Specification Document

Project Name: Kerridge Commercial Systems Application

Team Number 8

Document Information

Project Name: Kerridge Commercial Systems Application

Prepared By: All of Team 8 **Preparation Date: 11/12/2018**

Document Version No: 1.9 **Document Version Date: 08/03/2019**

Version History

Ver. No.	Ver. Date	Revised By	Description
1.0	11/12/2018	Team 8	Formerly the draft specification document. Worked on all sections of the document following feedback on submission.
1.1	18/02/2019	Conor Lambert and Nathan Thompson	> Team Roles > Task Breakdown > Project Requirements
1.2	23/02/2019	Conor Lambert	> Definition of Terms > Constraints > Deliverables > Team Roles
1.3	28/02/2019	Conor Lambert and Nathan Thompson	> References > Deliverables > Design Introduction > Assumptions > Project Plan
1.4	03/03/2019	Conor Lambert	> Document Section Breakdown
1.5	04/03/2019	Conor Lambert, Nathan Thompson and Ben Harris	> Task Breakdown > Design Section > Test Plan
1.6	05/03/2019	Conor Lambert, Nathan Thompson and Ben Harris	> Rewrote the document to be far more professional and meet the template layout for easier reading.
1.7	06/03/2019	Conor Lambert, Ben Harris, Jason Nevill and Nick Asher	> Roles and Deliverables > Background and Analysis
1.8	07/03/2019	Team 8	> Other Considerations > Roles and Deliverables > Background and Analysis
1.9	08/03/2019	Conor Lambert, Matthew Trimble, Nick Asher, Ross Dodds and Ben Harris	> Cover Page > Executive Summary > Software Design > Roles and Deliverables > Contribution Matrix

Table of Contents

Purpose (Executive summary)

This document explains the layout and execution of the project that is to be used by the Team. It covers all details that are necessary to complete the application. The document is made up of various sections of these significant details.

Below is a breakdown of the sections that are included in the document.

Section	Sub-sections	Descriptions
Background & Analysis	Analysis Process Background and Analysis Project Purpose	This section gives a brief overview of what the project entails and all the research that has been performed by the team to discover the best and worst aspects of what a sales platform should contain.
Roles and Deliverables	Team Roles and Responsibilities Deliverables	Team Roles breaks down the role of each person in the team followed by a description of their responsibilities throughout this document. The Deliverables section describes both the methodology that has been employed and the deadlines that are expected for the project.
Project Plan	(N/A)	The Project Plan shows how the team is going to go about completing the project. By following a specific plan, the team knows what to do and where and this avoids a few problems. The things that the team wants to implement have already been laid out and therefore can avoid feature creep.
Hardware and Software Resources	(N/A)	Hardware and Software Resources explains the resources that are to be used by the team during the project; including both communication and programming applications.
References	(N/A)	The References section provides the relevant references (cited using the Harvard referencing style) for any sources of information that have been gathered from and used.
Definition of Terms	(N/A)	The Definition of Terms is a table of abbreviations or acronyms that are used throughout the project and thusly implemented into this document to provide knowledge of what they are.
Solution Requirements	Functional Requirements Non-Functional Requirements	These sections explain the various requirements for the project. Each table contains different information such as the requirement, its priority and the supplier compliance.
Other Considerations	Assumptions Constraints and Dependencies	The Assumptions section explains what is assumed about the project including but not limited to the type of data in a database or the user assumptions such as device usage. Constraints consider the problems that may arise throughout the project, and therefore the team can come up with both workarounds and solutions to each problem. The Dependencies section describes what the user (or the tester) of the application would need for it to work as intended.

Software Design	> Overview of the System > Modifications > System Architecture > High-Level Overview of Functionality and Responsibilities of the System > Package and Class Diagrams > Dynamic Behaviour of the System > GUI and Human Interface Views > Website Design	This section describes the entire process that is involved with when making the application. This includes the diagrams used to annotate different aspects of the system, the Website design for the website being create alongside the application and various things that application will and won't do.
Test Plan	(N/A)	The Test Plan describes how the testing is to be carried out both during and after the completion of the application

1 Background & Analysis

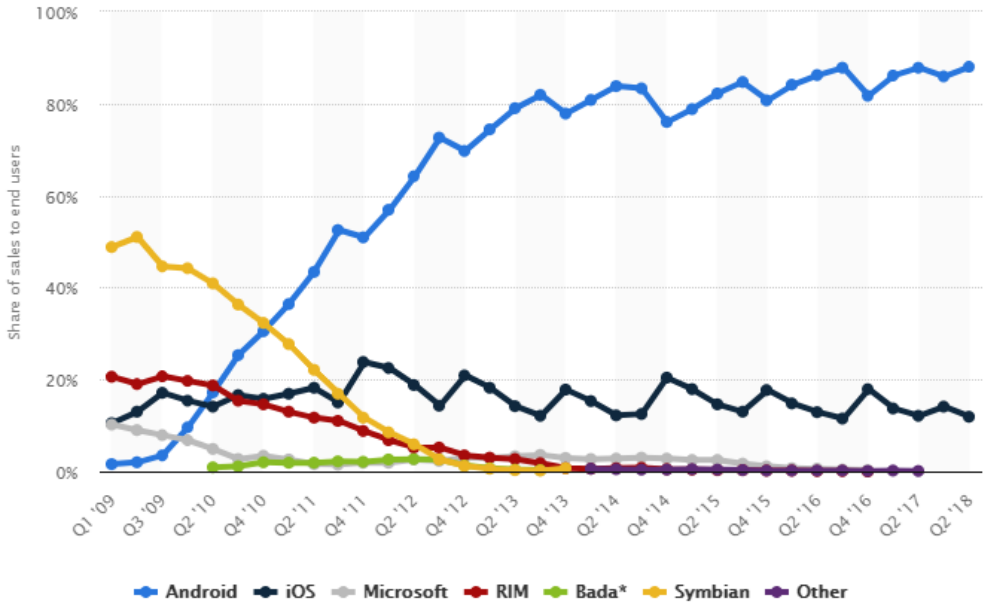
1.1 Analysis Process

The basis of our analysis begins with the amount of research that has been gathered. By finding various apps that represent similarities to what the team wants out of the KCS application, the team will gain a better understanding of what goals are needed to be achieved.

Most of the research has been done by searching for high-end applications that are like what KCS has specified that they would like as features. All this research was done through Google as it would be difficult otherwise. After finding suitable applications, the next step was to find out what the advantages and disadvantages were to using each application by searching around different review sites that have been referenced in the later references section.

1.2 Research Analysis

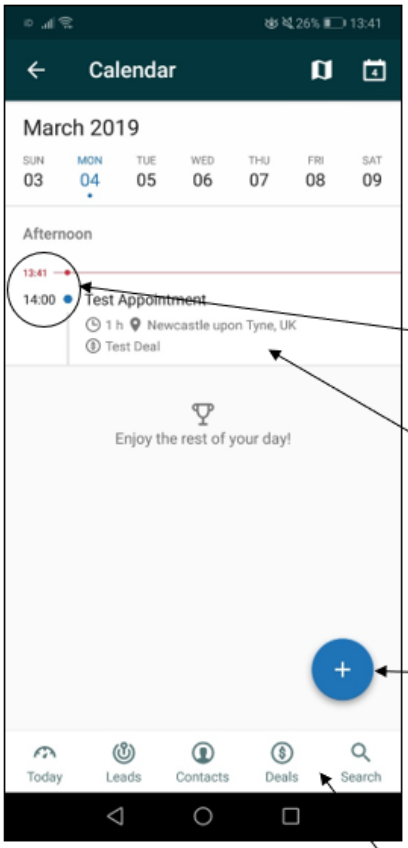
When it came to research current apps in the same field, the team first had to consider which platform would be desirable to design the app for. Through research, it became apparent that Android accounts for more than 80 per cent of all smartphone sales to users worldwide (as of Q2 2018, see image below). Given this fact, it was decided that it would be the best option to develop our app to cater to Android phones rather than Apple. With that being said, the team chose to develop our app in Android studio using Java, a language that everyone was all familiar with. Below is a graph of various Operating Systems and their current use in the world.



When researching current apps on the market, it became apparent that many of the effective apps followed the same colour scheme that their companies adopted. One app in the same CRM section of the app store known as 'Zendesk Sell' maintained a consistent colour scheme with their company's logo and its website. This gave the product a familiar and professional feel and would be wise to adopt the same consistency in our application.

Zendesk Sell has a remarkably consistent layout allowing their users to carry out tasks such as scheduling and communication with clients both simply and easily. This is what is expected of the final application build for KCS. The application should not be a chore to use and therefore allow it to be recommended as an application that can be used for salespeople to be efficient when contacting and meeting customers. One of the problems that Zendesk Sell faces, however, is a large amount of data that is being passed through. The export options that the platform has been limited, and the resulting database can be too large to handle within Microsoft Excel. Though the application allows for individual records to be exported, relevant information is left out and therefore requires manual input that is time-consuming and inefficient. To enable the application to for KCS to perform as intended, exporting information should come with options that will enable any information to be either output or left alone, without causing any issues with the amount of data at hand. To accomplish this, it may be worth looking into various databasing methods that can handle large amounts of data.

Below is a detailed set of pictures from the Zendesk Sell App:



This is the calendar page.

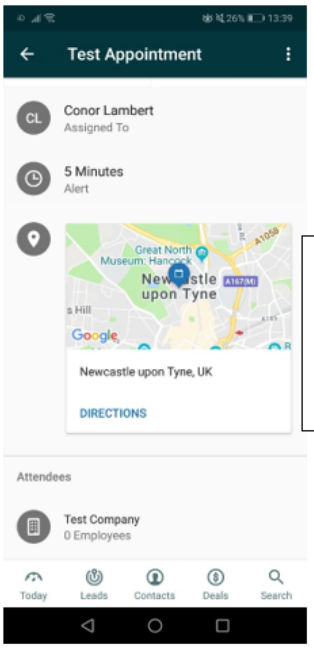
The event made shows the current time against the time of the upcoming events.

If the event needs to be changed, it can be easily done by clicking the event and then edit.

The user can add an event to the calendar.

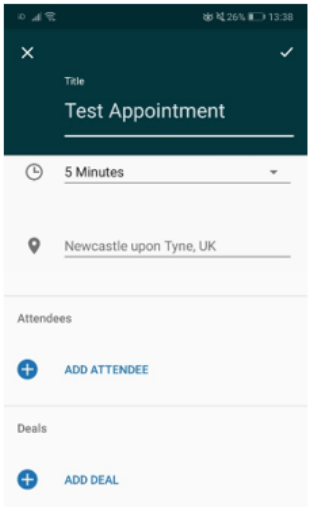
Easy to use navigation bar.

These images are from the Appointment page.



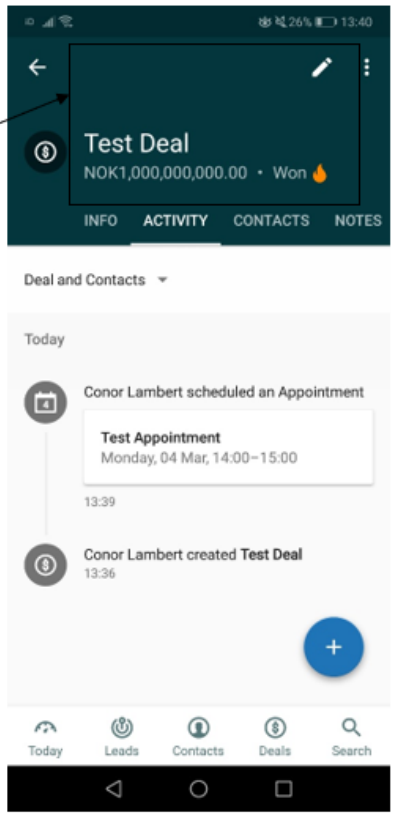
If an appointment is created with a location, the location can be seen with a built in Google maps app.

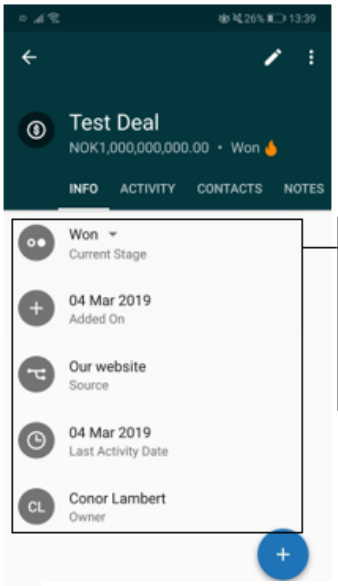
The user can create an appointment with the appropriate attendees and the deal at stake.



This is the screen where a user can create a “Deal”.

The user can edit the name, the value and the status of a deal.



