 'Number Of Clients Lost',
61             data: clientsLostDataPoints,
62             backgroundColor: 'lightcoral',
63             borderWidth: 1,
64             borderColor: '#777',
65         }]
66     },
67     options: {
68         responsive: true,
69         maintainAspectRatio: false,

```

```
70     title: {
71         display: true,
72         text: 'Clients Lost',
73         fontSize: 25,
74     },
75     legend:{  
76         display: false  
77     },  
78 }  
79 );  
80  
81 // // creates the graph for number of clients using the chart.js library  
82 // let numberOfClientsGraph = new Chart(numberOfClients , {  
83 //     type: 'line',  
84 //     data: {  
85 //         labels: numberOfClientsLabels,  
86 //         datasets:[{  
87 //             label:'Number Of Clients',  
88 //             data: numberOfClientsDataPoints,  
89 //             backgroundColor: 'lightblue',  
90 //             borderWidth: 1,  
91 //             borderColor: '#777',  
92 //         }]  
93 //     },  
94 //     options: {  
95 //         responsive: true,  
96 //         maintainAspectRatio: false,  
97 //         title: {  
98 //             display: true,  
99 //             text: 'Number Of Clients',  
100 //             fontSize: 25,  
101 //         },  
102 //         legend:{  
103 //             display: false  
104 //         },  
105 //     }  
106 // });
```

ChangePasswordScripts.js

Kieran Marcus Williams

```
1
2
3 //toggles the type in the password text boxes between password and text
4 // activated by the check box onClick()
5 function showPassword() {
6
7     let oldPasswordField = document.querySelector("#old-password");
8     let newPasswordField = document.querySelector("#new-password");
9     let confirmPasswordField = document.querySelector("#confirm-password");
10
11    if(oldPasswordField.type === "password" && newPasswordField.type === "password"
12        && confirmPasswordField.type === "password") {
13        oldPasswordField.type = "text";
14        newPasswordField.type = "text";
15        confirmPasswordField.type = "text";
16    }
17    else {
18        oldPasswordField.type = "password";
19        newPasswordField.type = "password";
20        confirmPasswordField.type = "password";
21    }
22}
```

DashboardScripts.js

Kieran Marcus Williams

```
1 // function that prevents the enter/ return key from being used
2 function returnKeyBlocker(e) {
3     // sets e to be equal to either the event passed as argument or the window.event
4     // object for IE support
5     e = e || window.event;
6     // gets the key of e but if that is deprecated or isn't supported by the browser
7     // then then charCode is used
8     let keyCode = e.keyCode || e.charCode;
9     return keyCode !== 13;
}
```

EncryptionNotesPasswordScripts.js

```
1
2
3 //toggles the type in the password text boxes between password and text
4 // activated by the check box onClick()
5 function showPassword() {
6
7     let PasswordField = document.querySelector("#password");
8     let confirmPasswordField = document.querySelector("#confirmPassword");
9
10    if(PasswordField.type === "password" && confirmPasswordField.type === "password") {
11        PasswordField.type = "text";
12        confirmPasswordField.type = "text";
13    }
14    else {
15        PasswordField.type = "password";
16        confirmPasswordField.type = "password";
17    }
18}
19
```

IndividualContactScripts.js

```
1 // function that prevents the enter/ return key from being used
2 function returnKeyBlocker(e) {
3     // sets e to be equal to either the event passed as argument or the window.event
4     // object for IE support
5     e = e || window.event;
6     // gets the key of e but if that is deprecated or isn't supported by the browser
7     // then then charCode is used
8     let keyCode = e.keyCode || e.charCode;
9     return keyCode !== 13;
}
```

Stylesheets

Directory

Contents

AnalyticsStyles.css

```
1 .top-half-of-analytics {
2     height: 450px;
3     display: flex;
4     justify-content: space-around;
5 }
6
7 .bottom-half-of-analytics {
8     display: flex;
9     justify-content: space-between;
10}
11
12 .middle-bottom-analytics {
13     width: 250px;
14 }
15
16 .custom-select {
17     width: 150px;
18 }
19
20 .time-scale-separator {
21     display: flex;
22     justify-content: space-between;
23 }
24
25 .conversion-rate-holder {
26     padding-top: 20px;
27     width: 400px;
28     display: flex;
29     justify-content: space-between;
30 }
31
32
33 .remove-margin {
34     margin: 0;
35 }
36
37 .conversion-rate-table {
38     width: 100%;
39     height: 75px;
40     border: 1px solid black;
41 }
42
43 .row-of-buttons {
44     width: 250px;
45     margin: 10px;
46     display: flex;
47     justify-content: space-between;
48 }
49
50 .button-holder-analytics {
51     width: fit-content;
52 }
53
54 td {
55     border: 1px solid black;
56     text-align: center;
57     padding: 10px;
58 }
59
60 body {
61     margin: 40px 80px;
62 }
63
64
65 canvas {
66     width: 35vw;
67 }
68
69
70
71
72
73
```

ChangePasswordStyles.css

```
1 body {
2     margin: 40px 40px;
3 }
4
5 .container {
6     width: 800px;
7 }
8
9
10 .form-control {
11     border: 1px solid grey;
12 }
13
14 .custom-select {
15     border: 1px solid grey;
16     height: 40px;
17 }
18
19 .btn-submit {
20     position: relative;
21     margin-top: 40px;
22     margin-left: 20%;
23     margin-right: 20%;
24     width: 60%;
25     height: 50px;
26 }
27
28
```

ContactsListStyles.css

```
1 body {
2     margin: 40px 40px;
3 }
4
5 #modify-list-results-form {
6
7     width: fit-content;
8     padding: 10px;
9 }
10
11 #contact-management {
12     margin-top: 30px;
13     width: fit-content;
14     padding: 10px;
15 }
16
17 #left-side-container {
18
19     height: 700px;
20     margin: 0;
21     display: flex;
22     flex-direction: column;
23     justify-content: space-between;
24 }
25
26 #right-side-container {
27     margin-right: 2%;
28     height: 600px;
29     width: 1000px;
30     overflow: scroll;
31 }
32
33
34 .table {
35     position: inherit;
36     top: 0;
37     bottom: 0;
38     left: 0;
39     right: 0;
40     width: 100%;
41 }
42
43
44 .holder {
45     margin: 30px 0;
46     width: 100%;
47     display: flex;
48     justify-content: space-between;
49 }
50
51 .submit-col {
52     display: flex;
53 }
```

CreateContactsStyles.css

```
1 body {
2     margin: 40px 40px;
3 }
4
5 .container {
6     width: 800px;
7 }
8
9 .form {
10    margin-top: 20px;
11    padding: 30px;
12 }
13
14 .form-control {
15     border: 1px solid grey;
16 }
17
18 .custom-select {
19     border: 1px solid grey;
20     height: 40px;
21 }
22
23 .btn-submit {
24     position: relative;
25     margin-top: 40px;
26     margin-left: 20%;
27     margin-right: 20%;
28     width: 60%;
29     height: 50px;
30 }
```

DashboardStyles.css

```
1 #dashboard-separator {
2   display: flex;
3   flex-direction: row;
4   justify-content: space-between;
5 }
6
7 #schedule-task-form {
8   margin: 15px;
9 }
10
11 #today {
12   margin: 15px;
13 }
14
15 #future {
16   margin: 15px;
17 }
18
19 .scheduled-display {
20   display: flex;
21 }
22
23 .trash-button {
24   color: darkgray;
25 }
26
27 .trash-button:hover {
28
29   color: grey;
30 }
31
32
33
34 .dashboard-slots {
35   border: solid 1px grey;
36   padding: 30px 25px;
37   width: 400px;
38   height: 700px;
39   overflow-y: scroll;
40   word-wrap: break-word;
41 }
42
43
44 ul {
45   list-style: none;
46   margin: 0;
47   padding: 0;
48 }
49
50 textarea {
51   resize: none;
52 }
53
54 body {
55   margin: 40px 80px;
56 }
```

EditContactStyles.css

```
1 body {
2     margin: 40px 40px;
3 }
4
5 .container {
6     width: 800px;
7 }
8
9
10
11 .form-control {
12     border: 1px solid grey;
13 }
14
15 .custom-select {
16     border: 1px solid grey;
17     height: 40px;
18 }
19
20 .btn-submit {
21     position: relative;
22     margin-top: 40px;
23     margin-left: 20%;
24     margin-right: 20%;
25     width: 60%;
26     height: 50px;
27 }
```

HomeStyles.css

```
1 body {
2     margin: 20px 40px;
3 }
4
5 h1 {
6     margin-top: 60px;
7 }
8
9 .link-separator {
10    margin-left: 20px;
11 }
12
13 #login-btn {
14     margin-right: 40px;
15 }
16
17
18
19
20
21
22
```

IndividualContactStyles.css

Kieran Marcus Williams

```
1 #halfway-separator {
2     display: flex;
3     justify-content: space-between;
4 }
5
6 #left-side {
7     display: flex;
8     flex-direction: column;
9     justify-content: space-around;
10 }
11
12 .contact-details{
13     width: fit-content;
14 }
15
16 .add-note-section {
17     width: fit-content;
18 }
19
20 .btn-submit-add-note {
21     margin-top: 10px;
22     float: right;
23 }
24
25 .btn-container-extra-notes {
26     float: right;
27     margin-top: 10px;
28     width: 40%;
29 }
30
31 #extra-notes-btn-group {
32     display: flex;
33     justify-content: space-evenly;
34 }
35
36 .btn-submit-extra-notes {
37 }
38
39 .btn-submit-extra-notes-encrypt {
40 }
41
42 span {
43     font-weight: bold;
44     font-size: x-large;
45     margin-right: 1em;
46 }
47
48 .contact-management-buttons {
49     margin-top: 30px;
50     width: 40%;
51     display: flex;
52 }
53
54 .btn-edit-contact {
55     margin-right: 20%;
56 }
57
58 .table {
59     position: inherit;
60     top: 0;
61     bottom: 0;
62     left: 0;
63     right: 0;
64     width: 100%;
65 }
66
67 .td-borededer {
68     border-left: 1px solid lightgrey;
69 }
70
71 .td-borededer {
72     border-left: 1px solid lightgrey;
73 }
```

```
74  
75 .thead-grey {  
76     background-color: lightgrey;  
77 }  
78  
79  
80 #table-holder-notes{  
81     width: 680px;  
82     height: 400px;  
83     overflow: scroll;  
84 }  
85  
86 textarea {  
87     resize: none;  
88 }  
89  
90 body {  
91     margin: 40px 60px;  
92 }  
93
```

NavbarLoggedInStyles.css