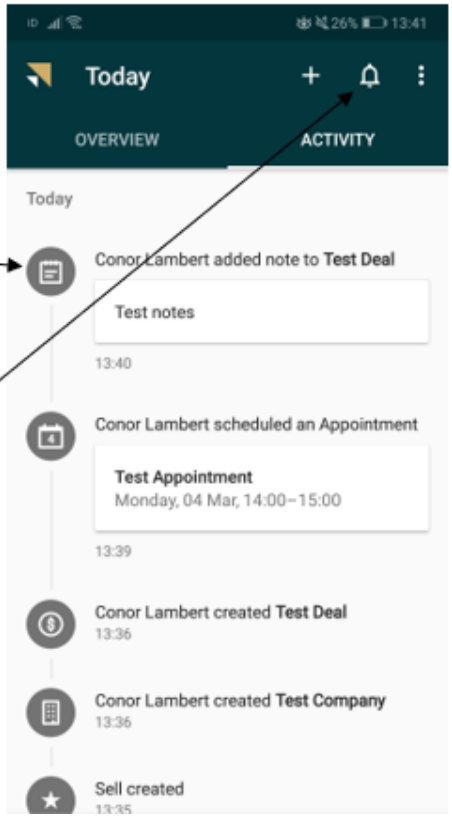


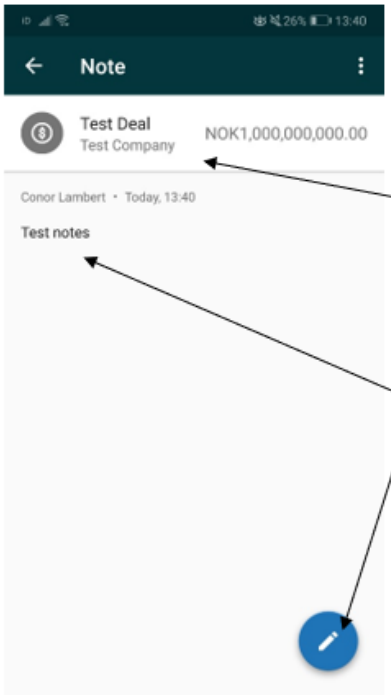
The user can also get more information about each deal by scrolling though the available tabs.

This is the activity page. It provides an idea of what has been done by the user.

Clear description of the activity.

Easy access to notification tab.

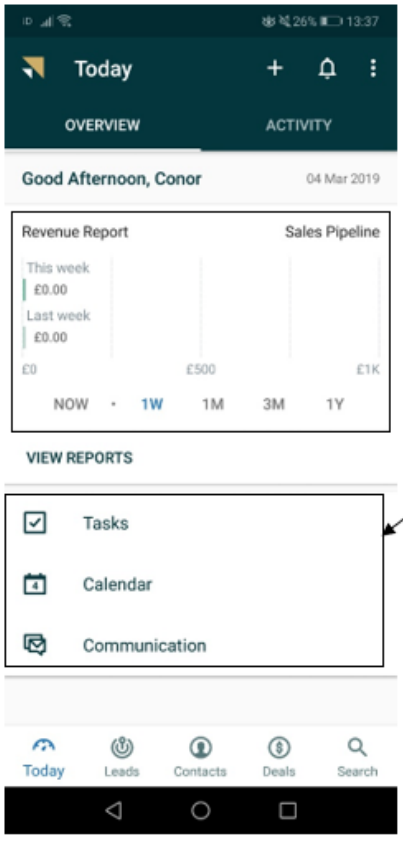




This is the notes section. It is not very intuitive.

The user can create a note and add the relevant Deal to the note for reference.

This is where the user writes their notes. Changes are saved automatically.



Finally, this is the overview screen. The user can see relevant information about their progress.

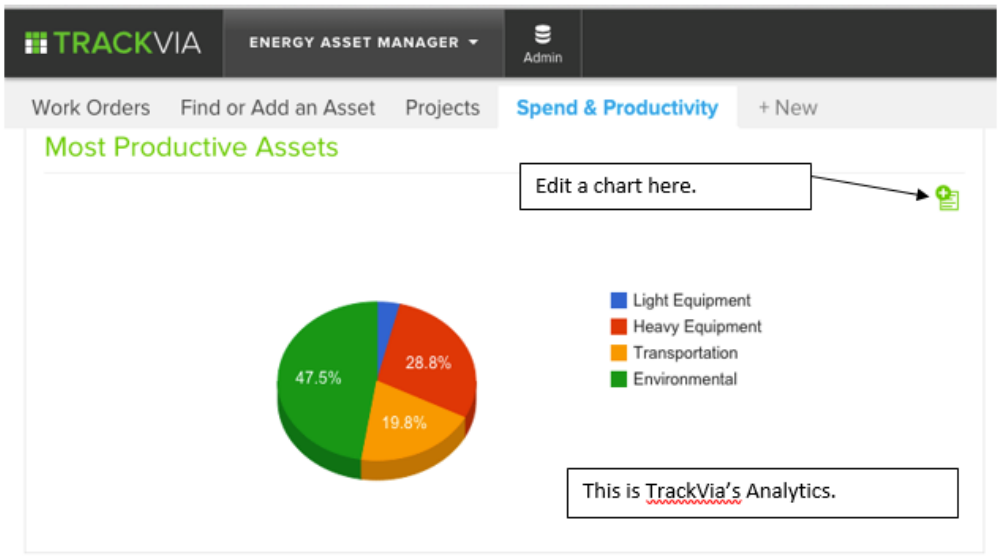
The user can see their revenue here.

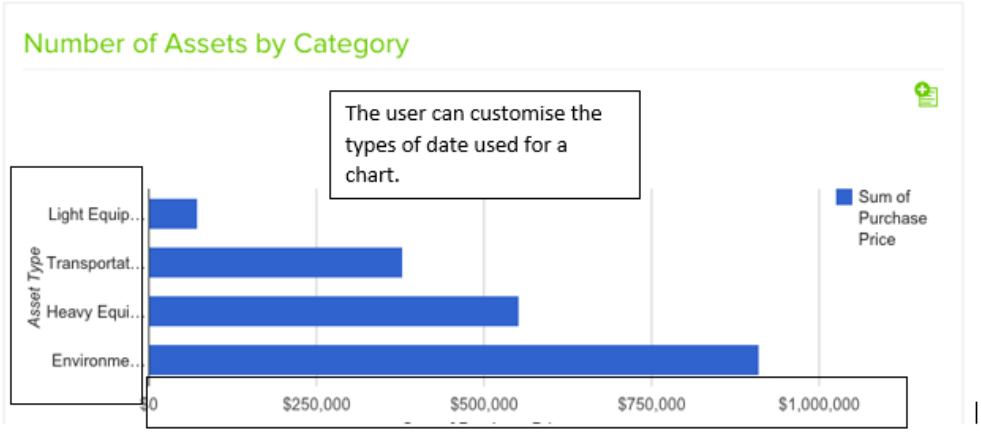
Easy access to other areas of the app.

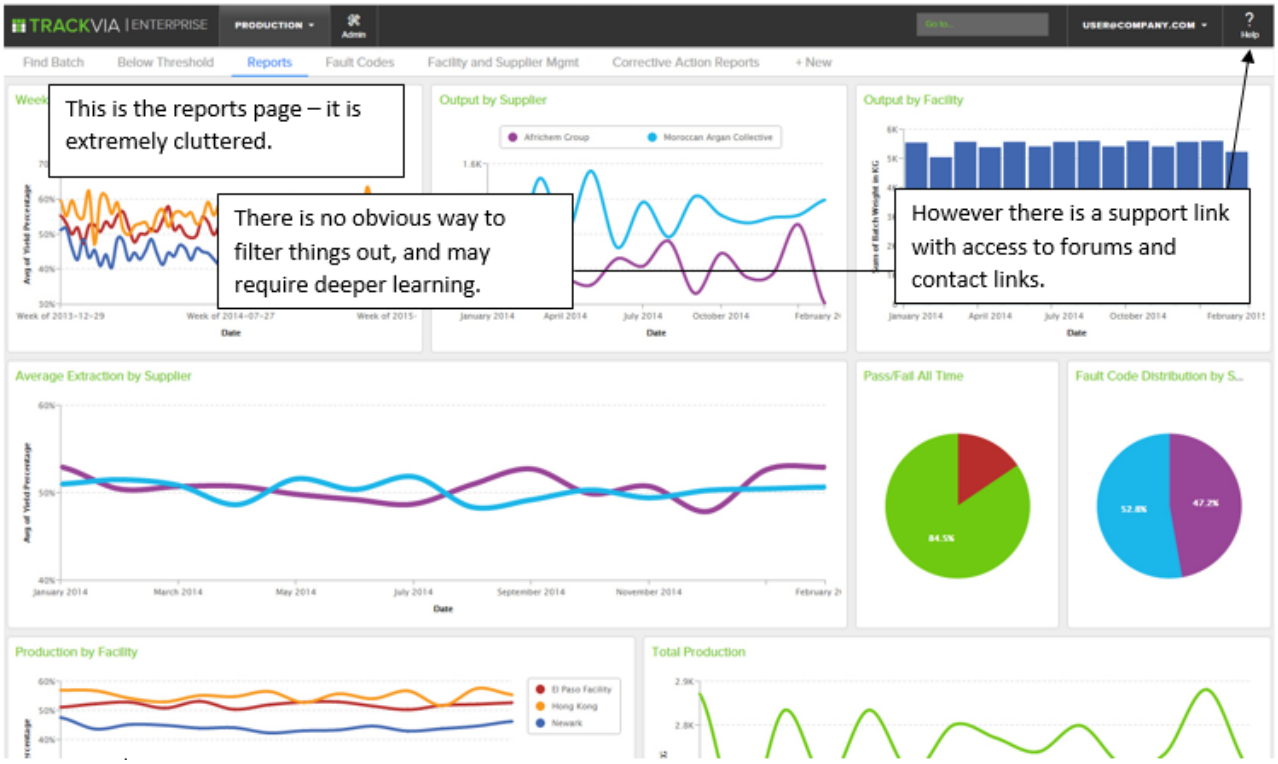
During the research, it was realized that our application needed to include several key features to make it an effective CRM application. To begin with, it was clear that the application needed to include several unique features that would make it worth considering and used by the salespeople. An example of one of these platforms is TrackVia.

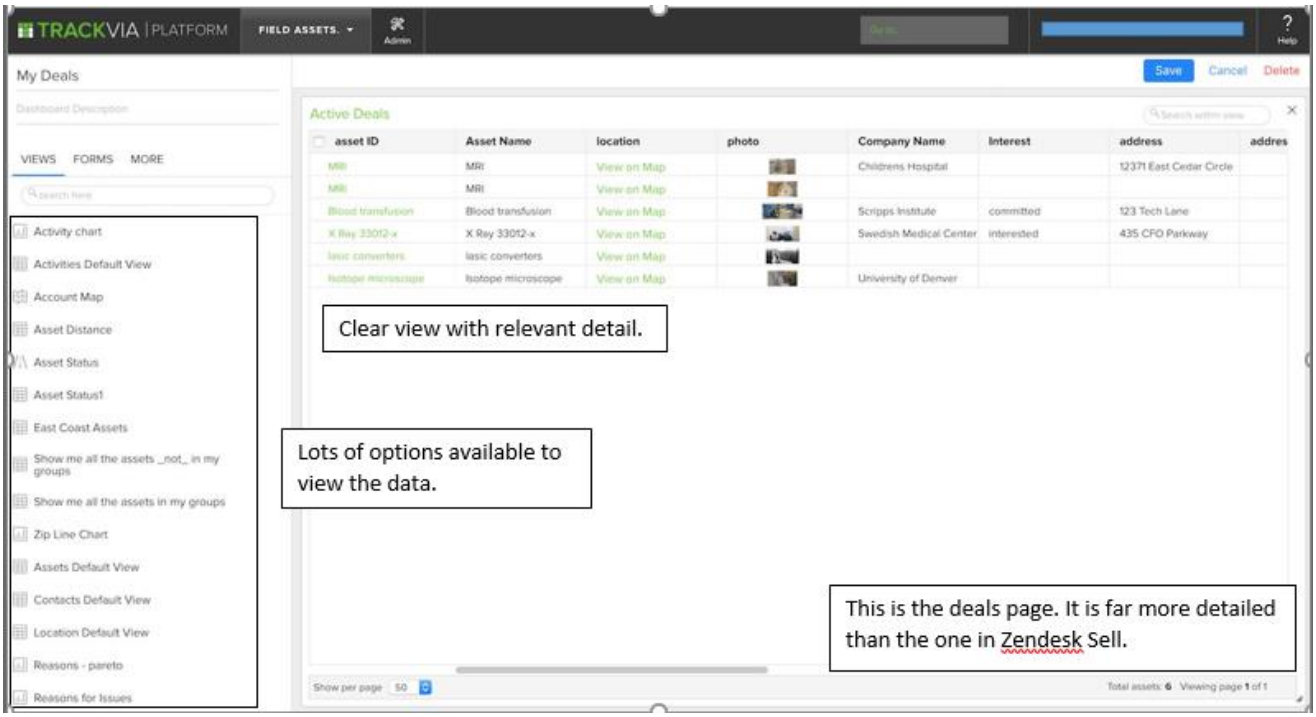
One of TrackVia's main advantages over other types of software is that it is very intuitive, making it easy to learn for a lot of people looking for a sales application. Despite this fact, however, the layout of the application and its overall design could be improved to provide a more comfortable suite for salespeople. Furthermore, a lot of people who gave reviews on the system said that transitioning to another version of the application was not easy as significant things had been changed or improved, slowing the process of a transition incredibly. However, TrackVia's analytics system works well, allowing users to have real-time statistics based on their work. This may include factors such as time spent somewhere, or profit made. Due to this fact, TrackVia is easy to use outside of the office and on the road to and from customers, especially with trackers that allow for current meetings and notes.