```
1  .separator {
2      width: 100%;
3      display: flex;
4      justify-content: space-between;
5  }
6
7
8  .nav-item{
9      font-size: large;
10     margin: 0 20px;
11 }
12
13 .small-margin-right {
14     margin-left: 20px;
15 }
16
17 .mid-nav{
18     width: 50%;
19 }
20
21 .spread-nav-items{
22     width: 100%;
23     display: flex;
24     justify-content: space-around;
25 }
```

SettingsStyles.css

```
1 body {  
2     margin: 20px 40px;  
3 }  
4  
5 h1 {  
6     margin-top: 60px;  
7 }
```

Routes

Directory

Contents

analyticsRoutes.js

```

1  const express = require("express");
2  const router = express.Router();
3  const CSVHandler = require("../classes/CSVHandler");
4  const DateHandler = require("../classes/DateHandler");
5  const Analyticals = require("../classes/Analyticals");
6
7
8
9
10 ///////////////////////////////////////////////////////////////////
11 ///////////////////////////////////////////////////////////////////
12 // GET ROUTES
13 ///////////////////////////////////////////////////////////////////
14 ///////////////////////////////////////////////////////////////////
15
16
17 router.get("/", (req, res) => {
18   res.redirect("/analytics/daily");
19 });
20
21
22 // route that will give graph data for the past 14 days
23 router.get("/daily", (req, res) => {
24
25   let clientsGainedFile = new CSVHandler(__dirname +
26     '/dataStoreFiles/clientsGained.csv');
27   let clientsLostFile = new CSVHandler(__dirname +
28     '/dataStoreFiles/clientsLost.csv');
29   let leadsLostFile = new CSVHandler(__dirname + '/dataStoreFiles/leadsLost.csv');
30
31   // find the conversion rate for the month in ratio and percentage ==> put into
32   // the ejs file
33   let conversionRateObj = analytics.GetConversionRateForTheMonth();
34
35   // find the data points and labels for the past 14 days ==> put into the ejs file
36   let dataPointsAndLabels = analytics.getDataPointsAndLabelsForThePast14Days(); // //
37   // will store data points and labels for clients gained and lost in data points
38
39   res.render("Analytics", {conversionRate: conversionRateObj, graphData:
40   dataPointsAndLabels, timeScale: "daily"});
41
42
43
44
45
46 // route that will give graph data for the past 12 weeks
47 router.get("/weekly", (req, res) => {
48
49   let clientsGainedFile = new CSVHandler(__dirname +
50     '/dataStoreFiles/clientsGained.csv');
51   let clientsLostFile = new CSVHandler(__dirname +
52     '/dataStoreFiles/clientsLost.csv');
53   let leadsLostFile = new CSVHandler(__dirname + '/dataStoreFiles/leadsLost.csv');
54
55   let analytics = new Analyticals(clientsGainedFile, clientsLostFile,
56     leadsLostFile);
57
58   // find the conversion rate for the month in ratio and percentage ==> put into
59   // the ejs file
60   let conversionRateObj = analytics.GetConversionRateForTheMonth();
61
62   // find the data points and labels for the past 14 days ==> put into the ejs file
63   let dataPointsAndLabels = analytics.getDataPointsAndLabelsForThePast12Weeks(); //
64   // will store data points and labels for clients gained and lost in data points

```

```

61
62
63     res.render("Analytics", {conversionRate: conversionRateObj, graphData:
64         dataPointsAndLabels, timeScale: "weekly"});
65
66
67
68
69 // route that will give graph data for the past 12 months
70 router.get("/monthly", (req, res) => {
71
72     let clientsGainedFile = new CSVHandler(__dirname +
73         '/dataStoreFiles/clientsGained.csv');
74     let clientsLostFile = new CSVHandler(__dirname +
75         '/dataStoreFiles/clientsLost.csv');
76     let leadsLostFile = new CSVHandler(__dirname + '/dataStoreFiles/leadsLost.csv');
77
78     let analytics = new Analyticals(clientsGainedFile, clientsLostFile,
79         leadsLostFile);
80
81     // find the conversion rate for the month in ratio and percentage ==> put into
82     // the ejs file
83     let conversionRateObj = analytics.GetConversionRateForTheMonth();
84
85     // find the data points and labels for the past 14 days ==> put into the ejs file
86     let dataPointsAndLabels = analytics.getDataPointsAndLabelsForThePast12Months();
87     // will store data points and labels for clients gained and lost in data points
88
89
90
91
92
93 //////////////////////////////////////////////////////////////////
94 //////////////////////////////////////////////////////////////////
95 //////////////////////////////////////////////////////////////////
96 //////////////////////////////////////////////////////////////////
97 //////////////////////////////////////////////////////////////////
98 //////////////////////////////////////////////////////////////////
99
100 // route that will add a record to clients gained file
101 router.get("/clientGained", (req, res) => {
102     let clientsGainedFile = new CSVHandler(__dirname +
103         '/dataStoreFiles/clientsGained.csv');
104     let monthAndYear = DateHandler.getCurrentDateTime();
105     // makes the month and year ==> yyyy-mm and puts it back into a string
106     monthAndYear = (monthAndYear.split(""));
107     monthAndYear.splice(7,18);
108     monthAndYear = monthAndYear.join("");
109
110     clientsGainedFile.addRecord([monthAndYear], true, true);
111     res.redirect("/analytics");
112 }
113
114
115 // route that will add a record to clients lost file
116 router.get("/clientLost", (req, res) => {
117     let clientsLostFile = new CSVHandler(__dirname +
118         '/dataStoreFiles/clientsLost.csv');
119     let monthAndYear = DateHandler.getCurrentDateTime();
120     // makes the month and year ==> yyyy-mm and puts it back into a string
121     monthAndYear = (monthAndYear.split(""));
122     monthAndYear.splice(7,18);
123     monthAndYear = monthAndYear.join("");

```

```

123
124     clientsLostFile.addRecord([monthAndYear], true, true);
125     res.redirect("/analytics");
126   });
127
128
129
130 // route that will add a record to leads lost file
131 router.get("/leadLost", (req, res) => {
132   let leadsLostFile = new CSVHandler(__dirname + '/dataStoreFiles/leadsLost.csv');
133   let monthAndYear = DateHandler.getCurrentDateTime();
134   // makes the month and year ==> yyyy-mm and puts it back into a string
135   monthAndYear = (monthAndYear.split(""));
136   monthAndYear.splice(7,18);
137   monthAndYear = monthAndYear.join("");
138
139   leadsLostFile.addRecord([monthAndYear], true, true);
140   res.redirect("/analytics");
141 });
142
143
144 ///////////////////////////////////////////////////
145 ///////////////////////////////////////////////////
146 //                                         DELETE ROUTES
147 ///////////////////////////////////////////////////
148 ///////////////////////////////////////////////////
149
150
151 router.delete("/analytics/removeAllData", (req, res) => {
152
153   // delete records
154   let leadsLostFile = new CSVHandler(__dirname + '/datastoreFiles/leadsLost.csv');
155   let clientsLostFile = new CSVHandler(__dirname +
156     '/datastoreFiles/clientsLost.csv');
157   let clientsGainedFile = new CSVHandler(__dirname +
158     '/datastoreFiles/clientsGained.csv');
159
160   leadsLostFile.deleteAllRecords();
161   clientsLostFile.deleteAllRecords();
162   clientsGainedFile.deleteAllRecords();
163
164   // redirect to the analytics page
165   res.redirect("/analytics")
166
167 });
168 module.exports = router;

```

contactsRoutes.js

```

1 const
2   express = require("express"),
3   router = express.Router(),
4   CSVHandler = require("../classes/CSVHandler"),
5   StringUtility = require('../utilityMethods/StringUtility'),
6   DateHandler = require("../classes/DateHandler"),
7   Sort = require("../classes/Sort"),
8   Encryption = require("../classes/Encryption"),
9   PasswordValidation = require("../classes>PasswordValidation"),
10  ContactsValidation = require("../classes/ContactsValidation"),
11  AddNoteValidation = require("../classes/AddNoteValidation");
12
13
14 ///////////////////////////////////////////////////////////////////
15 ///////////////////////////////////////////////////////////////////
16 // GET ROUTES
17
18 ///////////////////////////////////////////////////////////////////
19
20
21
22
23
24 //renders the contacts list page that outputs all the contacts in a table
25 router.get("/", (req, res) => {
26   let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
27   let allContactRecords = contactsFile.getAllRecords(); // gets all the contact
28   records and stores in a 2D array
29   let message = req.session.message;
30   let errorMessage = req.session.errorMessage;
31
32   // an object with a property containing the 2D Array is passed to the ejs template
33   // searchValue: undefined allows me to use the same ejs page in the
34   .post("/search") route
35   res.render("ContactsList", {contactRecords: allContactRecords, titleAddition:
36   undefined, flashMessage: message, errorMessage: errorMessage},
37   function (err, html) {
38     req.session.message = undefined;
39     req.session.errorMessage = undefined;
40     res.send(html);
41   });
42 });
43
44
45 // downloads the contacts file then redirects to the base route
46 router.get("/download", (req, res) => {
47
48   res.download(__dirname + '/dataStoreFiles/contacts.csv',
49   'contactRecords' + ':' + DateHandler.getCurrentDateTime() + '.csv', // sets
50   the name to be contacts records with the current date in with the title
51   function(err){
52     if (err) {
53       console.log(err); // logs the error to the console
54     } else {
55       console.log("downloaded: " + __dirname + '/dataStoreFiles/contacts.csv');
56     }
57   });
58 });
59
60
61
62 // renders the individual contact page for the contact with the id in the url
63 router.get("/viewContact:id", (req, res) => {
64
65   let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
66   let notesFile = new CSVHandler(__dirname + '/dataStoreFiles/contactsNotes.csv');
67   let extraNotesFile = new CSVHandler(__dirname +
68   '/dataStoreFiles/contactsExtraNotes.csv');
69   let id = req.params.id;
70   // find contact by id to get the contacts record

```