Below is a set of detailed images of TrackVia's System; unfortunately, the application on the phone requires an actual company to be set up and therefore cannot be done – so the images will be taken from Google and references will be available in the references.



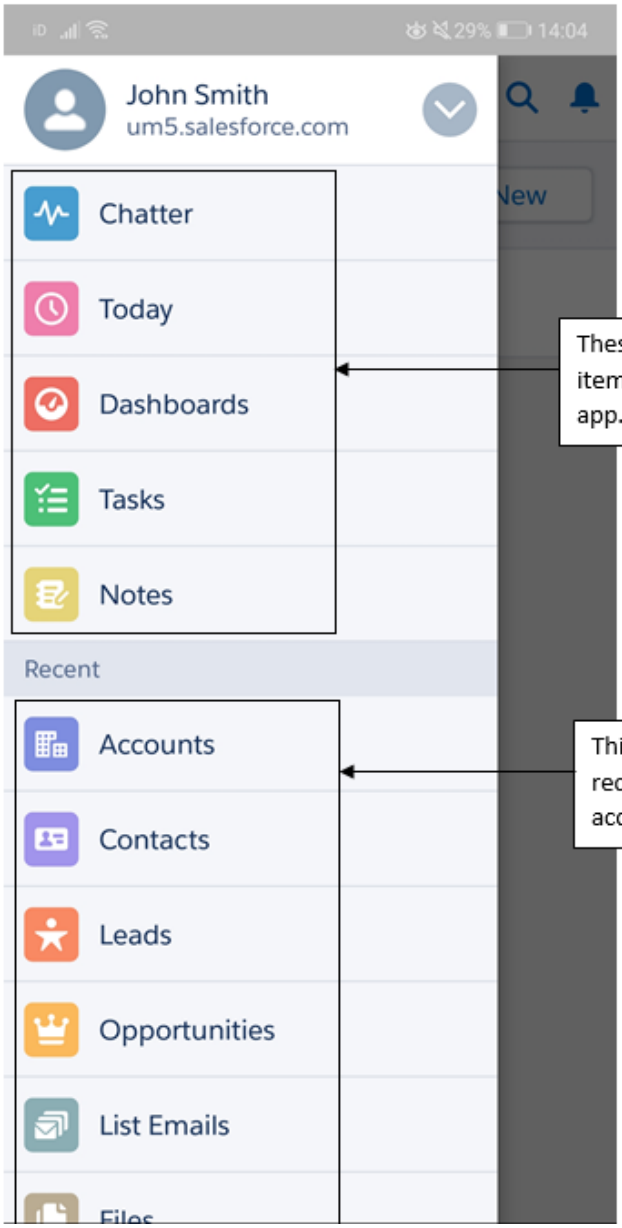



TRACKVIA | ENTERPRISE
PRODUCTION
Admin
USER@COMPANY.COM
Help
Find Batch
Below Threshold
Reports
Fault Codes
Facility and Supplier Mgmt
Corrective Action Reports
+ New
This is the reports page – it is extremely cluttered.
There is no obvious way to filter things out, and may require deeper learning.
However there is a support link with access to forums and contact links.
Output by Supplier
Output by Facility
Average Extraction by Supplier
Pass/Fail All Time
Fault Code Distribution by S...
Production by Facility
Total Production


TRACKVIA | PLATFORM
FIELD ASSETS.
Admin
Help
My Deals
Save
Cancel
Delete
VIEWS
FORMS
MORE
Activity chart
Activities Default View
Account Map
Asset Distance
Asset Status
Asset Status1
East Coast Assets
Show me all the assets _not_ in my groups
Show me all the assets in my groups
Zip Line Chart
Assets Default View
Contacts Default View
Location Default View
Reasons - pareto
Reasons for Issues
Active Deals
asset ID
Asset Name
location
photo
Company Name
Interest
address
Clear view with relevant detail.
Lots of options available to view the data.
This is the deals page. It is far more detailed than the one in Zendesk Sell.

Salesforce is a very highly rated platform and offers a vast variety of features that are useful for all salespeople while they are meeting customers or managing the data that they have collected over a specified period. Salesforce provides an external product called Trailhead which allows newer users to learn how to use salesforce much faster than just self-learning. This allows quick and easy integration with a company. The analytics suite provided offers a considerable range of charts and graphs that provide a clear overview of how a business is performing, whether it be how many sales have been made, ratios of profit to expenses and much more. Salesforce provides a platform that can be adapted easily to the changing needs of a company, meaning that a company doesn't need to worry about overhauling their entire operation to meet new marketing demands. Despite how good Salesforce may appear, however, all these good qualities come at a major cost, and that is the fact that the interface is incredibly complex and can be difficult to navigate and take full advantage of. Despite having a learning platform for Salesforce, it is still a steep learning curve. There is a lot to learn before a user can take advantage of the full functionality. Although this is perfect for large organisations who have various sections and specialisations in certain areas, it can be difficult for solo businesses.

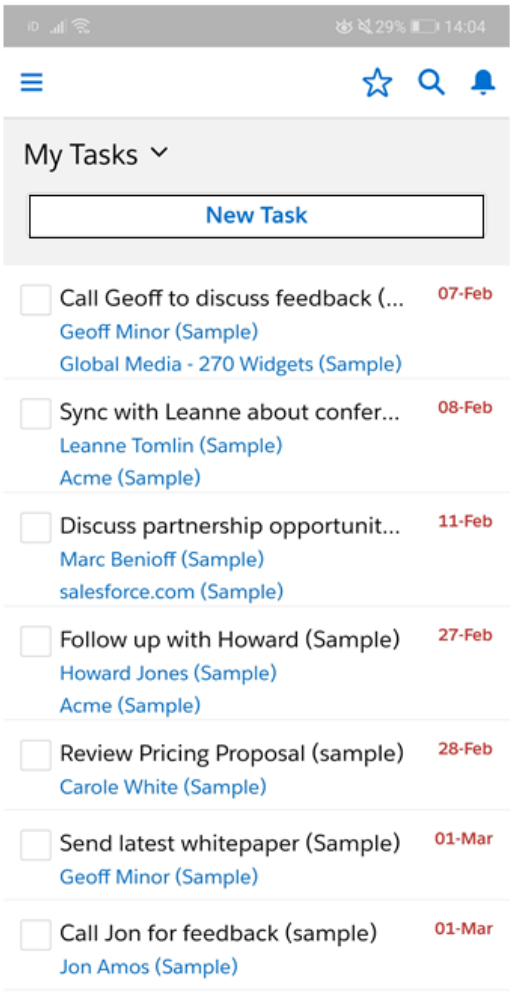
Below is the breakdown of the system:



This is the main Navigation tab. It is accessed by clicking the burger icon on the home screen.

These sections give access to the main items that are used throughout the app.

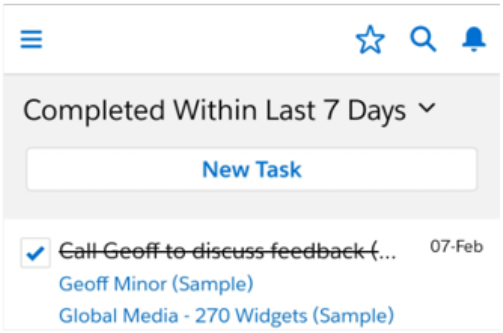
This section provides access to recently edited sections for quick access to previous changes.

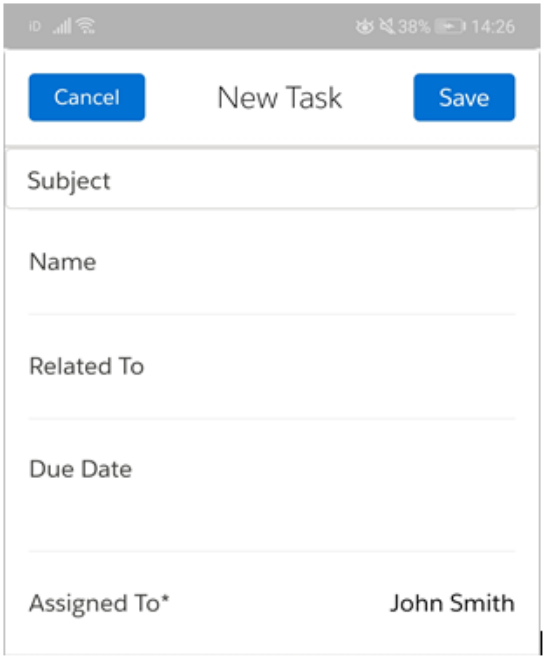


This is the user's task page.

Tasks are easily created with this button.

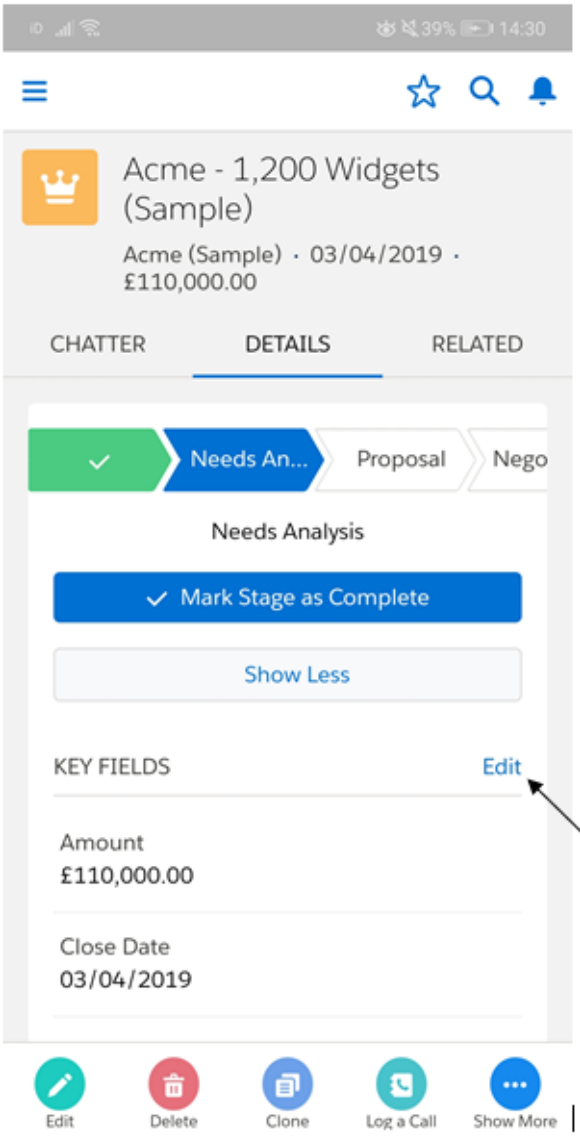
Completed Tasks can be ticked off at any time.





This is the page for creating a new task.

Basic information can be input for basic task and an employee's name can be entered for whoever should be assigned to it. That employee will receive a notification by email and in app.



This is the "deals" screen.

The user can cycle through each stage to view the progress of the deal.

The user has plenty of information about the deal and it is clearly laid out for ease.

Information at each stage can be easily edited.

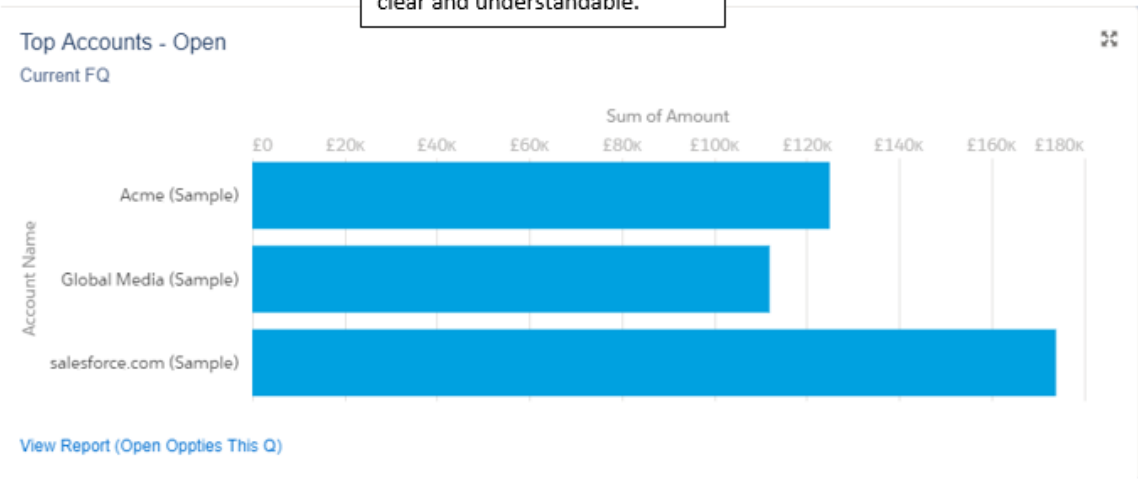
It should be noted that when using the application to generate the above screenshots, the phone being used is relatively high-end and new and the application was performing considerably worse than expected for a highly rated application. One thing that was particularly frustrating about the application is that there is no access to this fantastic analytics suite that the website boasts. Unfortunately, if the user wants to see the analytics on the page, they must load up the web app to see this information.

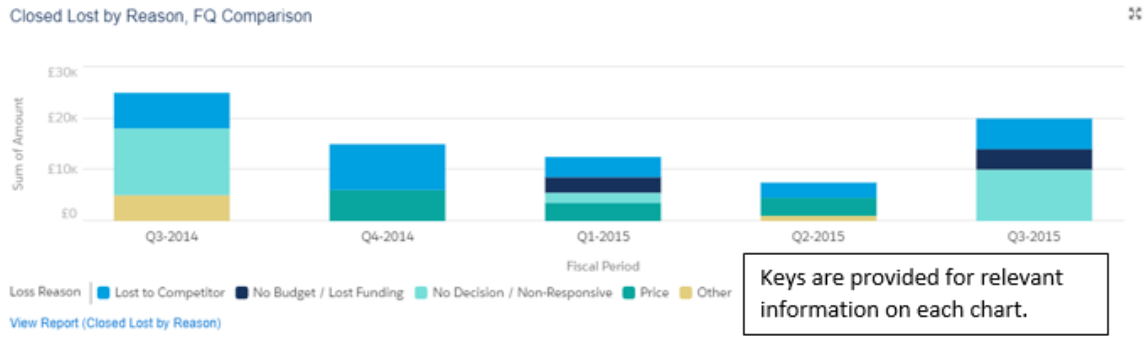
Furthermore, trying to navigate the far more in-depth web app requires a steep learning curve which means that transitioning to using the software is incredibly tedious, something that must be avoided when creating the application for KCS. The user must be able to access all the information from their phone, as this is what was initially projected by the team.

This is the whole dashboard for one of the deals created:

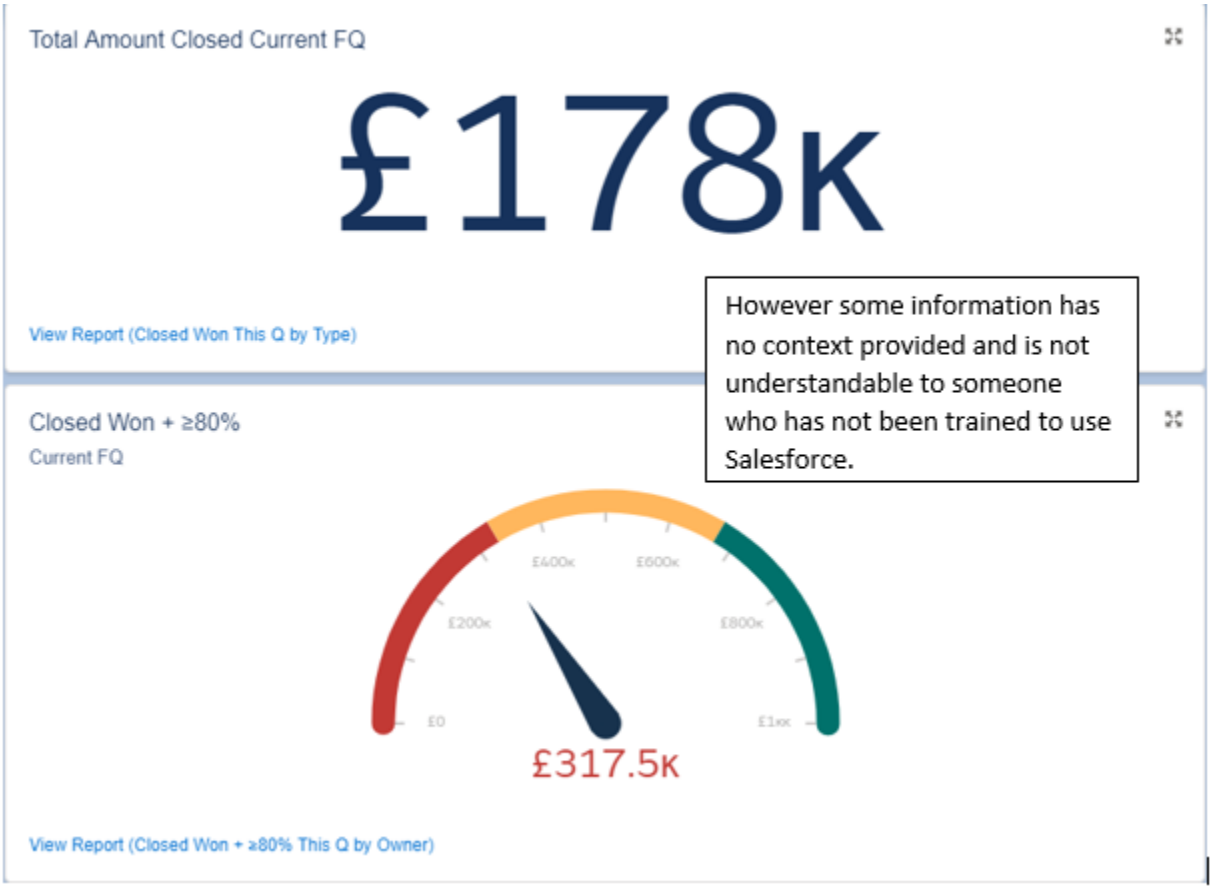
There is a lot of information available on the screen and therefore can be broken down further:

The information on each chart is clear and understandable.





Keys are provided for relevant information on each chart.



To conclude the research on other CRM applications, the KCS application must be simple, easy to use and comfortable for new salespeople. However, being simple does not mean it cannot be intuitive and therefore should aim to have a good analytics suite that provides various statistics based on the customers and the profits made over different periods.

1.3 Project purpose

The team has been asked by to design and build a mobile application for the sales company Kerridge Commercial Systems. KCS requested that an application is built that their salespeople can use to plan customer visits, receive updates on how a customer's account is doing, as well as pass information back to the office after a visit. In brief supplied by KCS, they requested that the team should consider implementing a few specific features into the application. Some of these would be basic contact information, a history of previous visits, recent sales to the customer and a feed of recent company events (using either social media or news sources).

2 Roles and Deliverables

2.1 Team Roles and Responsibilities

Team Leader

Cédric's role is to ensure the success of the project. He must establish proper coordination between the team members as well as make sure each member has a fair amount of work and help settle disagreements. He oversees the accomplishment of the tasks by each deadline and therefore, in charge of the project management software. He keeps track of the minutes and produces the agenda for the meetings. He also handles the liaison with the Module Leaders.

Documentation

Nathan and Conor's roles are to take the information from the programmers about functions and operations and write it up in a way that could be understood by anyone without experience. It should read clearly and accurately to ensure that the document covers all ends of the application. They also oversee proofreading of any formal documents related to the project including the project specification and team meeting minutes.

Senior Programmer

Ben and Jason are advanced-level programmers who have an in-depth knowledge of programming concepts and of the programming language. Ben is the Lead Programmer; he links the Team Leader and the Programming Team. He is responsible for the soft deadlines related to the programming part of the app. Jason will take a senior programmer role and will work with the lead designer to ensure that the design is technically feasible and to produce a high-level architecture for the solution.

Junior Programmer

Ross, Matthew and Raven are junior level programmers who will take direction from the senior programmers in developing various aspects of the application. Whilst discussing their ideas and plans amongst themselves and with the senior programmers, the programming work shall be distributed evenly.

Tester

All the team members that are not Senior or Junior programmers will try to spot any deficiency of the system by running some code testing as well as unexpected usage of the built app. They will produce a bug report to allow the programmers to fix these bugs. This is done to get a broader range of perspectives about how the app can be used as one user may encounter a bug that another would not come across.

Web Developer

Cédric (helped by the programming team) oversees the development of the marketing website. This role will have to decide the framework and development process to use for creation of the website and may produce specifications for the programming team to follow.

Lead Designer

Nick is the Lead Designer. He translates ideas into blueprints. He must establish the design of the app with guidelines to follow to achieve the implementation of the design. He will aim to produce an app layout which is both functional and aesthetically pleasing, which will be possible through research into common mobile app layouts and colour schemes.

On top of all the roles that have been split for each team member, there is also the concern of testing the system. The idea behind how this will be that everyone in the team will contribute to the testing aspect of the programming however the majority should be done by someone who did not work on a certain part of the system will be the tester for that part. That way they can clearly spot where there are problems or attributes that could be refined to make the application more of what the client is expecting as a final product.

2.2 Deliverables

Mobile application

The primary deliverable of the project is to create a fully-featured application for the Android platform which meets the requirements of the client. The application is targeted for salespeople completing client visits and therefore must be simple to understand for non-technical users. The application will also follow modern design and usability principles. The primary key features of the application are to schedule meetings with KCS's customers, to record notes for those meetings and for the application to be scalable to multiple salespeople. Data created by the salespeople will be seamlessly and securely uploaded to the remote database.

Interactive website

The secondary deliverable of the project is to create an interactive website to advertise the application's features. This website will also have the extended functionality of being able to view meeting notes that salespeople have taken within the mobile application, on the website.

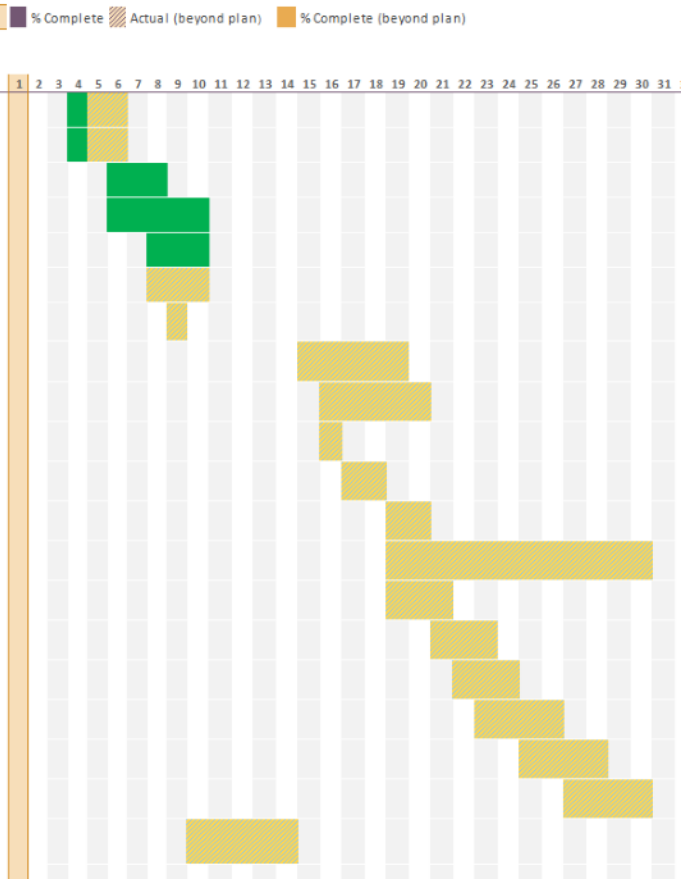
3 Project Plan

Team 8 Project Plan

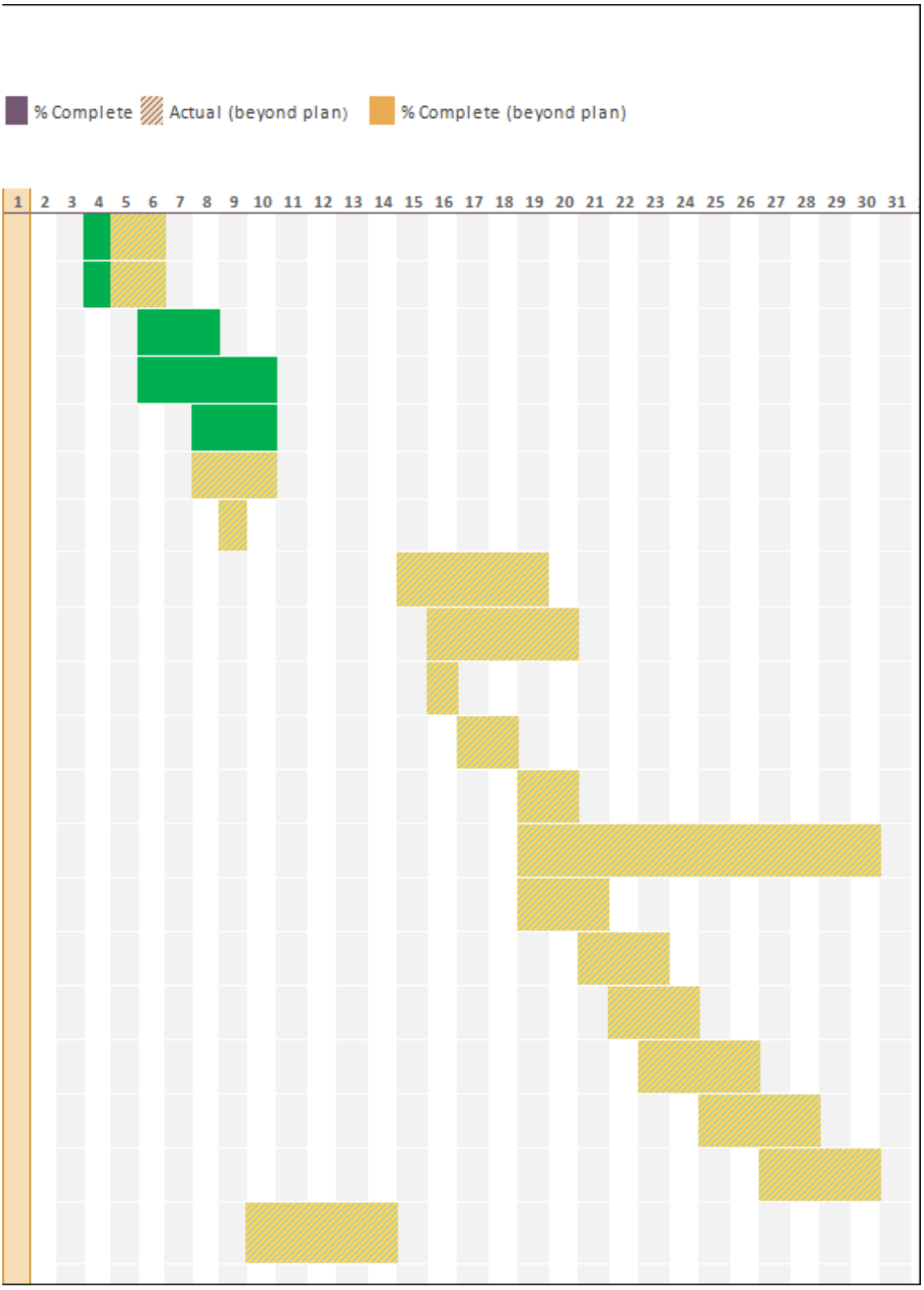
Select a period to highlight at right. A legend describing the charting follows.