```

67 let contactDetailsRecord = contactsFile.selectRecordByID(id);
68
69
70 // gets all the notes relating to the current contact that is being viewed
71 // stored in the contactNotes variable as a 2D array with each inner array being
72 // a record from the notes csv file
73 let contactsNotes = notesFile.getRecordsOnlyByField([3], id);
74
75
76 let contactsExtraNotes = extraNotesFile.getRecordsOnlyByField([2], id); // gets
77 // the contacts extra notes
78
79 contactsNotes = Sort.bubbleSort2D(contactsNotes, [1], false); // sorts the notes
80 // ascending by their date time stamp
81
82
83 if (contactsExtraNotes.length !== 0) { // if there is data in the contacts extra
84 // notes 2d array
85 contactsExtraNotes = contactsExtraNotes[0][3]; // gets the data from the
86 // extra notes data column
87 }
88
89 if (req.session.decryptedData !== undefined) {
90 contactsExtraNotes = req.session.decryptedData;
91 }
92
93
94
95 let flashMessage = req.session.message;
96 let errorMessage = req.session.errorMessage;
97
98 res.render("ViewIndividualContact",
99 {
100   contactDetails: contactDetailsRecord[0].record,
101   flashMessage: flashMessage,
102   errorMessage: errorMessage,
103   contactNotes: contactsNotes,
104   extraNotes: contactsExtraNotes
105 },
106 (err, html) => {
107   req.session.message = undefined;
108   req.session.errorMessage = undefined;
109   req.session.decryptedData = undefined;
110   res.send(html);
111 }
112 );
113
114 //renders the create contact page
115 router.get("/new", (req, res) => {
116   let message = req.session.message; // if there is a message in the session
117   // object then it will be outputted to the user
118   let errorMessage = req.session.errorMessage;
119
120   let businessName = req.session.businessName;
121   let firstName = req.session.firstName;
122   let lastName = req.session.lastName;
123   let email = req.session.email;
124   let telephone = req.session.telephone;
125   let city = req.session.city;
126   let postcode = req.session.postcode;
127   let address = req.session.address;
128
129   let valuesStoredInSessionFromBody = [businessName, firstName, lastName, email,
130   telephone, city, postcode, address];
131
132   res.render("CreateContact",
133   {

```



```

200
201 // post route to add a record to the contacts page
202
203 router.post("/", (req, res) => {
204     let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
205
206     // stores the values from the text fields into variables
207     // the some of the values get capitalized to aid the data maintainability
208     let businessName = StringUtility.capitalizeString(req.body.businessName);
209     let firstName = StringUtility.capitalizeString(req.body.firstName);
210     let lastName = StringUtility.capitalizeString(req.body.lastName);
211     let email = req.body.email;
212     let telephone = req.body.telephone;
213     let city = StringUtility.capitalizeString(req.body.city);
214     let postcode = req.body.postcode;
215     let address = StringUtility.capitalizeString(req.body.address);
216     let status = req.body.status;
217
218     // do validation here
219
220     let Validation = new ContactsValidation(businessName,firstName, lastName, email,
221         telephone, city, postcode, address, status);
222     let validationResult = Validation.checkFieldsAreValid(); // this will either
223     // store the boolean value true or a message of what needs to be fixed
224
225     if (validationResult === true) {
226         // if it passes validation, add record and redirect to contacts list page
227         contactsFile.addRecord([businessName, firstName, lastName, email, telephone,
228             city, postcode, address, status]);
229         req.session.message = "Successfully Created A Contact";
230         res.redirect("/contacts/");
231     }
232
233     else {
234         // if something does not pass the validation then the user will get
235         // redirected to the
236         // create contacts page and a req.session.message will get set to the
237         // validationResult output
238         // this can then be accessed at the /contacts/new route to notify the user
239         // where they went wrong
240
241         req.session.message = validationResult; // validation message
242
243         // there will also be other properties attached to the session memory
244         // property containing the entered form data
245         // so that the fields don't go blank if any of the user's inputs did not
246         // pass the validation
247         req.session.businessName = businessName;
248         req.session.firstName = firstName;
249         req.session.lastName = lastName;
250         req.session.email = email;
251         req.session.telephone = telephone;
252         req.session.city = city;
253         req.session.postcode = postcode;
254         req.session.address = address;
255
256         res.redirect("/contacts/new")
257     }
258 }
259 });
260
261 //renders the contacts list page that outputs any contacts that match the search value
262 router.post("/search", (req, res) => {
263     let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
264
265     let searchBarValue = req.body.searchBar; // gets the search value from the form

```

```

265 // finds all matches of the search value by looking in every column for one
266 let matchingRecords =
267 contactsFile.getRecordsOnlyByField([0,1,2,3,4,5,6,7,8,9,10], searchValue);
268
269 // if there is no match then do the search again but with the search value
270 // capitalized
271 if (matchingRecords.length === 0) {
272     searchValue = StringUtility.capitalizeString(searchValue);
273     matchingRecords =
274     contactsFile.getRecordsOnlyByField([0,1,2,3,4,5,6,7,8,9,10], searchValue);
275
276     // if there is still no match, the user gets redirected to the contacts list
277     // page with all the contacts
278     if (matchingRecords.length === 0) {
279         req.session.errorMessage = "No Matches found";
280         res.redirect("/contacts/");
281     } else {
282
283         res.render("ContactsList", {
284             contactRecords: matchingRecords,
285             titleAddition: "Searched for: " + searchValue,
286             flashMessage : matchingRecords.length.toString() + " Matches Found",
287             errorMessage: undefined
288         });
289     }
290 } else {
291     res.render("ContactsList", {
292         contactRecords: matchingRecords,
293         titleAddition: "Searched for: " + searchValue,
294         flashMessage : matchingRecords.length.toString() + " Matches Found",
295         errorMessage: undefined
296     });
297
298
299
300
301
302 router.post("/sortBy", (req, res) => {
303     let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
304     let sortByValue = req.body.sortBy; // will return undefined if no value is
305     selected
306     let ascOrDec = req.body.order; // will be either ascending or descending
307     let sortedArray = [];
308
309     // turns the names gotten back from the sort form into the index that the
310     // records will need to be sorted by
311     switch (sortByValue) {
312         case "date":
313             sortByValue = {index: 1, label: "Date"}; // date is stored at index 1 in
314             the contacts csv file
315             break;
316         case "city":
317             sortByValue = {index: 7, label: "City"}; // city is stored at index 7 in
318             the contacts csv file
319             break;
320         case "businessName":
321             sortByValue = {index: 2, label: "Business Name"}; // business name is at
322             index 2 in the contacts csv file
323             break;
324         default:
325             // if undefined is returned because there is no value selected in the form
326             // then the sortByValue will be set to the value 3 which is the index of
327             // the first name
328             sortByValue = {index: 3, label: "First Name"};
329     }
330
331     // turns the ascending/ descending option into a boolean value which can be used
332     // by the sortBy method
333     switch (ascOrDec) {

```

```

327     case "ascending":
328         ascOrDec = true;
329         break;
330     case "descending":
331         ascOrDec = false;
332         break;
333     default:
334         ascOrDec = false;
335     }
336
337     // sort the array by the sortByValue in ascending or descending order dependent
338     // on the ascOrDec boolean argument that
339     // gets passed in. the sorted array gets stored into the sortedArray variable
340     sortedArray = contactsFile.sortBy(sortByValue.index, ascOrDec);
341
342     res.render("ContactsList",
343     {
344         contactRecords: sortedArray,
345         titleAddition: "Sorted By: " + sortByValue.label,
346         flashMessage: undefined,
347         errorMessage: undefined
348     });
349 }
350
351
352
353
354
355
356
357 // this route is used to add a note to the contactNotes file
358 router.post("/addnote:id", (req, res) => {
359     let contactsNotes = new CSVHandler(__dirname +
360         '/dataStoreFiles/contactsNotes.csv');
361
362     let contactId = req.params.id; // gets the contacts id from the url
363     let note = req.body.addNote; // gets the data in the text area and stores it
364     // gets the current date and then converts it into dd/mm/yyyy format
365     let date =
366         DateHandler.standardDateTo_Date_DDMMYYYY(DateHandler.getCurrentDateTime());
367
368     // do validation on the note
369     let validation = new AddNoteValidation(note);
370     let validationValue = validation.checkFieldsAreValid();
371
372     if (validationValue === true) { // if it passes the validation then add the record
373         // add note to the file as: noteId, dateTimeStamp, contactId, note
374         contactsNotes.addRecord([date, contactId, note]);
375         req.session.message = "Successfully Added A Note";
376         res.redirect("/contacts/viewContact" + contactId);
377     }
378
379     else {
380         // if the data does not pass the validation the user will get sent back to
381         // the same page with a
382         // message telling where they went wrong
383         req.session.errorMessage = validationValue;
384         res.redirect("/contacts/viewContact" + contactId);
385     }
386
387 }
388
389
390
391
392
393 // will take the user to the password page so that the user can enter a password to
394 // encrypt or decrypt the
395 // Extra Notes text area data

```

```

395
396 router.post("/viewContact:id/encryptNotes_:encOrDec", (req, res) => {
397   let id = req.params.id; // get id from the url
398   let encryptOrDecrypt = req.params.encOrDec; // stores if the user wants to
399   encrypt or decrypt the data
400   let message = req.session.message;
401
402   if (req.body.extraNotes !== undefined) {
403     req.session.extraNotesData = req.body.extraNotes; // get the extra notes
404     data from the text area and store in the session memory
405   }
406
407
408
409
410 // preform encryption on the data and append a new record to the database if there
411 // is no record containing the contact's id
412 // if a record is found with the users id then the record will be edited to contain
413 // the new encrypted data
414 router.post("/viewContact:id/encrypt", (req, res) => {
415
416   let id = req.params.id;
417   let extraNotesFile = new CSVHandler(__dirname +
418     '/dataStoreFiles/contactsExtraNotes.csv');
419
420
421   // does password validation and gets either true or a string value returned
422   let passwordValidation = new PasswordValidation(req.body.password,
423     req.body.confirmPassword);
424   let validationResult = passwordValidation.checkFieldsAreValid();
425   let encryptedData;
426   let extraNotesRecord;
427
428   // do password validation, if it passes validation then continue
429   if (validationResult === true) {
430
431     let extraNotesData = req.session.extraNotesData; // stores the
432     extraNotesData from the session memory
433     req.session.extraNotesData = undefined; // removes the property from the
434     session memory
435
436     // encrypt data with the password
437
438     encryptedData = Encryption.xorEncrypt(extraNotesData, req.body.password);
439
440     // look for a record with an id of the contacts
441     // id found ==> edit the data to store the encrypted data
442     // id not found ==> add record : [id, timestamp, contact id, data]
443
444     // search for a match for the contacts id in the contactsExtraNotes file
445
446     extraNotesRecord = extraNotesFile.getRecordsOnlyByField([2], id);
447     if(extraNotesRecord.length === 0){ // if no match is found
448       extraNotesFile.addRecord([id, encryptedData]); // add record
449       req.session.message = "Successfully Encrypted Data";
450       res.redirect("/contacts/viewContact" + id);
451     }
452
453     // match is found
454     else {
455       // edit the data in that record
456       extraNotesFile.editRecord(extraNotesRecord[0][0], [[3, encryptedData]]);
457       req.session.message = "Successfully Changed The Encrypted Data";
458       res.redirect("/contacts/viewContact" + id);
459     }
460
461   }
462
463   else {

```