Week Highlight: | % Complete | Actual (beyond plan) | % Complete (beyond plan)

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Design UI	4	3	4	1	100%
Plan Tech to use	4	3	4	1	100%
Dragon's Den	6	3	6	3	100%
Draft Specification	6	5	6	5	100%
Revised Draft Specification	8	3	8	3	100%
Test Plan	8	3			
Peer Review	9	1			
Design Diagrams/ UI designs	15	5			
Basic Implementation of App	16	5			
Implement Login Screen	16	1			
Implement Home Screen	17	2			
Create Database and Fill Mock Data	19	2			
Full Implementation of App	19	12			
Implement Calendar	19	3			
Implement Meeting Info	21	3			
Password Salt/Hashing in Database	22	3			
API	23	4			
Local Data Caching	25	4			
Deal with No Signal Situations	27	4			
Final Requirements Spec and Test Plan	10	5			



ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	Week Highlight: PERCENT COMPLETE
Design UI	4	3	4	1	100%
Plan Tech to use	4	3	4	1	100%
Dragon's Den	6	3	6	3	100%
Draft Specification	6	5	6	5	100%
Revised Draft Specification	8	3	8	3	100%
Test Plan	8	3			
Peer Review	9	1			
Design Diagrams/ UI designs	15	5			
Basic Implementation of App	16	5			
Implement Login Screen	16	1			
Implement Home Screen	17	2			
Create Database and Fill Mock Data	19	2			
Full Implementation of App	19	12			
Implement Calendar	19	3			
Implement Meeting Info	21	3			
Password Salt/Hashing in Database	22	3			
API	23	4			
Local Data Caching	25	4			
Deal with No Signal Situations	27	4			
Final Requirements Spec and Test Plan	10	5			


% Complete
Actual (beyond plan)
% Complete (beyond plan)
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

4 Hardware and Software Resources

Hardware

Client-side
- Android Smartphone

Backend
- Web Server
- MySQL Server

Software

Design and Diagrams
- Lucidchart
- Adobe Photoshop
- Adobe Illustrator
- Adobe XD
- ERDPlus
- draw.io

Client-side Development
- Android Studio
- GitHub

Backend Development
- NodeJS
- GitHub
- MySQL
- Visual Studio Code
- PuTTy

Organisation and Documentation
- Asana
- Discord
- Microsoft Word
- Microsoft Excel
- Microsoft PowerPoint
- Facebook Messenger

5 References

Capterra, *Salesforce Reviews by Customers*, accessed 18th February 2019, <https://www.trustradius.com/products/zendesk/reviews/pros-and-cons?cg=large>

G2Crowd, 2017, *TrackVia Reviews by Customers*, accessed 18th February 2019, <https://www.g2crowd.com/products/trackvia/reviews>

Statistica, 2018, *Global market share held by the leading smartphone operating systems in sales to end users from 1st quarter 2009 to 2nd quarter 2018*, accessed 23rd February 2019, <https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/>

TrackVia, *TrackVia Charts on parent forms: A hidden gem*, accessed 6th March 2019, <https://www.trackvia.com/blog/technology/trackvia-charts-parent-forms/> (Images used for TrackVia Breakdown)

TrustRadius, *Zendesk Reviews by Customers*, accessed 18th February 2019, <https://www.trustradius.com/products/zendesk/reviews/pros-and-cons?cg=large>

Zapier, 2016, *TrackVia - Features, Pricing, Alternatives, and More*, accessed 6th March 2019, <https://zapier.com/apps/trackvia> (Images used for TrackVia Breakdown)

6 Definition of terms

Abbreviation	Definition
KCS	Kerridge Commercial Systems
App	Application
API	Application Programming Interface
CRM	Customer Relationship Management
Android Studio	The IDE used to create the application for KCS
GUI UI	Graphical User Interface such as the one created for the application
ER	Entity-Relationship
JS	JavaScript programming Language
SQL	Structured Query Language
MYSQL	The implantation of SQL used for backend storage
Node.JS	JavaScript runtime for backend
Java	Object oriented programming language
Material Design	Design language developed by Google
XML	Extensible Markup Language, a markup language used heavily in android development
REST	Representational State Transfer
SDK	Software Development Kit, a set of tools used for development for a specific platform

7 Solution requirements

7.1 Functional Requirements

Requirement	Priority	Supplier Compliance
When opening the app, the user is required to sign in using their email and password	H	Full
If the user does not have an account, they are provided with an option to register a new account	H	Full
Once successfully logged in, the user will be taken to the salesperson homepage	H	Full
User is able to view a calendar display of meetings	M	Full
User is able to view meeting Information	H	Full
User can modify meeting Information	M	Full

User can update the database	M	Full
User can access recent customer page	M	Full
Within the recent customer dialogue, the user can view past appointments	M	Full
The user can view customer contact details	H	Full
The app displays a LinkedIn Feed	L	Full

7.2 Non-Functional Requirements

Requirement	Priority	Supplier Compliance
When attempting to log in, the app verifies that the email address and password belong to an account	H	Full
If a new user is creating an account, the app should ensure that the email and password are valid before allowing an account to be created with these details	H	Full
Within the view meeting dialogue, the app verifies the validity of any edits made to an order	H	Full

8 Other considerations

8.1 Assumptions

Mobile Application Assumptions

- Every user will be a member of KCS with an already registered KCS account consisting of a KCS email and a unique password
- Every user will be using an android phone running a minimum of Android 6.0 Marshmallow
- Every user will be using a phone with hardware capable of running the application
- Every user will have a phone with access to the internet

Database Assumptions

- All users are employees
- All users will have a client
- All users will have a KCS email address
- An order with a client will be assumed to be an ongoing subscription

8.2 Constraints and Dependencies

This application was developed from the start with Android users in mind. Cross-platform principles were considered by the team when we first began planning the application; however, due to the complexity of doing so and the programming team's strength in Java and android studio, it was decided that an android app would be created. What further aided this decision was the team's research into smartphone sales and revealing that Android phones account for over 80% of the worlds smartphones sales, so if we were only going to focus on one operating system for the initial build of the app, it made sense to use Android. Ideally, we would also like to develop the app for Apple iPhones, given their popularity and reliability. However, due to our time constraints, we were forced to pick just one platform to develop the app for. This does, therefore, mean that the user of the application would need an android phone to run the application.

When developing the database for the storage of information it will be a dummy database purely for demonstrational purposes. None of KCS' APIs will be used. This means that there is the risk of designing a database that is not a suitable representation of what data would be required which may be confidential and the team will not be able to gain access to. This means that a dummy database as accurate as possible needs to be built to give a fair representation of the needs of the client.

Some of the program developers may be lacking knowledge in areas required for the development of the app such in database design and the implementation of the database. In situations like these the optimal way of dealing with this issue would be to have the programmer with the most knowledge in each area work on it and learn what is required as they do so. This will limit the time it takes to develop the feature without costing too much time for the programming team.

The mobile application uses the internet to access the company's database, place orders and fetch client information. Therefore, a constraint in this scenario would be if the user loses their internet connection completely. We have tried to combat this constraint by adding the feature of being able to locally save data to the app until an internet connection is found. However, if the user was never to regain a connection again, or their phone loses this capability, the data can never be sent to the database.

Similarly, if the external database were to go down, then the user of the application would have no way of accessing their data, as well as placing orders for the client. The app is heavily reliant on the status of the database and has no way to combat this problem.

9 Software Design

9.1 Overview of what the software will do and not do.

The system WILL:

- Allow the user to book a meeting with a client and add the date to a calendar.
- Provide an alert for an upcoming meeting.
- Allow the user to view reports based on the number of products and orders by any clients.
- Allow the user to create notes in a meeting which are later uploaded for access through the main website.
- Allow a location to be input for a meeting and then Google maps will provide a route to the destination.
- Allow direct contact to a client.
- Cache meeting details locally until a suitable internet connection is made to upload to the central database

The system WILL NOT:

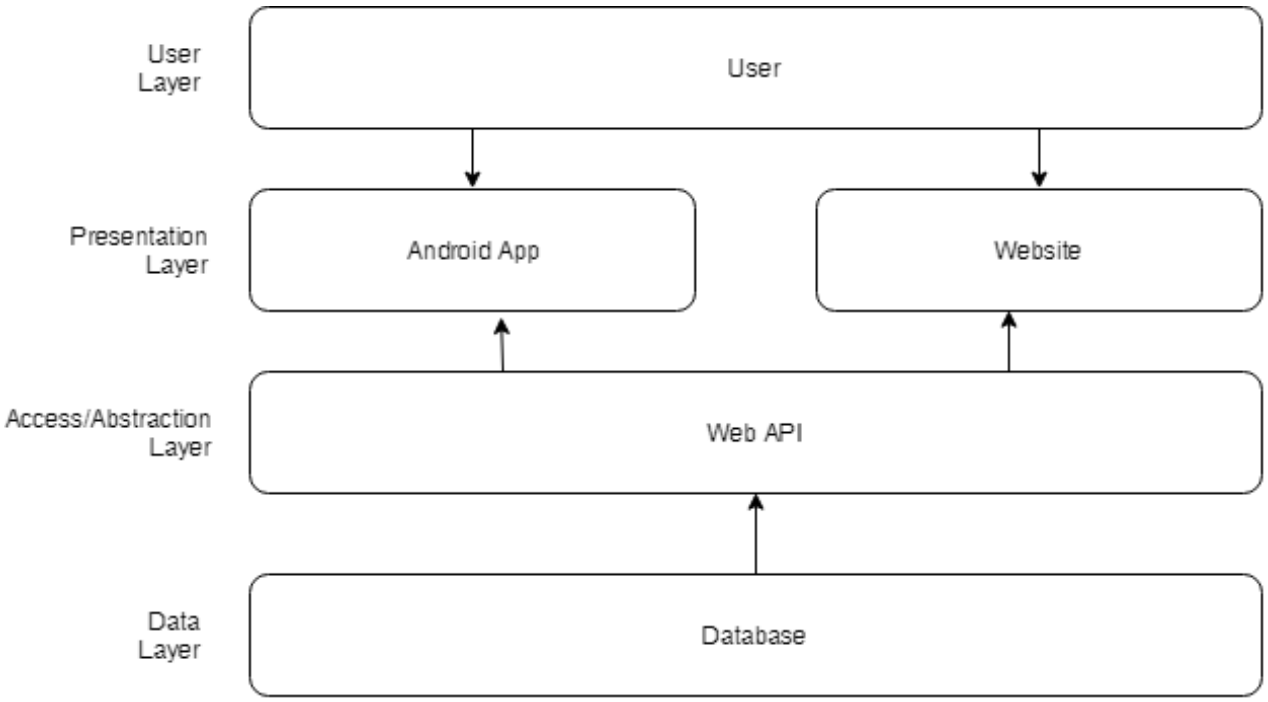
- Provide feedback on how well a salesperson has done in a week.
- Provide profit statistics.

9.2 Modifications considering comments made or changes deemed necessary

Throughout the design process the colour scheme used in the application has changed on multiple occasions, it first began as a solid white and pink colour scheme in the very first design, which then in the second iteration swapped the white for a charcoal grey which is a colour used within KCS' design principles. After some discussion the decision was made to revert to a white and pink colour scheme as the brighter aesthetic made the application much easier on the eyes. Small changes to the design of the application were also made after research into Material Design, such as the design and positioning of icons.

9.3 System Architecture – describe design decisions (Architecture diagram)

The architecture of the application follows the layered architecture design; this enables the project to have a greater level of abstraction than would be possible in other architectures. The team decided this would be the most appropriate architecture as it allows for the multiple front-end approaches without having to rework the backend layers, both presentational layer modules will access the data in the same way – via the web API.



The reason the team went with the approach, is that it is industry standard for projects that rely on remote connections, where there is a “stack” of software involved from bottom to top. The team divided the architecture into manageable sections to allow for development to be broken down into manageable chunks. The Data layer and Access/Abstraction layer together form the backend, with the Presentation and user layer forming the frontend.

9.4 High-level overview of how the functionality and responsibilities of the system are partitioned and assigned to components (deployment diagram, component diagram)

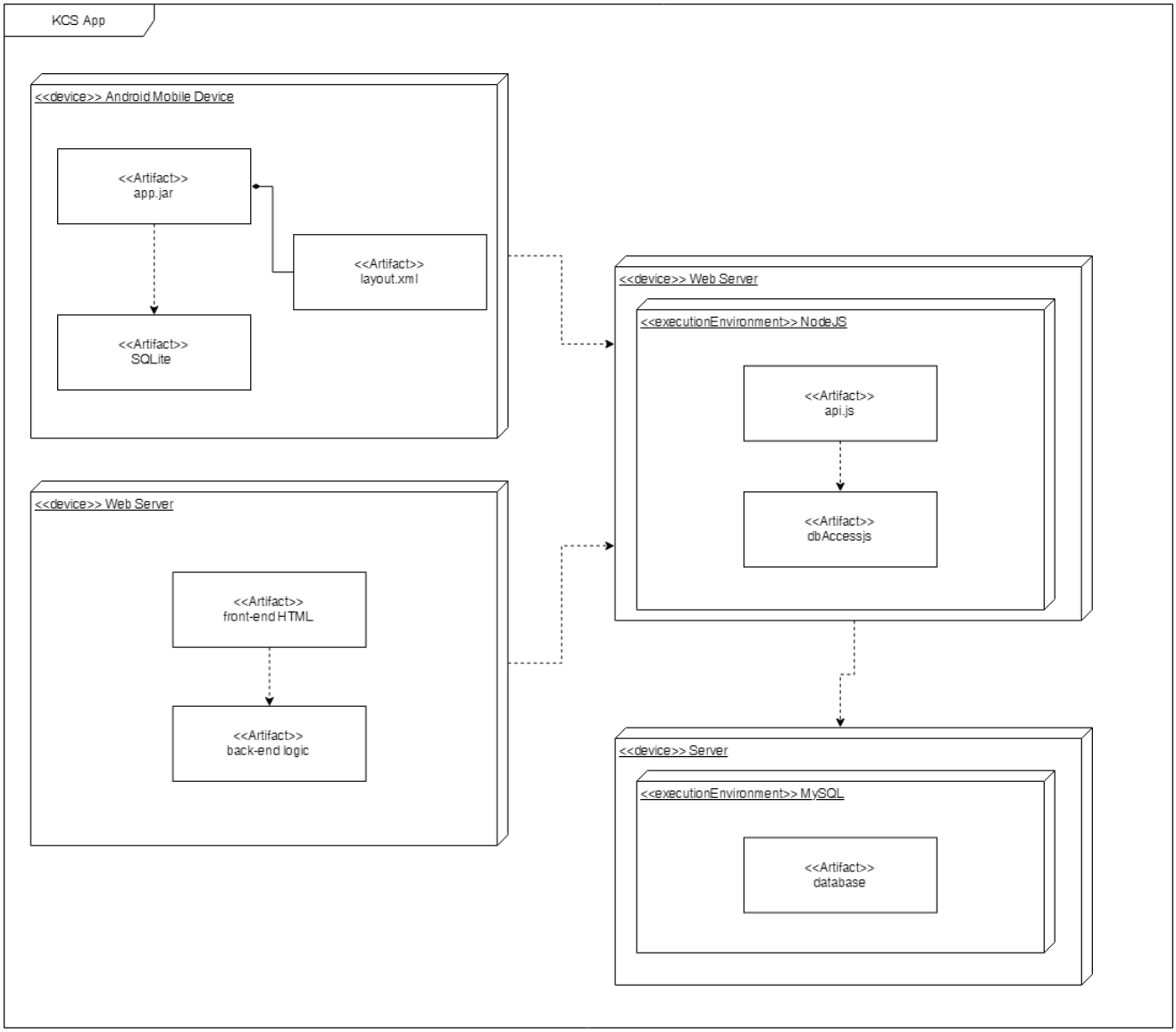
The deployment diagram shown below, show’s how the application could be deployed in the real world. In this diagram we have shown all web services running on separate devices, however this could also be deployed as services running on a singular machine.

The diagram is split in to the following sections:
The application running on the android device
This is made up of business logic, layout logic and a local caching database.

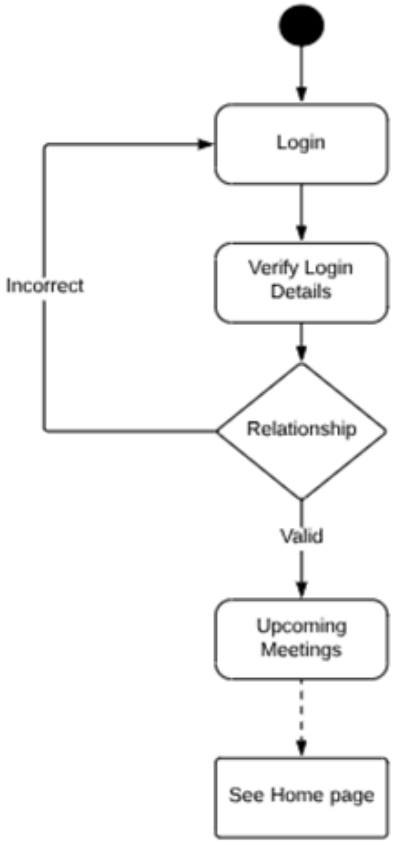
The public facing website
This is made up of frontend and backend, this website requires backend logic as the user will be able to login and view notes that other salespeople have created on the mobile application

The RESTful API
This is running a web server which allows the android app to communicate with the database, without having to maintain complex database connections. This is running on a web server running NodeJS to supply the backend logic and filtering.

The Database
This is running on a MySQL server, MySQL is an industry standard database service with a wide range of support and libraries already in existence.

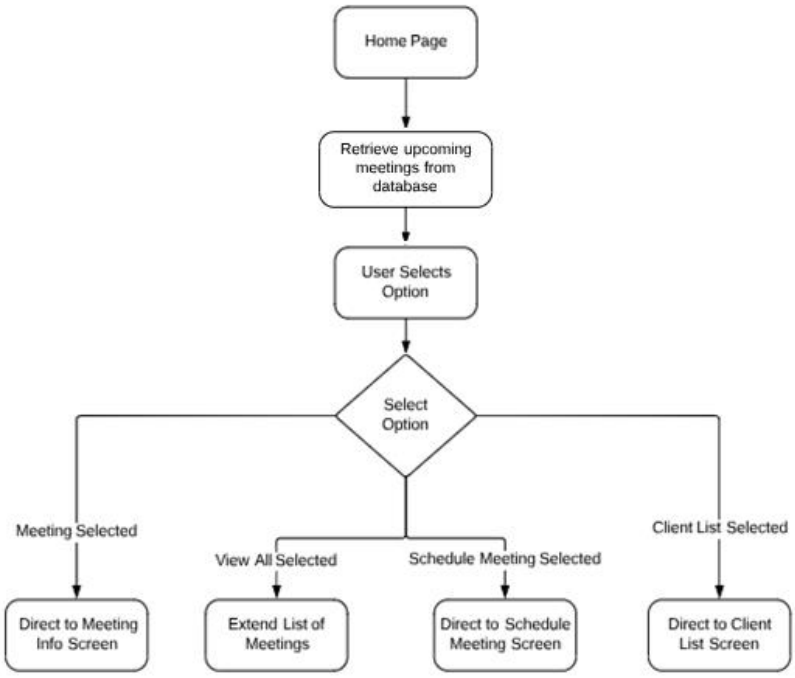


Component: Login
When using the app, the user will first be prompted to enter their login details. This information will then be checked against the database to ensure they are correct credentials. When valid they will be directed to the home page showing their upcoming meetings.



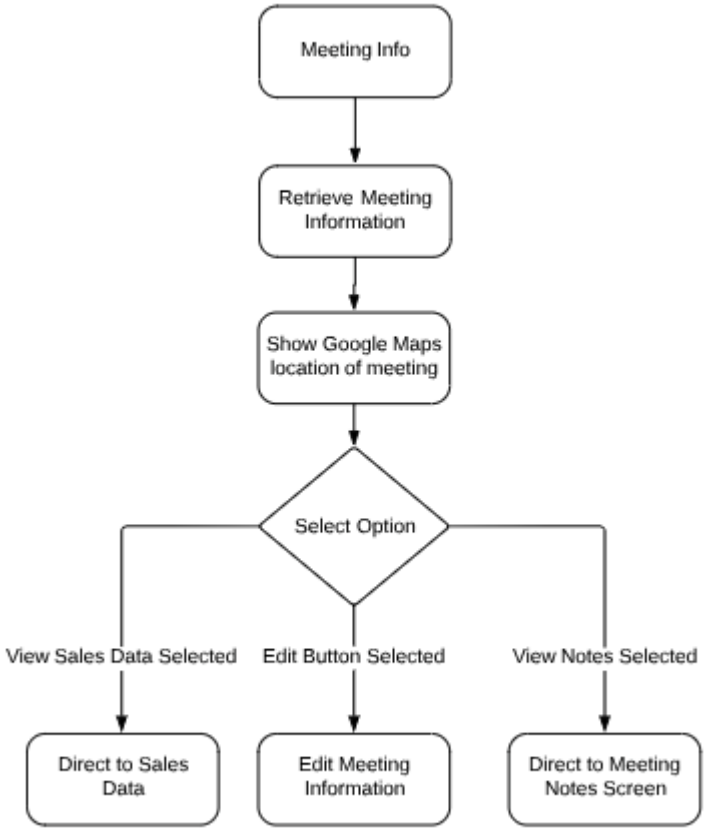
Component: Home Page

The Home page will be available to the user once they have successfully logged in. It provides information to the user based on their credentials. It retrieves information relating to the specific users upcoming meetings and populates them on the page. From here they can select a meeting to provide information relating to it. This will redirect them to the meeting information page. They can extend the list of meetings to see all their scheduled meetings or schedule a new meeting themselves. They can also select the client list button which will redirect them to the client list screen to view the full list of clients



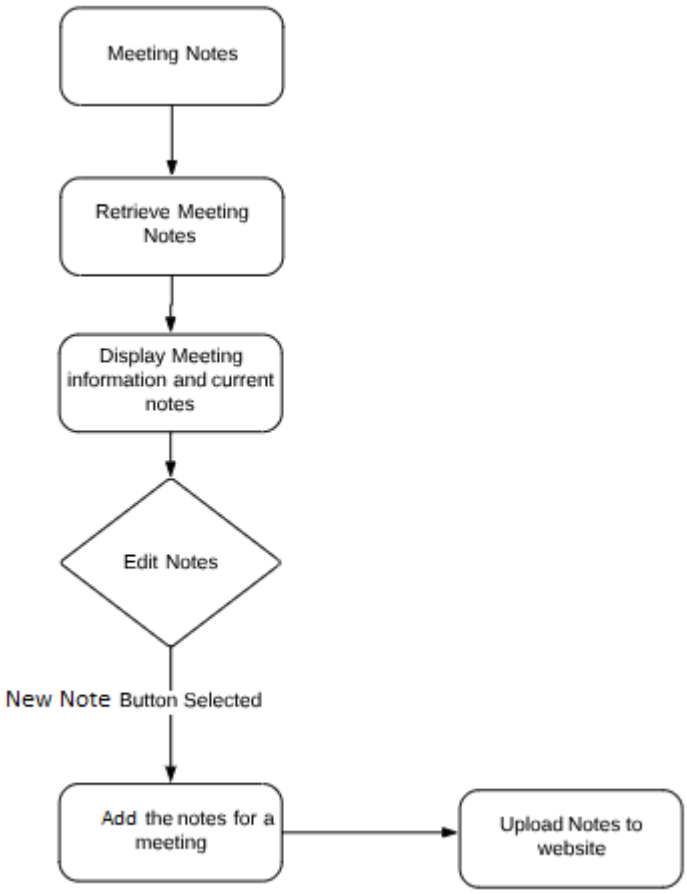
Component: Meeting Info

This Screen will show information about a selected meeting, retrieves information from the database which is then displayed to the user such as the time, location, client name and the people attending the meeting. From here the user can also see a map showing the location of where the meeting will be held. There are options to edit the meeting, view sales data for the meeting and to view the notes for the meeting. If the user wants to add notes about the meeting, they will need to select view notes taking them to the meeting notes screen where the notes can be added.



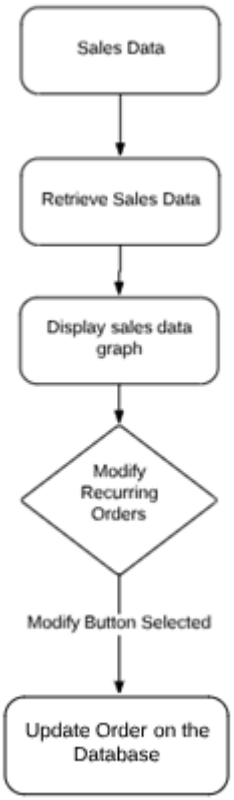
Component: Meeting Notes

This screen will show notes for a meeting. The notes can be updated at any point. It can be used to both make notes for a meeting regarding topics to discuss or to make notes regarding the events during the meeting. The notes will then be uploaded to the website which can then be viewed later. New notes can be added throughout the process allowing the notes to be dynamically updated.



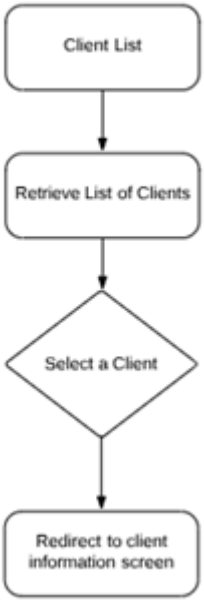
Component: Sales Data

The Sales Data screen can be used to display the sales records for a client. It provides a graph displaying the number of sales in previous months. The recurring orders for a client are displayed and can be modified by a user.