```

459     // if not, redirect to the /contacts/viewContact:id/encryptNotes_encrypt
460     // with a message of what went wrong
461     req.session.message = validationResult;
462     // redirects to the post route /viewContact:id/encrypt
463     res.redirect(307, "/contacts/viewContact" + id + "/encryptNotes_encrypt");
464   }
465
466
467
468
469
470
471
472 });
473
474
475
476
477 // preform decryption on the data stored in req.session.extraNotesData
478 // if req.session.extraNotesData === undefined then redirect to the individual
479 contacts page
480 router.post("/viewContact:id/decrypt", (req, res) => {
481   // do password validation, if it passes validation then continue
482   // if not, redirect to the /contacts/viewContact:id/encryptNotes_decrypt
483   // with a message of what went wrong
484
485   let id = req.params.id;
486   let extraNotesFile = new CSVHandler(__dirname +
487     '/dataStoreFiles/contactsExtraNotes.csv');
488
489   // does password validation and gets either true or a string value returned
490   let passwordValidation = new PasswordValidation(req.body.password,
491     req.body.confirmPassword);
492   let validationResult = passwordValidation.checkFieldsAreValid();
493   let extraNotesRecord;
494   let encryptedData;
495   let decryptedData;
496
497   // do password validation, if it passes validation then continue
498   if (validationResult === true) {
499
500     // decrypt data with the password
501
502     // look for a record with an id of the contacts
503     // id found ==> edit the data to store the encrypted data
504     // id not found ==> add record : [id, timestamp, contact id, data]
505
506     // search for a match for the contacts id in the contactsExtraNotes file
507     extraNotesRecord = extraNotesFile.getRecordsOnlyByField([2], id)[0];
508
509     if(extraNotesRecord.length === 0){ // if no match is found
510       req.session.errorMessage = "No Data To Decrypt";
511       res.redirect("/contacts/viewContact" + id);
512     }
513
514     // match is found
515     else {
516       // decrypt the encrypted data and put that into session memory
517       encryptedData = extraNotesRecord[3];
518       decryptedData = Encryption.xorDecrypt(encryptedData, req.body.password);
519       req.session.message = "Successfully Decrypted The Data";
520       req.session.decryptedData = decryptedData;
521       res.redirect("/contacts/viewContact" + id);
522     }
523
524   }
525
526   else {
527     // if not, redirect to the /contacts/viewContact:id/encryptNotes_encrypt
528     // with a message of what went wrong

```

```

529     req.session.message = validationResult;
530     // redirects to the post route /viewContact:id/encrypt
531     res.redirect(307, "/contacts/viewContact" + id + "/encryptNotes_decrypt");
532   }
533
534
535
536 });
537
538
539
540
541
542
543
544
545
546 //////////////////////////////////////////////////////////////////
547 //////////////////////////////////////////////////////////////////
548 //                                     PUT ROUTES
549
550 //////////////////////////////////////////////////////////////////
551
552
553
554
555 // route for updating files containing data
556 router.put("/:id", (req, res) => {
557   let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
558   let record = req.body; // stores the data from the edit contact form in an object
559   let recordArray = Object.values(record); // turns the record object into an array
560
561   let arrayOfEdits = [];
562
563   // populate an array of changes to be made
564   for (let i = 0; i < recordArray.length; i++) {
565     // capitalizes the values stored in the array at the bellow index's
566     if (i === 0 || i === 1 || i === 2 || i === 5 || i === 7) {
567       recordArray[i] = StringUtil.capitalizeString(recordArray[i]);
568     }
569     // puts the data into the format needed for the edit record method to work
570     let recordChange = [(i+2).toString(), recordArray[i]];
571     arrayOfEdits.push(recordChange);
572   }
573
574   // do validation here
575
576   let Validation = new ContactsValidation(record.businessName, record.firstName,
577   record.lastName,
578   record.email, record.telephone, record.city, record.postcode,
579   record.address, record.status);
580
581   let validationResult = Validation.checkFieldsAreValid(); // this will either
582   // store the boolean value true or a message of what needs to be fixed
583
584   if (validationResult === true) {
585     // update all fields in the record that has been updated
586     contactsFile.editRecord(req.params.id, arrayOfEdits);
587     req.session.message = "Contact Successfully Edited";
588     res.redirect("/contacts/"); // takes user back to the contact list page
589   }
590
591   else {
592     // if something does not pass the validation then the user will get
593     // redirected to the
594     // create contacts page and a req.session.message will get set to the
595     // validationResult output
596     // this can then be accessed at the /contacts/new route to notify the user
597     // where they went wrong
598
599     req.session.message = validationResult; // validation message

```

```

594
595
596     // if validation not met then a message gets generated and the user gets
597     // sent to the edit contact page
598     // with that message stored in the session memory object
599
600     res.redirect("/contacts/edit" + req.params.id); // takes user back to the
601     // contacts list page
602 }
603
604
605
606 ///////////////////////////////////////////////////////////////////
607 ///////////////////////////////////////////////////////////////////
608 //          DELETE ROUTES
609 ///////////////////////////////////////////////////////////////////
610 ///////////////////////////////////////////////////////////////////
611
612
613 // delete route to remove a record
614 // when a contact is deleted, all of its notes and its extra notes records should
615 // also get deleted
616 router.delete("/delete:id", (req, res) => {
617     let contactsFile = new CSVHandler(__dirname + '/dataStoreFiles/contacts.csv');
618     let contactsExtraNotesFile = new CSVHandler(__dirname +
619         '/dataStoreFiles/contactsExtraNotes.csv');
620     let id = req.params.id; // gets the contacts id from the url
621     let contactsExtraNoteRecord;
622
623     let contactsNotesTempFileForDelete;
624
625     contactsFile.deleteRecord(id); // deletes the contact record from the
626     // contacts.csv file
627
628     // gets the record with the contacts id at index 2 Then returns a 2D array with
629     // the record in the only array
630     // within the 2D array. record accessed with the [0] at the end
631     contactsExtraNoteRecord = contactsExtraNotesFile.getRecordsOnlyByField([2],
632     id)[0];
633     contactsExtraNotesFile.deleteRecord(contactsExtraNoteRecord[0]); // the [0] gets
634     // the data at the 0 index which is the id
635
636     res.redirect("/contacts/");
637 }
638
639
640
641 module.exports = router;

```

dashboardRoutes.js

```

1  const express = require("express");
2  const router = express.Router();
3  const CSVHandler = require("../classes/CSVHandler");
4  const StringUtility = require('../utilityMethods/StringUtility');
5  const DateHandler = require('../classes/DateHandler');
6  const Sort = require('../classes/Sort');
7  const DashboardValidation = require('../classes/DashboardValidation');

8
9
10
11 ///////////////////////////////////////////////////////////////////
12 ///////////////////////////////////////////////////////////////////
13 // GET ROUTES
14
15 ///////////////////////////////////////////////////////////////////
16 ///////////////////////////////////////////////////////////////////
17
18
19
20
21 // renders the Dashboard page that has a form to add a scheduling and outputs the
22 // current days scheduled tasks and any future scheduled tasks
23
24 router.get("/", (req, res) => {
25   let scheduleFile = new CSVHandler(__dirname + '/dataStoreFiles/scheduled.csv');
26   let tasksWithCurrentDate = []; // will store a 2D array of all the tasks that
27   // that match the current Date
28   let tasksWithFutureDate = [];
29   let idsOfTasksWithPastDates = [];
30   let timeStampOfTask = "";
31   let date = "";
32   let time = "";
33   let message = req.session.message;
34   let errorMessage = req.session.errorMessage;

35   // get all scheduled tasks
36   let allScheduledTasks = scheduleFile.getAllRecords();
37   let currentTime = DateHandler.getCurrentDateTime(); // gets the current date
38   // and time in a timestamp
39
40
41   // determine which tasks have today's date
42   allScheduledTasks.forEach((scheduledTaskRecord) => {
43
44     timeStampOfTask = scheduledTaskRecord[2]; // will store the date time stamp
45     // in this element
46
47     if (DateHandler.isToday(timeStampOfTask)) { // checks to see if the date of
48       // the task is equal to today's date
49       tasksWithCurrentDate.push(scheduledTaskRecord);
50     }
51     else {
52       // determine which tasks have a future date
53       if (timeStampOfTask > currentTime) {
54         tasksWithFutureDate.push(scheduledTaskRecord);
55       }
56       else {
57         // determine which tasks that have a date that has passed
58         if (timeStampOfTask < currentTime) {
59           idsOfTasksWithPastDates.push(scheduledTaskRecord[0]); // appends
60           // the schedule record's id to an past dates array
61         }
62       }
63     }
64   });
65
66   // gets rid of a record that has a date passed every time the route gets called
67   // this is done to help manage the data stored in the scheduled.csv file
68   // deleting multiple records every time can cause errors if the file doesn't

```

```

have enough time to complete
// the method but this time is variable and could take too long so one is
deleted at a time instead
if (idsOfTasksWithPastDates.length > 0) {
    scheduleFile.deleteRecord(idsOfTasksWithPastDates[0]);
}

// sorts the tasks by their time stamps

tasksWithCurrentDate = Sort.bubbleSort2D(tasksWithCurrentDate, 2, true);
tasksWithFutureDate = Sort.bubbleSort2D(tasksWithFutureDate, 2, true);

res.render("Dashboard",
{
    scheduledToday: tasksWithCurrentDate,
    scheduledFuture: tasksWithFutureDate,
    flashMessage: message,
    errorMessage: errorMessage
},
(err, html) => {
    req.session.message = undefined;
    req.session.errorMessage = undefined;
    res.send(html);
})};

});

// POST ROUTES
// do validation on the fields and if they pass then add record and redirect to
dashboard
let validation = new DashboardValidation(date, time, subject, message);
let validationResult = validation.checkFieldsAreValid();

if (validationResult === true) {

    // record will contain id, date created, date and time in standard date time
    // format, date Of scheduling, time of the scheduling, subject, message
    scheduleFile.addRecord([DateHandler.createTimeStamp(date, time), date, time,
    subject, message]); // appends the record to the csv file
    req.session.message = "Successfully Added A Scheduled Task";
    res.redirect("/dashboard/");
}

```

```
133
134     // if they fail create a session message to say they have failed and redirect to
135     // dashboard
136     else {
137         req.session.errorMessage = validationResult; // validation result will have
138         // the details of what needs changing
139         res.redirect("/dashboard/");
140     }
141
142
143
144 ///////////////////////////////////////////////////////////////////
145 ///////////////////////////////////////////////////////////////////
146 //          DELETE ROUTES
147 ///////////////////////////////////////////////////////////////////
148 ///////////////////////////////////////////////////////////////////
149
150
151 //route to delete a schedule record
152 router.delete("/delete:id", (req, res) => {
153     let scheduleFile = new CSVHandler(__dirname + '/dataStoreFiles/scheduled.csv');
154     let id = req.params.id;
155     scheduleFile.deleteRecord(id);
156     req.session.message = "Successfully Deleted A Scheduled Task";
157     res.redirect("/dashboard/");
158 });
159
160
161
162 module.exports = router;
```

index.js

```
1 const express = require("express");
2 const router = express.Router();
3
4 router.get("/", (req, res) => {
5   res.render("Home");
6 });
7
8
9 router.get("/settings", (req, res) => {
10   res.render("Settings");
11 });
12
13 router.get("/termsAndConditions", (req, res) => {
14   res.render("TermsAndConditions");
15 });
16
17
18
19 module.exports = router;
```

Utility Methods Directory Contents

stringUtility.js

```

1 // utility functions so that I can avoid adding things to the prototype object of
2 // the String object
3 // as this could result in conflict with other properties
4
5 function StringUtility() {}
6
7 // static method (can be called without making an instance of the class in other
8 // objects, functions and classes)
9 // turns the first character of every word separated by a space into a capital
10 letter if it is already lower case and a-z
11
12 StringUtility.capitalizeString = (inputString) => {
13
14     let words = inputString.split(" ");
15     let currentWordsCharacters = [];
16
17     let lowerCaseArray = "abcdefghijklmnopqrstuvwxyz".split("");
18     let upperCaseArray = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".split("");
19
20     for (let i = 0; i < words.length; i++) {
21         currentWordsCharacters = words[i].split(""); // splits the current word into
22         // an array of characters
23
24         // compare character to all the characters in the lowerCaseArray, if no
25         // match is found
26         // don't do anything with it.
27
28         for (let x = 0; x < 26; x++) {
29             // if match is found, replace that character element with the
30             // character stored in the same index but in the upperCaseArray to
31             // capitalize it
32             if (lowerCaseArray[x] === currentWordsCharacters[0]) {
33                 currentWordsCharacters[0] = upperCaseArray[x];
34                 break;
35             }
36         }
37
38         currentWordsCharacters = currentWordsCharacters.join(""); // puts the
39         // characters back together to form the string
40         words[i] = currentWordsCharacters; // the capitalized word replaces the non
41         // capitalized word in that index of the words array
42     }
43
44     words = words.join(" ");
45     return words;
46
47 };
48
49 module.exports = StringUtility;
50
51

```

Views Directory Contents

AnalyticsView.ejs

Kieran Marcus Williams

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Analytics</title>
6
7      <link rel="stylesheet"
8          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
9          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
10         " crossorigin="anonymous">
11
12      <link rel="stylesheet"
13          href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
14          integrity="sha384-UHRTzL1+pBxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/
15         " crossorigin="anonymous">
16
17      <script
18          src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.3/Chart.bundle.min.js"></
19      script>
20
21      <link rel="stylesheet" type="text/css" href="/stylesheets/AnalyticsStyles.css">
22      <link rel="stylesheet" type="text/css"
23          href="/stylesheets/NavBarLoggedInStyles.css">
24      <link rel="stylesheet" type="text/css"
25          href="/stylesheets/NavBarNotLoggedInStyles.css">
26
27  </head>
28  <body>
29
30
31  <% include partials/NavBarLoggedIn.ejs %>
32  <br><br>
33
34  <h1>Analytics</h1>
35  <br><br><br>
36
37  <!--top section with the graphs-->
38
39  <div class="split-screen-horizontal">
40
41      <div class="top-half-of-analytics">
42          <div class="first-graph">
43              <canvas id="clients-gained"></canvas>
44          </div>
45          <div class="second-graph">
46              <canvas id="clients-lost"></canvas>
47          </div>
48          <!--<div class="third-graph">-->
49              <!--<canvas id="number-of-clients"></canvas>-->
50          <!--</div>-->
51      </div>
52      <br><br><br>
53
54  <!--bottom section for conversion rate and the input analytical data buttons-->
55
56  <div class="bottom-half-of-analytics">
57
58      <div class="left-side-bottom-analytics">
59
60          <div class="conversion-rate-holder">
61              <h3>Conversion Rate This Month</h3>
62              <table class="conversion-rate-table">
63                  <tr>
64                      <td><h4
65                          class="remove-margin"><%=conversionRate.ratio%></h4></td>
66                      <td><h4
67                          class="remove-margin"><%=conversionRate.percentage%></h4></td>
68                  </tr>
69              </table>
70          </div>
71
72      </div>
73
```

```
61
62 <div class="middle-bottom-analytics">
63   <label for="time-scale">Time Scale</label>
64   <form id="time-scale-form" action="/analytics/" method="get">
65     <div class="time-scale-separator">
66       <select onchange="changeAction()" class="custom-select"
67         name="time-scale" id="time-scale">
68
69         <!--ejs tags help determine which of the options should be
70           made selected -->
71
72         <% if (timeScale === "daily") { %>
73           <option selected class="time-scale-option"
74             value="daily">Daily</option>
75         <% } else { %>
76           <option selected class="time-scale-option"
77             value="daily">Daily</option>
78         <% } %>
79
80         <% if (timeScale === "weekly") { %>
81           <option selected class="time-scale-option"
82             value="weekly">Weekly</option>
83         <% } else { %>
84           <option class="time-scale-option"
85             value="weekly">Weekly</option>
86         <% } %>
87
88         <% if (timeScale === "monthly") { %>
89           <option selected class="time-scale-option"
90             value="monthly">Monthly</option>
91         <% } else { %>
92           <option class="time-scale-option"
93             value="monthly">Monthly</option>
94         <% } %>
95
96       </select>
97       <button id="time-scale-btn" class="btn
98         btn-success">Submit</button>
99     </div>
100   </form>
101 </div>
102
103 <div class="right-side-bottom-analytics">
104
105   <div class="button-holder-analytics">
106
107     <div class="row-of-buttons">
108
109       <a class="btn btn-primary" id="client-lost-btn"
110         href="/analytics/clientLost">Client Lost</a>
111
112       <a class="btn btn-primary" id="client-gained-btn"
113         href="/analytics/clientGained">Client Gained</a>
114
115     </div>
116
117     <div class="row-of-buttons">
118
119       <a class="btn btn-primary" id="lead-lost-btn"
120         href="/analytics/leadLost">Lead Lost</a>
121
122       <a class="btn btn-danger" onclick="return confirm('Are You Sure
123         You Want To Delete ALL Analytics Data?')"
124         id="remove-all-data-btn"
125         href="/analytics/removeAllData">Remove All Data</a>
126
127     </div>
128   </div>
129 </div>
```

```
119     </div>
120 
121 </div>
122 
123 
124 
125 
126 
127 
128 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
129 <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
130 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSffFWpilMquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
131 
132 
133 
134 
135 <!-- sets the variables for the graphs to use - data points and labels-->
136 <script>
137     let clientsGainedLabels = "<%=graphData.clientsGained.labels%>".split(", ");
138     let clientsGainedDataPoints =
139         "<%=graphData.clientsGained.dataPoints%>".split(", ");
140 
141     let clientsLostLabels = "<%=graphData.clientsLost.labels%>".split(", ");
142     let clientsLostDataPoints = "<%=graphData.clientsLost.dataPoints%>".split(", ");
143 
144 </script>
145 <script src="/scripts/AnalyticsScripts.js"></script>
146 
147 </body>
148 </html>
149 
```

ContactsListView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Contacts</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          " crossorigin="anonymous">
10     <link rel="stylesheet"
11         href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
12         integrity="sha384-UHRTzLI+pbxtHCWplt77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/
13         " crossorigin="anonymous">
14     <link rel="stylesheet" href="/stylesheets/ContactsListStyles.css">
15     <link rel="stylesheet" href="/stylesheets/NavBarLoggedInStyles.css">
16     <link rel="stylesheet" href="/stylesheets/NavBarNotLoggedInStyles.css">
17 </head>
18 <body>
19
20
21
22
23
24
25
26
27
28
29
30
31 <!--page for displaying the users contacts in a table-->
32
33 <h1>Contacts List
34     <% if(titleAddition !== undefined) { %>
35         -   <em style="margin-left: 15px;"><%= titleAddition %></em>
36     <% } %>
37 </h1>
38
39 <!--page is split up vertically-->
40
41 <div class="holder">
42
43     <!--left side will contain the searching and sorting options along with options
44         to add a contact, import contacts with a csv file
45         and export contacts as a csv file-->
46
47     <div id="left-side-container">
48         <div class="jumbotron">
49             <div id="modify-list-results-form">
50
51                 <form action="/contacts/search" method="post">
52                     <label for="searchBar">Search</label><br>
53                     <input name="searchBar" id="searchBar" type="text">
54                     <button class="btn btn-sm btn-outline-dark"><i class="fas fa-search"></i></button>
55
56                 </form>
57                 <br>
58
59                 <form action="/contacts/sortBy" method="post">
60
61                     <h5>Sort By</h5>
62
63                     <input type="radio" name="sortBy" id="date" value="date"> <label
64                         for="date">Date</label> <br>
65
66                     <input type="radio" name="sortBy" id="city" value="city"> <label
67                         for="city">City</label>
68
69                 </form>
70
71             </div>
72         </div>
73     </div>
74
75     <div id="right-side-container">
76
77         <table border="1">
78             <thead>
79                 <tr>
80                     <th>Name</th>
81                     <th>Phone Number</th>
82                     <th>Address</th>
83                     <th>Email</th>
84                     <th>Actions</th>
85                 </tr>
86             </thead>
87             <tbody>
88                 <tr>
89                     <td>John Doe</td>
90                     <td>(555) 123-4567</td>
91                     <td>123 Main St</td>
92                     <td>john.doe@example.com</td>
93                     <td><a href="#">Edit</a> <a href="#">Delete</a></td>
94                 </tr>
95             </tbody>
96         </table>
97
98     </div>
99
100 </div>
101 </body>
102
```

```

64         for="city">City</label> <br>
65
66         <input type="radio" name="sortBy" id="business-name"
67           value="businessName"> <label for="business-name">Business
68           Name</label> <br><br>
69
70           <label for="order">Order</label>
71           <select class="custom-select" name="order" id="order">
72             <option value="ascending">ascending</option>
73             <option value="descending">descending</option>
74           </select> <br><br>
75
76           <input class="btn btn-success" type="submit">
77         </form>
78       </div>
79
80       <hr>
81
82       <div id="contact-management">
83
84         <a class="btn btn-primary" id="add-contact" href="/contacts/new">Add
85           Contact</a>
86
87         <hr>
88
89         <!-- UNCOMMENT TO BE ABLE IMPORT CONTACTS HTML WHEN IT IS TO GET
90           IMPLEMENTED -->
91
92         <!-->
93         <!--<form>-->
94           <!--<div class="form-group">-->
95             <!--<label for="import-contacts">Import Contacts (.CSV
96               file)</label>-->
97             <!--<input type="file" class="form-control-file"
98               id="import-contacts">-->
99             <!--</div>-->
100            <!--</form>-->
101            <!--<br>-->
102
103
104           <!--<form action="#">-->
105             <!--<button class="btn btn-secondary"
106               id="import-contact-btn">import Contact</button>-->
107           <!--</form>-->
108
109           <br>
110
111           <form action="/contacts/download" method="get">
112             <button class="btn btn-secondary" id="export-contact-btn">Export
113               Contacts</button>
114           </form>
115
116           </div>
117
118           </div>
119
120           <!--on the right side there is a table containing all the contacts with a link
121             to the contact's individual page in the name field-->
122           <div id="right-side-container">
123             <table class="table">
124               <thead class="thead-dark">
125                 <tr>
126                   <th>Name</th>
127                   <th>Business</th>
128                   <th>City</th>
129                   <th>Email</th>
130                   <th>Telephone</th>
131                 </tr>

```