Component: Client List

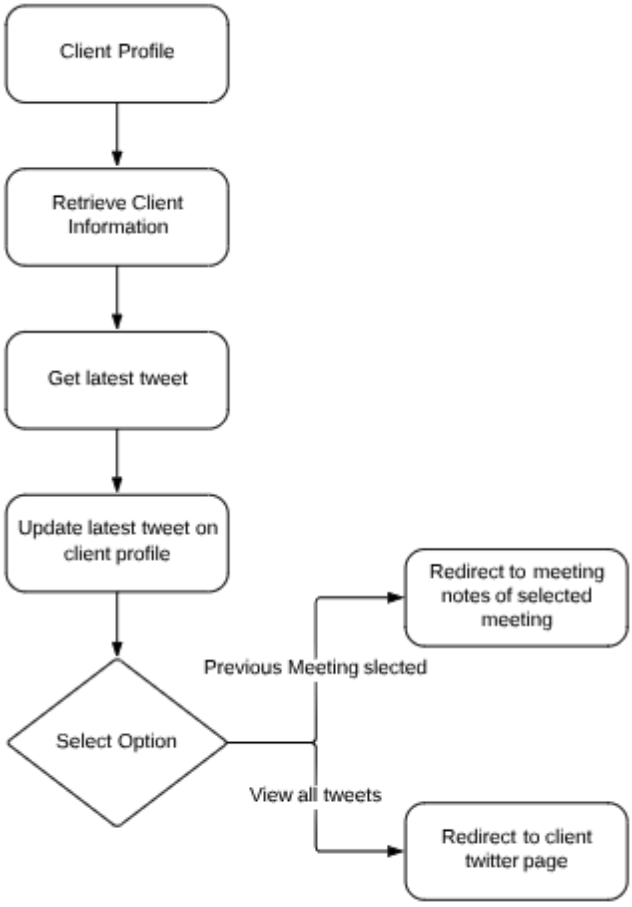
Displays the list of clients alphabetically. A client can be selected which will navigate to their profile displaying various information about the client.



Component: Client Profile

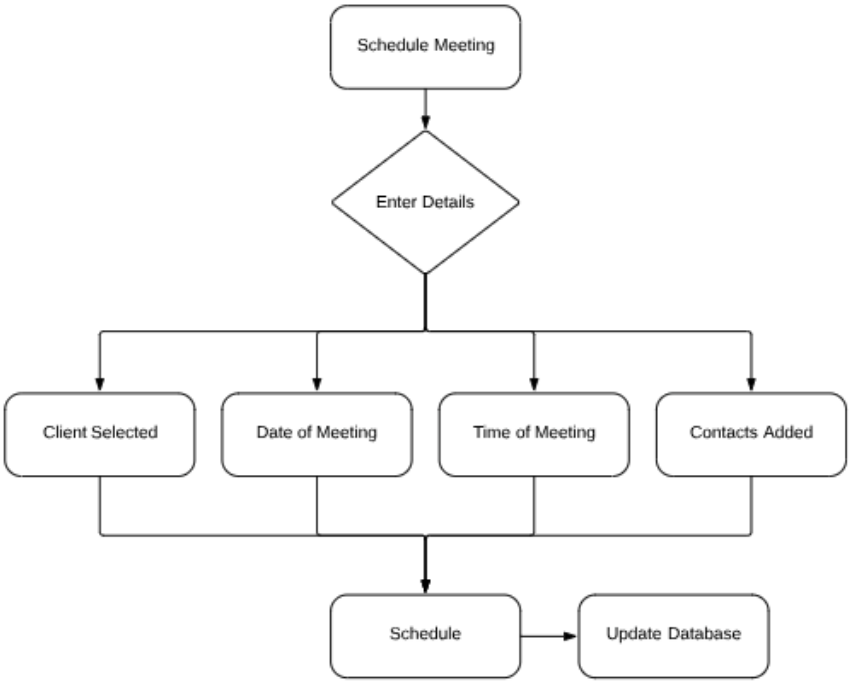
Displays information regarding the client. Here the user can look at previous meetings that have been held with a client, view recent news about the client such as their tweets and a link to their twitter page. It also

includes information about the client such as name and location. This can be used to update the user to any recent news about a client they are going to meet.



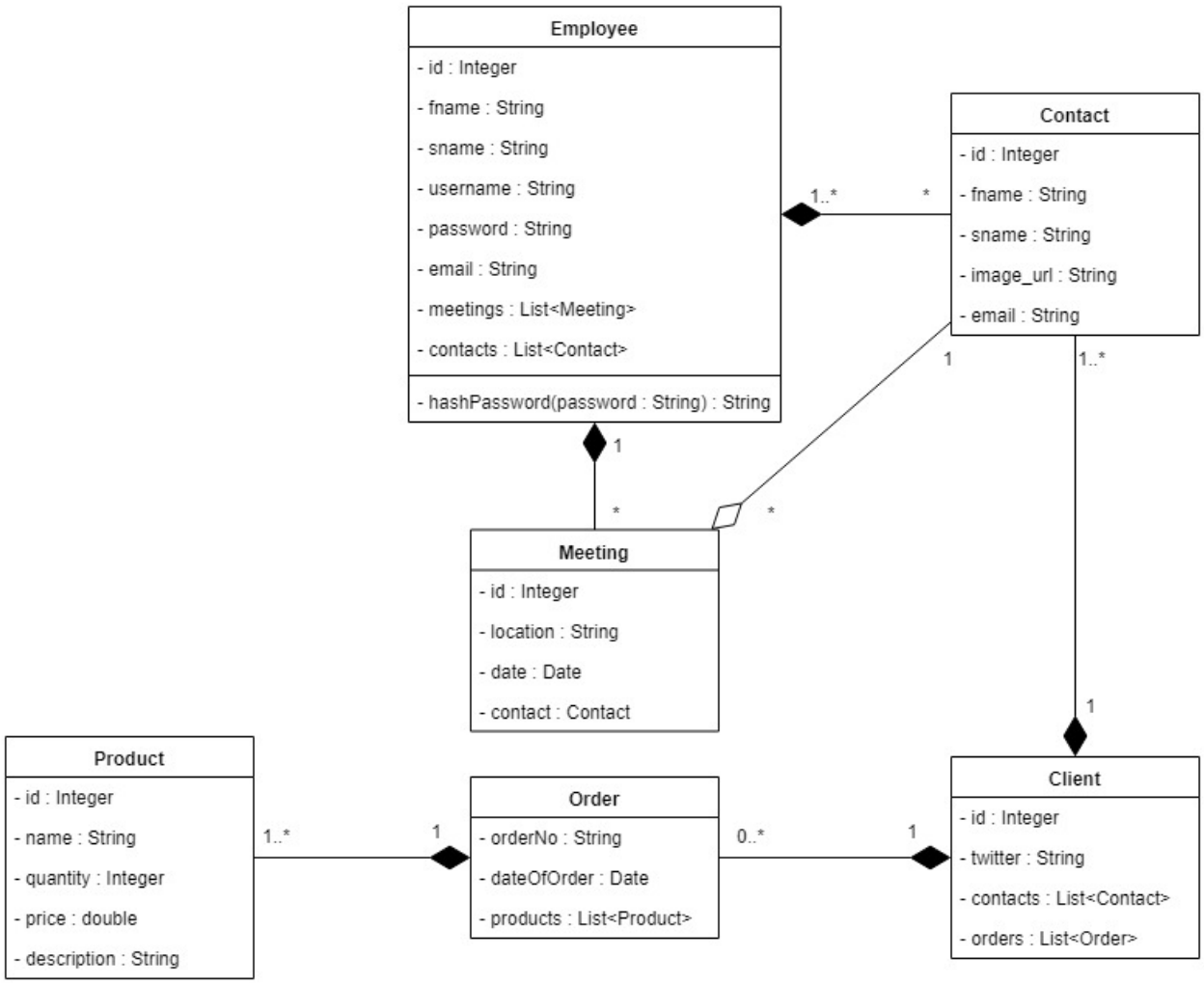
Component: Schedule Meeting

From this screen the user can schedule a meeting with a client. There are input fields that allow the user to select which client the meeting will be with. From here they are required to input information such as the date, location and time of the meeting. They can also add contacts depending on who will be attending the meeting from the client end.

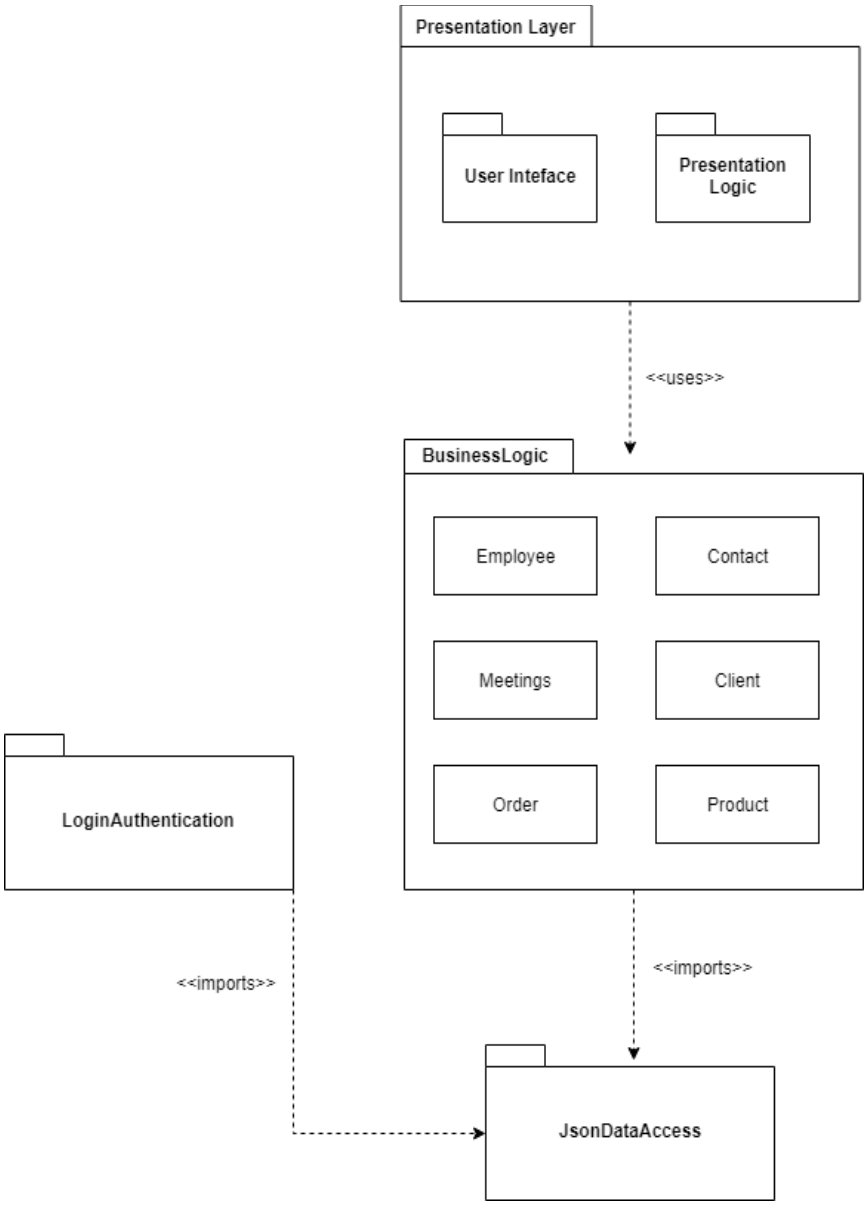


9.5 Package and Class diagrams which show dependencies between components

The class diagram shown below shows how the primary business logic classes are planned to interact with each other. The classes Employee, Contact, meeting, Client, Order and Product make up the business logic of the app, with other classes being used to display data (presentational classes) and classes to fetch data.