```
126      </thead>
127
128      <tbody>
129
130          <!--gets the contacts details and puts it into the table row by row-->
131          <% contactRecords.forEach((record) => { %>
132              <tr>
133                  <td><a
134                      href="/contacts/viewContact<%=record[0]%>"><%=record[3] + %
135                          " + record[4]></a></td>
136                  <td><%=record[2]> </td>
137                  <td><%=record[7]></td>
138                  <td><%=record[5]></td>
139                  <td><%=record[6]></td>
140              </tr>
141          <%});%>
142
143      </tbody>
144  </table>
145  </div>
146</div>
147
148 <script src="/scripts/ContactListScripts.js"></script>
149 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
150 <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
151 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
152 </body>
153 </html>
```

CreateContactView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Create Contact</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          crossorigin="anonymous">
10     <link rel="stylesheet" href="/stylesheets/CreateContactStyles.css">
11     <link rel="stylesheet" href="/stylesheets/NavbarLoggedInStyles.css">
12     <link rel="stylesheet" href="/stylesheets/NavbarNotLoggedInStyles.css">
13     <script src="/scripts/CreateContactScripts.js"></script>
14 </head>
15 <body>
16
17
18 <%if (flashMessage !== undefined){%>
19 <div class="alert alert-danger">
20     <%= flashMessage %>
21 </div>
22 <%}%>
23
24 <!--basic form using bootstrap that will be used as a data capture to make create a
contact-->
25
26 <h1>Create Contact</h1>
27 <div class="container">
28     <div class="jumbotron">
29         <form class="form" action="/contacts" method="post">
30             <h3>Contact Details</h3><br>
31             <div class="row">
32                 <div class="col">
33                     <label for="business">Business</label>
34                     <input class="form-control" id="business" name="businessName"
35                         value="<%=savedBodyValue[0]%>" type="text"><br>
36                 </div>
37                 <div class="col">
38                     <label for="first-name">First Name</label>
39                     <input class="form-control" id="first-name" name="firstName"
40                         value="<%=savedBodyValue[1]%>" type="text"><br>
41                 </div>
42             <div class="row">
43                 <div class="col">
44                     <label for="last-name">Last Name</label>
45                     <input class="form-control" id="last-name" name="lastName"
46                         value="<%=savedBodyValue[2]%>" type="text"><br>
47                 </div>
48                 <div class="col">
49                     <label for="email">Email</label>
50                     <input class="form-control" id="email" name="email"
51                         value="<%=savedBodyValue[3]%>" type="text"><br>
52                 </div>
53             <div class="row">
54                 <div class="col">
55                     <label for="telephone">Telephone</label>
56                     <input class="form-control" id="telephone" name="telephone"
57                         value="<%=savedBodyValue[4]%>" type="text"><br>
58                 </div>
59                 <div class="col">
60                     <label for="city">City</label>
61                     <input class="form-control" id="city" name="city"
62                         value="<%=savedBodyValue[5]%>" type="text"><br>
63                 </div>
64             </div>
65         <div class="row">
66             <div class="col">

```

```

63         <label for="postcode">Postcode</label>
64         <input class="form-control" id="postcode" name="postcode"
65             value="<%>=savedBodyValue[6]<%>" type="text"><br>
66     </div>
67     <div class="col">
68         <label for="address">Address</label>
69         <input class="form-control" id="address" name="address"
70             value="<%>=savedBodyValue[7]<%>" type="text"><br>
71     </div>
72 </div>
73 <div class="row">
74     <div class="col">
75         <label for="status">Contact Status</label>
76         <select class="custom-select" name="status" id="status">
77             <option value="lead">Lead</option>
78             <option value="recontact-lead">Recontact Lead</option>
79             <option value="dead-lead">Dead Lead</option>
80             <option value="current-client">Current Client</option>
81             <option value="past-client">Past Client</option>
82         </select>
83     </div>
84     <input class="btn btn-primary btn-lg btn-submit" type="submit">
85 </form>
86 </div>
87
88
89
90
91
92 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
93     integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
94     crossorigin="anonymous"></script>
95 <script
96     src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
97     integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
98     crossorigin="anonymous"></script>
99 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
100    integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQxSffFWpi1MquVdAyjUar5+76PVCmYl"
101    crossorigin="anonymous"></script>
</body>
</html>

```

DashboardView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Dashboard</title>
6
7      <link rel="stylesheet"
8          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
9          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
10         crossorigin="anonymous">
11     <link rel="stylesheet"
12         href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
13         integrity="sha384-UHRTZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/"
14         crossorigin="anonymous">
15     <link rel="stylesheet" href="/stylesheets/NavbarLoggedInStyles.css">
16     <link rel="stylesheet" href="/stylesheets/NavbarNotLoggedInStyles.css">
17     <link rel="stylesheet" href="/stylesheets/DashboardStyles.css">
18
19
20     <% include partials/NavbarLoggedIn.ejs %>
21
22     <%if (flashMessage !== undefined){%>
23         <div class="alert alert-success">
24             <%= flashMessage %>
25         </div>
26     <%}%>
27
28     <%if (errorMessage !== undefined){%>
29         <div class="alert alert-danger">
30             <%= errorMessage %>
31         </div>
32     <%}%>
33
34     <h1>Dashboard</h1>
35
36     <!-- The Dashboard is split into 3 sections -->
37     <div id="dashboard-separator">
38
39         <!--on the left there is the form to create a scheduling -->
40
41         <div id="schedule-task-form">
42
43             <h3>Schedule Task</h3>
44
45
46             <form class="dashboard-slots" title="Schedule Task" action="/dashboard/"
47                 method="post" id="Schedule Task">
48                 <label for="date">date:</label> <br>
49                 <input id="date" name="date" title="date" type="text"
50                     placeholder="dd/mm/yyyy"> <br><br>
51
52                 <label for="time">time:</label><br>
53                 <input id="time" name="time" title="time" type="text" placeholder="24
54                     hour time: hh:mm"> <br><br>
55
56                 <label for="subject">subject:</label><br>
57                 <input id="subject" name="subject" type="text"> <br><br>
58
59                 <label for="message">message:</label> <br>
60                 <textarea onkeypress="return returnKeyBlocker(event)" id="message"
61                     title="message" name="message" cols="30" rows="10"></textarea> <br><br>
62
63                 <input class="btn btn-success" type="submit">
64             </form>
65         </div>

```

```

63
64    <!-- in the middle there will be an array that shows anything that is
65    scheduled.csv for the current date -->
66    <div id="today">
67        <h3>Today</h3>
68        <div class="dashboard-slots">
69
70
71
72
73        <%scheduledToday.forEach((schedule) => { %>
74
75            <tr>
76                <td>
77
78                    <div class="schedule-info">
79                        <div class="time-display-today"><%=schedule[3]%></div>
80                        <div class="date-display-today"><%=schedule[4]%></div>
81                        <span style="font-weight: bold"><div
82                            class="subject-display-today"><%=schedule[5]%></div></span
83                            >
84                            <div class="message-display-today"><%=schedule[6]%></div>
85                        </div>
86
87                </td>
88
89                <td>
90                    <form
91                        action="/dashboard/delete<%=schedule[0]%>?_method=DELETE"
92                        method="post">
93                        <button style="float: right; margin-bottom: 10px;
94                            font-size: x-large;" class="btn"><i class="trash-button
95                            fas fa-trash-alt"></i></button>
96                    </form>
97
98                </td>
99
100            </tr>
101            <br><br><hr>
102
103        <%}); %>
104
105        </div>
106    </div>
107
108
109
110
111
112
113    <!--on the right all the scheduled.csv tasks further on into the future from
114    just the the current day will be put into this list -->
115
116    <div id="future">
117        <h3>Future</h3>
118        <div class="dashboard-slots">
119
120            <%scheduledFuture.forEach((schedule) => { %>
121
122                <tr>
123                    <td>
124
125                        <div class="schedule-info">
126                            <div class="time-display-today"><%=schedule[3]%></div>
127                            <div class="date-display-today"><%=schedule[4]%></div>
128                            <span style="font-weight: bold"><div
129                                class="subject-display-today"><%=schedule[5]%></div></span>
130                            <div class="message-display-today"><%=schedule[6]%></div>
131                        </div>
132
133                    </td>
134
135                    <td>
136                        <form action="/dashboard/delete<%=schedule[0]%>?_method=DELETE"
137                        method="post">
138                            <button style="float: right; margin-bottom: 10px; font-size:

```

```
        x-large;" class="btn">><i class="trash-button fas fa-trash-alt"></i></button>
125      </form>
126    </td>
127
128  </tr>
129  <br><br><hr>
130
131
132
133  <%});%>
134  </div>
135  </div>
136</div>
137
138
139
140<script src="/scripts/DashboardScripts.js"></script>
141<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkYIJK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
142<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
143<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
144</body>
145</html>
```

EditContactView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Edit Contact</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          crossorigin="anonymous">
10     <link rel="stylesheet" href="/stylesheets/EditContactStyles.css">
11     <link rel="stylesheet" href="/stylesheets/NavbarLoggedInStyles.css">
12     <link rel="stylesheet" href="/stylesheets/NavbarNotLoggedInStyles.css">
13     <script src="/scripts/EditContactScripts.js"></script>
14 </head>
15 <body>
16
17 <% include partials/NavbarLoggedIn.ejs %>
18
19 <%if (flashMessage !== undefined){%>
20     <div class="alert alert-danger">
21         <%= flashMessage %>
22     </div>
23 <%}%>
24
25 <!--form containing a contacts data in the text fields-->
26 <h1>Edit Contact</h1>
27 <div class="container">
28
29     <div class="jumbotron">
30         <!--action url is set to put to have the correct http verb for updating db-->
31         <form class="form" action="/contacts/<%= contact.record[0] %>?_method=PUT"
32             method="post">
33             <h3>Contact Details</h3><br>
34             <div class="row">
35                 <div class="col">
36                     <label for="business">Business</label>
37                     <input class="form-control" id="business" type="text"
38                         name="businessName" value="<%= contact.record[2] %>"><br>
39                 </div>
40                 <div class="col">
41                     <label for="first-name">First Name</label>
42                     <input class="form-control" id="first-name" type="text"
43                         name="firstName" value="<%= contact.record[3] %>"><br>
44                 </div>
45             </div>
46             <div class="row">
47                 <div class="col">
48                     <label for="last-name">Last Name</label>
49                     <input class="form-control" id="last-name" type="text"
50                         name="lastName" value="<%= contact.record[4] %>"><br>
51                 </div>
52                 <div class="col">
53                     <label for="email">Email</label>
54                     <input class="form-control" id="email" type="text" name="email"
55                         value="<%= contact.record[5] %>"><br>
56                 </div>
57             </div>
58             <div class="row">
59                 <div class="col">
60                     <label for="telephone">Telephone</label>
61                     <input class="form-control" id="telephone" type="text"
62                         name="telephone" value="<%= contact.record[6] %>"><br>
63                 </div>
64                 <div class="col">
65                     <label for="city">City</label>
66                     <input class="form-control" id="city" type="text" name="city"
67                         value="<%= contact.record[7] %>"><br>
68                 </div>
69             </div>
70             <div class="row">
71                 <div class="col">

```

```

63         <label for="postcode">Postcode</label>
64         <input class="form-control" id="postcode" type="text"
65           name="postcode" value="<% contact.record[8] %>"><br>
66     </div>
67     <div class="col">
68       <label for="address">Address</label>
69       <input class="form-control" id="address" type="text"
70         name="address" value="<% contact.record[9] %>"><br>
71     </div>
72   </div>
73   <div class="row">
74     <div class="col">
75       <label for="status">Contact Status</label>
76       <select class="custom-select" name="status" id="status">
77
78         <!--the bellow conditionals check to see the value of the
79         clients current status and makes the
80         the right one the selected value out of the drop down
81         menu-->
82
83         <%if(contact.record[10] === "lead") { %>
84           <option selected value="lead">Lead</option>
85         <%} else { %>
86           <option value="lead">Lead</option>
87         <%}%>
88
89         <%if(contact.record[10] === "recontact-lead") { %>
90           <option selected value="recontact-lead">Recontact
91             Lead</option>
92         <%} else { %>
93           <option value="recontact-lead">Recontact Lead</option>
94         <%}%>
95
96         <%if(contact.record[10] === "dead-lead") { %>
97           <option selected value="dead-lead">Dead Lead</option>
98         <%} else { %>
99           <option value="dead-lead">Dead Lead</option>
100        <%}%>
101
102
103         <%if(contact.record[10] === "current-client") { %>
104           <option selected value="current-client">Current
105             Client</option>
106         <%} else { %>
107           <option value="current-client">Current Client</option>
108         <%}%>
109
110         </select>
111     </div>
112   </div>
113   <input class="btn btn-primary btn-lg btn-submit" type="submit">
114 </form>
115 </div>
116 </div>
117
118
119
120
121
122 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
123   integrity="sha384-KJ3o2DKtIkVYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
124   crossorigin="anonymous"></script>
125 <script
126   src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
127   integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
128   crossorigin="anonymous"></script>

```

```
124 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
125   integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
126   crossorigin="anonymous"></script>
</body>
</html>
```

EncryptNotesView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Encrypt Extra Notes</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          crossorigin="anonymous">
10     <link rel="stylesheet" href="/stylesheets/ChangePasswordStyles.css">
11     <link rel="stylesheet" href="/stylesheets/NavBarLoggedInStyles.css">
12     <link rel="stylesheet" href="/stylesheets/NavBarNotLoggedInStyles.css">
13     <script src="/scripts/EncryptionNotesPasswordScripts.js"></script>
14 </head>
15 <body>
16
17
18 <%if (flashMessage !== undefined){%>
19 <div class="alert alert-danger">
20     <%= flashMessage %>
21 </div>
22 <%}%>
23
24
25 <h1>Encryption Password</h1>
26 <div class="container">
27
28
29
30     <!--basic form to change the password-->
31     <div class="jumbotron">
32         <form class="form"
33             action="/contacts/viewContact<%=contactId%>/<%=encryptOrDecrypt%>"
34             method="post">
35             <div class="row">
36
37                 <div class="col">
38                     <label for="password">Password</label>
39                     <input class="form-control" name="password" id="password"
40                         type="password"><br>
41
42                 <div class="col">
43                     <label for="confirmPassword">Confirm Password</label>
44                     <input class="form-control" name="confirmPassword"
45                         id="confirmPassword" type="password"><br>
46
47             </div>
48
49             <div class="row">
50                 <div class="col">
51                     <div class="form-check">
52                         <input class="form-check-input" id="see-password"
53                             type="checkbox" onclick="showPassword();">
54                         <label class="form-check-label" for="see-password">See
55                             Password</label>
56
57                     </div>
58
59                 </div>
60
61             <small id="password-help" class="form-text text-muted">
62                 Your password must be longer than 8 characters, contain upper and
63                 lower case letters, numbers,
64                 and must not contain spaces, special characters, or emoji.
65             </small>
66             <input class="btn btn-primary btn-lg btn-submit" type="submit">
67
68         </form>
69
70     </div>
71 </div>

```

```
63
64
65
66
67
68 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
       integrity="sha384-KJ3o2DKtIkYIK3UENzm7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
       crossorigin="anonymous"></script>
69 <script
       src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
       integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPsvXusvfa0b4Q"
       crossorigin="anonymous"></script>
70 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
       integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
       crossorigin="anonymous"></script>
71 </body>
72 </html>
```

HomeView.ejs

Kieran Marcus Williams

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Home</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          crossorigin="anonymous">
10     <link rel="stylesheet"
11         href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
12         integrity="sha384-UHRTzL1+pbtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/
13         crossorigin="anonymous">
14     <link rel="stylesheet" href="/stylesheets/HomeStyles.css">
15     <link rel="stylesheet" href="/stylesheets/NavbarLoggedInStyles.css">
16     <link rel="stylesheet" href="/stylesheets/NavbarNotLoggedInStyles.css">
17     <script src="/scripts/HomeScripts.js"></script>
18 </head>
19 <body>
20
21     <% include partials/NavbarLoggedIn.ejs %>
22
23     <!--contains a description of the site and links to register an account or log in-->
24
25     <div class="container">
26
26         <h1>Welcome To KW CRM</h1>
27
28         <br><br>
29
30         <div class="jumbotron">
31             <h3>About This Customer Relationship Management System</h3><br><br>
32             <p>Many large organisations will have their own bespoke software that they
33                 would use as a customer relationship management system (CRM). However, this
34                 site is
35                     a generic CRM that is web based and will allow for many businesses that
36                     are unable to afford the bespoke software to have access to the services
37                     provided by them.
38
39                     The problem with many of the other online CRM's is that they offer too
40                     many services and overcomplicate things for many end users. The aim of
41                     this system is to incorporate
42
43                     only what small business owners and freelancers need and not overwhelm
44                     users with an abundance of features.
45
46                     Other online CRMS also often cost a monthly subscription to use.</p><br>
47
48             <hr>
49             <br>
50             <h4><em>KW CRM is your best option for a free simple Customer Relationship
51                 Management System</em></h4><br><br>
52         </div>
53     </div>
54
55
56
57     <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
58         integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
59         crossorigin="anonymous"></script>
60     <script
61         src="https://cdn.jsdelivr.net/npm/@popperjs/core@1.12.9/dist/umd/popper.min.js"
62         integrity="sha384-ApNbgh9B+y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
63         crossorigin="anonymous"></script>
64     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
65         integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRQQxSffFWpi1MquVdAyjUar5+76PVCmYl"
66         crossorigin="anonymous"></script>
67
68 </body>
69 </html>

```

IndividualContactView.ejs

Kieran Marcus Williams

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Individual Contact</title>
6
7      <link rel="stylesheet"
8          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
9          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
10         crossorigin="anonymous">
11
12     <link rel="stylesheet"
13         href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
14         integrity="sha384-UHRTZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/"
15         crossorigin="anonymous">
16
17     <link rel="stylesheet" href="/stylesheets/ViewIndividualContactStyles.css">
18     <link rel="stylesheet" href="/stylesheets/NavBarLoggedInStyles.css">
19     <link rel="stylesheet" href="/stylesheets/NavBarNotLoggedInStyles.css">
20
21 </head>
22 <body>
23
24 <% include partials/NavBarLoggedIn.ejs %>
25
26
27 <%--if any error or success messages need to be outputted to the user then they will
28 be entered in here--%>
29
30 <%if (flashMessage !== undefined){%>
31     <div class="alert alert-success">
32         <%= flashMessage %>
33     </div>
34 <%}%>
35
36 <%if (errorMessage !== undefined){%>
37     <div class="alert alert-danger">
38         <%= errorMessage %>
39     </div>
40 <%}%>
41
42
43 <%--this page will show all of the details of an individual contact--%>
44 <%--the page is vertically split in half--%>
45 <div id="halfway-separator">
46
47     <%--contacts main details are here--%>
48     <div id="left-side">
49         <h2><%= contactDetails[3] + " " + contactDetails[4]></h2><br>
50         <div class="contact-details">
51             <div class="jumbotron">
52                 <table>
53                     <tr>
54                         <td>
55                             <h5><span>Business: </span></h5>
56                         </td>
57                         <td>
58                             <h5><%= contactDetails[2]></h5>
59                         </td>
60                         <td>
61                             <h5><span>Email: </span></h5>
62                         </td>
63                         <td>
64                             <h5><%= contactDetails[5]></h5>
65                         </td>
66                     </tr><tr>
67                         <td>
68                             <h5><span>Phone Number: </span></h5>
69                         </td>
70                         <td>
71                             <h5><%= contactDetails[6]></h5>
72                         </td>
73                     </tr><tr>
74                         <td>
75                             <h5><span>Address: </span></h5>
76                         </td>
77                         <td>
78                             <h5><%= contactDetails[7]></h5>
79                         </td>
80                     </tr>
81
82     </div>
83
84 </div>
85 </body>
86
```

```

66
67             <td>
68                 <h5><span>City: </span></h5>
69             </td>
70                 <h5><%= contactDetails[7] %></h5>
71             </td>
72         </tr><tr>
73             <td>
74                 <h5><span>Postcode: </span></h5>
75             </td>
76                 <h5><%= contactDetails[8] %></h5>
77             </td>
78         </tr><tr>
79             <td>
80                 <h5><span>Address: </span></h5>
81             </td>
82                 <h5><%= contactDetails[9] %></h5>
83             </td>
84         </tr>
85
86
87
88     <!-- below code changes the data from the csv file for client
89     status into something that looks better
90     eg. current-client ===> Current
91     Client
92     -->
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123             </div>
124         </div>
125
126     <!--this text area form allows for the user to add a note which will get
127     displayed in the notes table to the left-->
128     <div class="add-note-section">
129
130         <form action="/contacts/addnote<%=contactDetails[0] %>" method="post">
131
132             <h3><label for="addNote">Add Note</label><br></h3>
133             <textarea name="addNote" onkeypress="return returnKeyBlocker(event)" id="addNote" cols="60" rows="7"></textarea>

```

```

133
134         <br>
135         <input class="btn-submit-add-note btn btn-primary" type="submit">
136     </form>
137
138     </div>
139
140     <!--edit contact button will redirect the user to the edit contact page with
141     that contacts details filled in-->
142     <!--the delete contact will ask for verification before deleting the
143     contact-->
144
145     <div class="contact-management-buttons">
146
147         <a class="btn btn-secondary btn-edit-contact"
148             href="/contacts/edit<%=contactDetails[0]%>">Edit Contact</a>
149
150         <form action="/contacts/delete<%=contactDetails[0]%>?_method=DELETE"
151             onsubmit="return confirm('Are You Sure You Want To Delete This
152             Contact');" method="post">
153             <button class="btn btn-danger btn-delete-contact">Delete
154             Contact</button>
155         </form>
156
157     </div>
158
159
160     <!--on the right side there is another text area that can be used to add, edit
161     data and view data-->
162
163     <div id="right-side">
164         <div class="extra-notes">
165             <br><br><br>
166
167             <!--the place holder will display instructions as to how to save,
168             encrypt and decrypt the data as
169             it is not super intuitive at first -->
170             <form
171                 action="/contacts/viewContact<%=contactDetails[0]%>/encryptNotes_encrypt"
172                 method="post">
173                 <h3><label for="extraNotes">Extra Notes</label></h3>
174                 <textarea name="extraNotes" id="extraNotes" onkeypress="return
175                 returnKeyBlocker(event)" cols="80" rows="10"
176                 placeholder="">
177
178                 REMEMBER THE PASSWORD! There Is No Way Of Getting The Data Back If
179                 You Forget It. Type Data into text area and press save & encrypt to save the data to
180                 the file with xor encryption done to
181                 to it. this data can be changed by first decrypting it, changing the data and then
182                 pressing the save & encrypt button again.
183                 When data is decrypted it will not save anything back to the file. If the wrong
184                 decrypt password is entered, the decrypted text will
185                 not be the original text
186             ><%=extraNotes%></textarea><br>
187
188             <!--when the save and encrypt btn is pressed, the data in the text
189             field will get encrypted and saved to the database-->
190             <!--when the decrypt button is pressed the data will get retrieved
191             from the database and outputted to the text area-->
192             <!--a password page will appear before the encryption or decryption
193             happens though-->
194             <div class="btn-container-extra-notes">
195
196                 <div id="extra-notes-btn-group">
197
198                     <input class="btn-submit-extra-notes btn btn-secondary"
199                         type="submit" value="Decrypt"
200                         formaction="/contacts/viewContact<%=contactDetails[0]%>
201                         /encryptNotes_decrypt" formmethod="post">
202
203                     <input class="btn-submit-extra-notes btn btn-primary"
204                         type="submit" value="Save & Encrypt">
205
206             </div>
207         </div>
208
209     </div>
210
211 
```

```

185                         </div>
186
187                         </div>
188                     </form>
189                 </div>
190
191             <br><br><br><br><br>
192
193             <!--this is a table that will display all the notes and the date that each
194             note was created next to it -->
195             <div id="table-holder-notes">
196                 <table class="table ">
197                     <thead class="thead-grey">
198                         <tr>
199                             <th>Date</th>
200                             <th>Notes</th>
201                         </tr>
202                     </thead>
203
204                     <tbody>
205
206                         <%contactNotes.forEach( (noteRecord) => {%
207
207                             <tr>
208                                 <!-- the notes creation date gets put into here-->
209                                 <td class="td-borededer">
210                                     <%=noteRecord[2]>
211                                 </td>
212                                 <!--the note data will be in here-->
213                                 <td class="td-borededer">
214                                     <%=noteRecord[4]>
215                                 </td>
216
217                             </tr>
218
219                         <%});%>
220
221                     <tbody>
222                 </table>
223             </div>
224
225
226         </div>
227
228     </div>
229
230
231     <script src="/scripts/ViewIndividualContactScripts.js"></script>
232     <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
233         integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
234         crossorigin="anonymous"></script>
235     <script
236         src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
237         integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4O"
238         crossorigin="anonymous"></script>
239     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
240         integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSffFWpi1MquVdAyjUar5+76PVCmYl"
241         crossorigin="anonymous"></script>
242
243     </body>
244
245 </html>

```

SettingsView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Terms And Conditions</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          crossorigin="anonymous">
10     <link rel="stylesheet"
11         href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
12         integrity="sha384-UHRTZLI+pbxtHCWplt77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/
13         crossorigin="anonymous">
14     <link rel="stylesheet" href="/stylesheets/SettingsStyles.css">
15     <link rel="stylesheet" href="/stylesheets/NavbarLoggedInStyles.css">
16     <link rel="stylesheet" href="/stylesheets/NavbarNotLoggedInStyles.css">
17     <script src="/scripts/SettingsScripts.js"></script>
18 </head>
19 <body>
20
21 <% include partials/NavbarLoggedIn.ejs %>
22
23 <!--This page will display links to the edit account page and to the Terms and
24 Conditions Page-->
25 <h1>Settings</h1>
26 <br><br><br>
27 <div class="jumbotron">
28
29     <!-- UNCOMMENT WHEN IMPLEMENTATION OF USERS IS ADDED IN -->
30     <!--<h4><a href="#">Edit Account Details</a></h4>-->
31
32     <br>
33
34     <h4><a href="/termsAndConditions">Terms And Conditions</a></h4>
35 </div>
36
37 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
38     integrity="sha384-KJ3o2DKtIkY1K3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
39     crossorigin="anonymous"></script>
40 <script
41     src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
42     integrity="sha384-ApNbgh9B+y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
43     crossorigin="anonymous"></script>
44 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
45     integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRQQxSffFWpi1MquVdAyjUar5+76PVCmYl"
46     crossorigin="anonymous"></script>
47 </body>
48 </html>

```

TermsAndConditionsView.ejs

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Terms And Conditions</title>
6      <link rel="stylesheet"
7          href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
8          integrity="sha384-Gn5384xqQlaoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
9          " crossorigin="anonymous">
10     <link rel="stylesheet"
11         href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
12         integrity="sha384-UHRTZLI+pbxtHCWplt77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ81WUE00s/
13         " crossorigin="anonymous">
14     <link rel="stylesheet" href="/stylesheets/TermsAndConditionsStyles.css">
15     <link rel="stylesheet" href="/stylesheets/NavBarLoggedInStyles.css">
16     <link rel="stylesheet" href="/stylesheets/NavBarNotLoggedInStyles.css">
17 </head>
18 <body>
19
20 <% include partials/NavBarLoggedIn.ejs %>
21
22 <br><br>
23 <h1>Terms And Conditions</h1>
24 <br><br>
25 <div class="jumbotron">
26     <h5>By Using This Website You Are Agreeing To These Terms And
27     Conditions.</h5>
28     <br>
29     <p>
30         Any data held on your contacts is your responsibility. <br>
31         You agree to be liable for any GDPR law breeches that are related to the
32         contacts you have created. <br><br>
33
34         If you wish to receive data that we hold regarding your usage of the system
35         <br>
36         you must make a formal request in the form of an email to
37         kieranmarcus@KWCrm.com
38     </p>
39 </div>
40
41 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
42 integrity="sha384-KJ3o2DKtIkYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
43 crossorigin="anonymous"></script>
44 <script
45     src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
46     integrity="sha384-ApNbgh9B+y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
47     crossorigin="anonymous"></script>
48 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
49     integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSffFWpi1MquVdAyjUar5+76PVCmYl"
50     crossorigin="anonymous"></script>
51 </body>
52 </html>
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
617
618
619
619
620
621
622
623
624
625
626
627
627
628
629
629
630
631
632
633
634
635
635
636
637
637
638
639
639
640
641
642
643
643
644
645
645
646
647
647
648
649
649
650
651
652
653
653
654
655
655
656
657
657
658
659
659
660
661
661
662
663
663
664
665
665
666
667
667
668
669
669
670
671
671
672
673
673
674
675
675
676
677
677
678
679
679
680
681
681
682
683
683
684
685
685
686
687
687
688
689
689
690
691
691
692
693
693
694
695
695
696
697
697
698
699
699
700
701
701
702
703
703
704
705
705
706
707
707
708
709
709
710
711
711
712
713
713
714
715
715
716
717
717
718
719
719
720
721
721
722
723
723
724
725
725
726
727
727
728
729
729
730
731
731
732
733
733
734
735
735
736
737
737
738
739
739
740
741
741
742
743
743
744
745
745
746
747
747
748
749
749
750
751
751
752
753
753
754
755
755
756
757
757
758
759
759
760
761
761
762
763
763
764
765
765
766
767
767
768
769
769
770
771
771
772
773
773
774
775
775
776
777
777
778
779
779
780
781
781
782
783
783
784
785
785
786
787
787
788
789
789
790
791
791
792
793
793
794
795
795
796
797
797
798
799
799
800
801
801
802
803
803
804
805
805
806
807
807
808
809
809
810
811
811
812
813
813
814
815
815
816
817
817
818
819
819
820
821
821
822
823
823
824
825
825
826
827
827
828
829
829
830
831
831
832
833
833
834
835
835
836
837
837
838
839
839
840
841
841
842
843
843
844
845
845
846
847
847
848
849
849
850
851
851
852
853
853
854
855
855
856
857
857
858
859
859
860
861
861
862
863
863
864
865
865
866
867
867
868
869
869
870
871
871
872
873
873
874
875
875
876
877
877
878
879
879
880
881
881
882
883
883
884
885
885
886
887
887
888
889
889
890
891
891
892
893
893
894
895
895
896
897
897
898
899
899
900
901
901
902
903
903
904
905
905
906
907
907
908
909
909
910
911
911
912
913
913
914
915
915
916
917
917
918
919
919
920
921
921
922
923
923
924
925
925
926
927
927
928
929
929
930
931
931
932
933
933
934
935
935
936
937
937
938
939
939
940
941
941
942
943
943
944
945
945
946
947
947
948
949
949
950
951
951
952
953
953
954
955
955
956
957
957
958
959
959
960
961
961
962
963
963
964
965
965
966
967
967
968
969
969
970
971
971
972
973
973
974
975
975
976
977
977
978
979
979
980
981
981
982
983
983
984
985
985
986
987
987
988
989
989
990
991
991
992
993
993
994
995
995
996
997
997
998
999
999

```