Employee
- id : Integer
- fname : String
- sname : String
- username : String
- password : String
- email : String
- meetings : List<Meeting>
- contacts : List<Contact>
- hashPassword(password : String) : String
Contact
- id : Integer
- fname : String
- sname : String
- image_url : String
- email : String
Meeting
- id : Integer
- location : String
- date : Date
- contact : Contact
Product
- id : Integer
- name : String
- quantity : Integer
- price : double
- description : String
Order
- orderNo : String
- dateOfOrder : Date
- products : List<Product>
Client
- id : Integer
- twitter : String
- contacts : List<Contact>
- orders : List<Order>
1..*
*
1
1..*
0..*

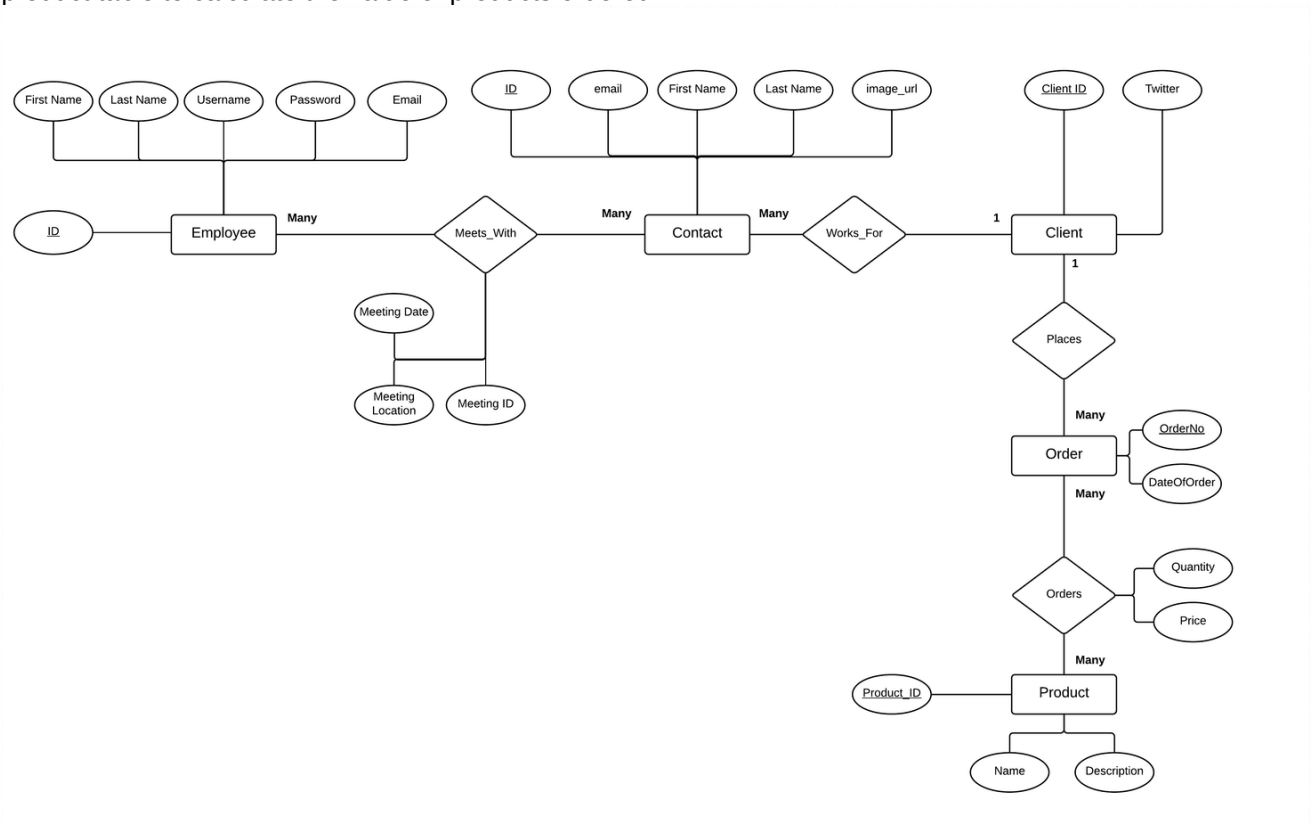
The package diagram below shows how the business logic classes fit into the overall application package design, the presentational layer relies on the business logic to display information to the user. The business logic classes rely on being able to acquire data from the remote database using the JsonDataAccess package.

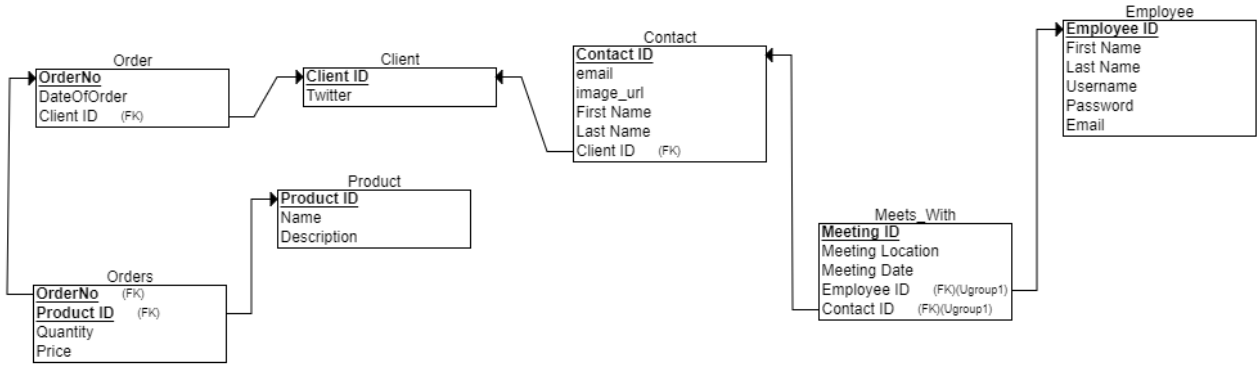


9.6 Show the dynamic behaviour of the system

The entity relationship diagrams below show how the key data objects: Employee, Contact, Client, Order and Product, relate to each other. One key part of the application is the ability for salespeople to create and update notes on the mobile application. To do this the change also has to be effectively replicated on the database.

The reason the system has been designed this way, is that the application focuses around upcoming meetings and allows salespeople to select which contacts at a client they will be meeting from a list, the salespeople must also be able to view sales information associated with that client hence the need for the orders and product table to calculate the value of products ordered.

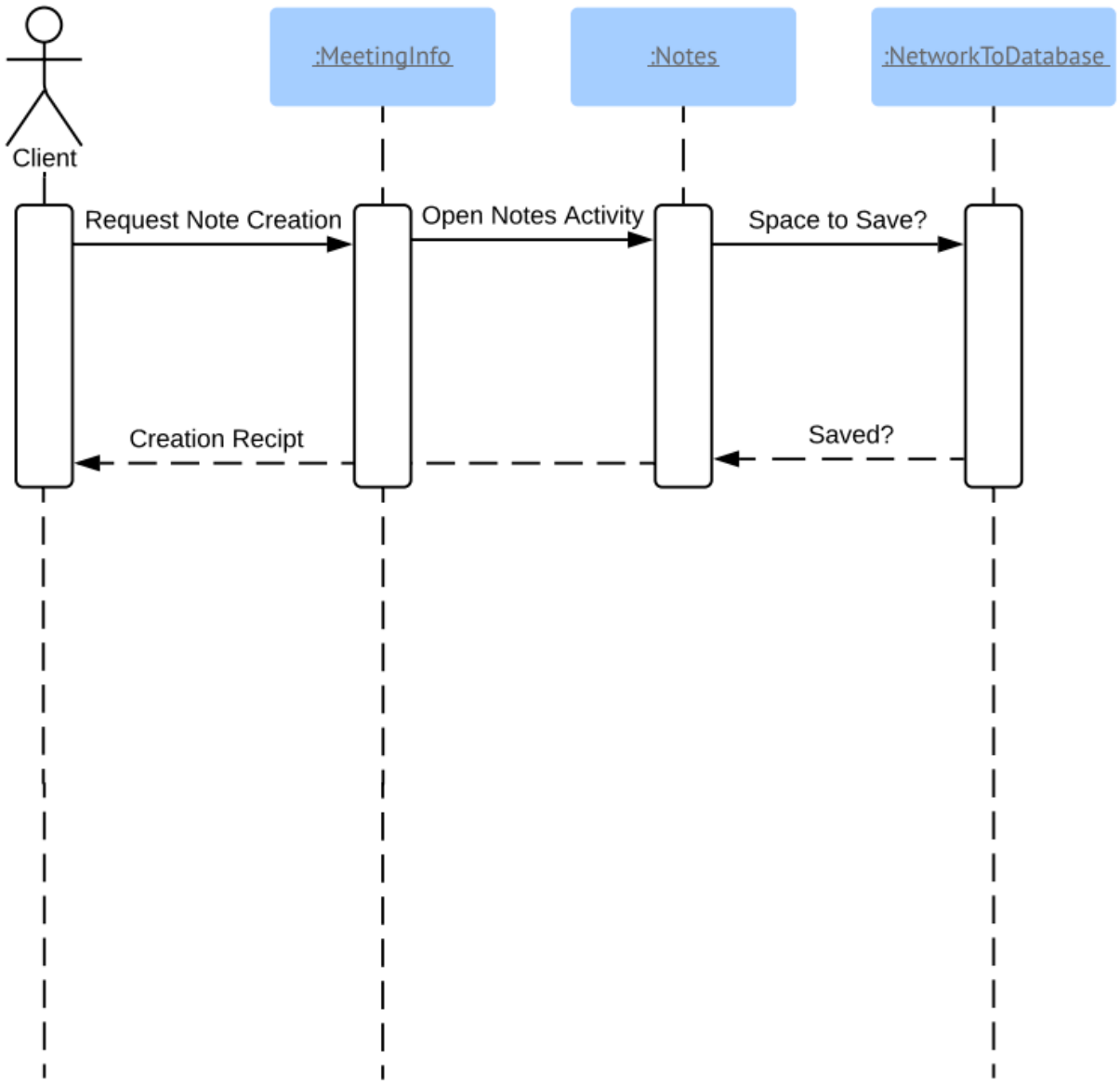




The sequence diagrams below show one of the key features of the application, the ability to create, update and store notes taken from a meeting. These diagrams represent the salesperson's interaction with the application.

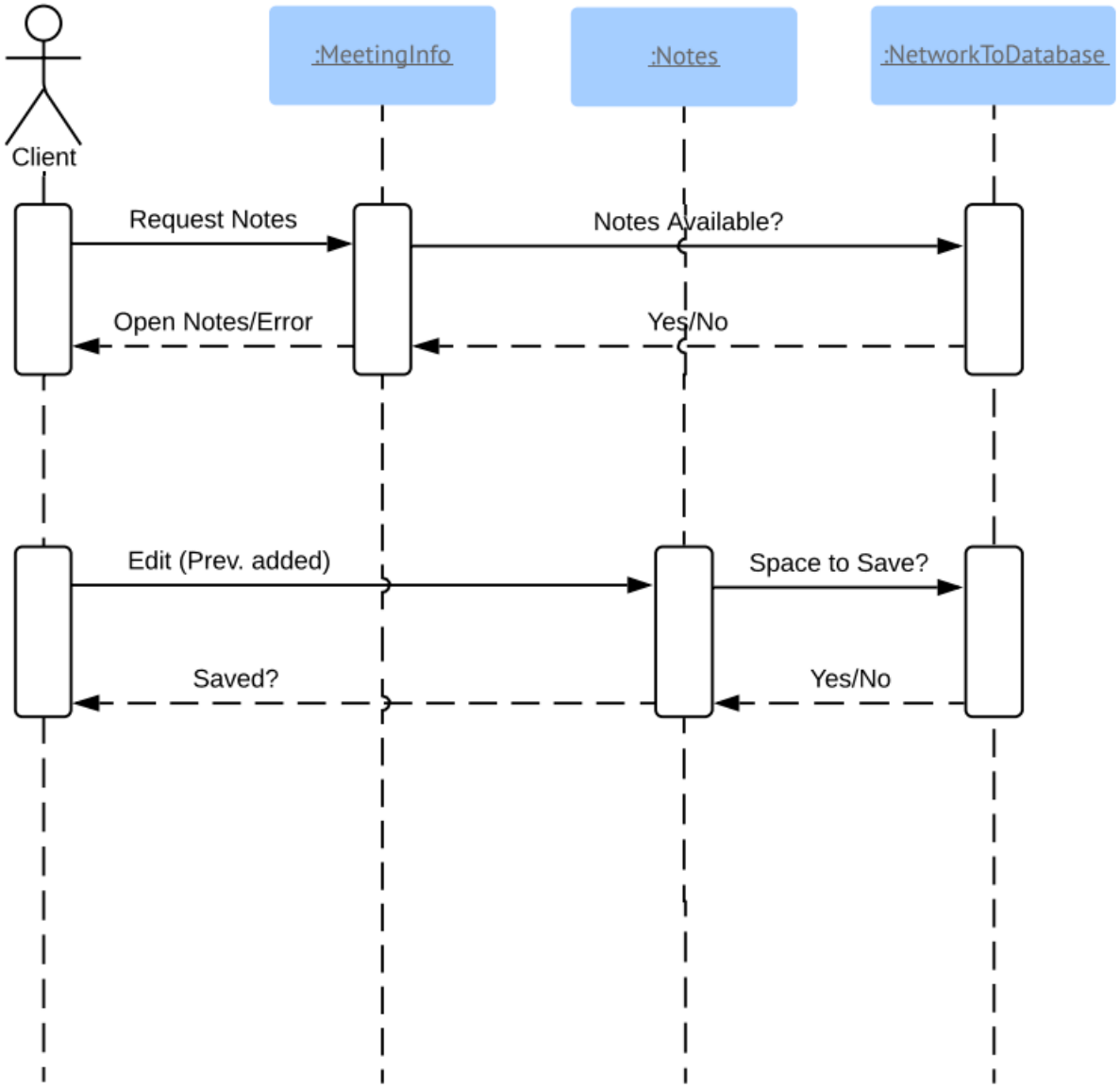
The first diagram shows creation of a new meeting note, so this would be the first time that a salesperson creates a note for a specific meeting, this is less complicated than modifying as it doesn't involve retrieving existing notes prior to making any changes. In this process the system will check that it is possible to save the notes to the remote database, otherwise feed back an error to the user.

When the meeting note gets successfully created it will send back a positive return signal (called a creation receipt) to let the application confirm to the user that the note has been successfully stored remotely.

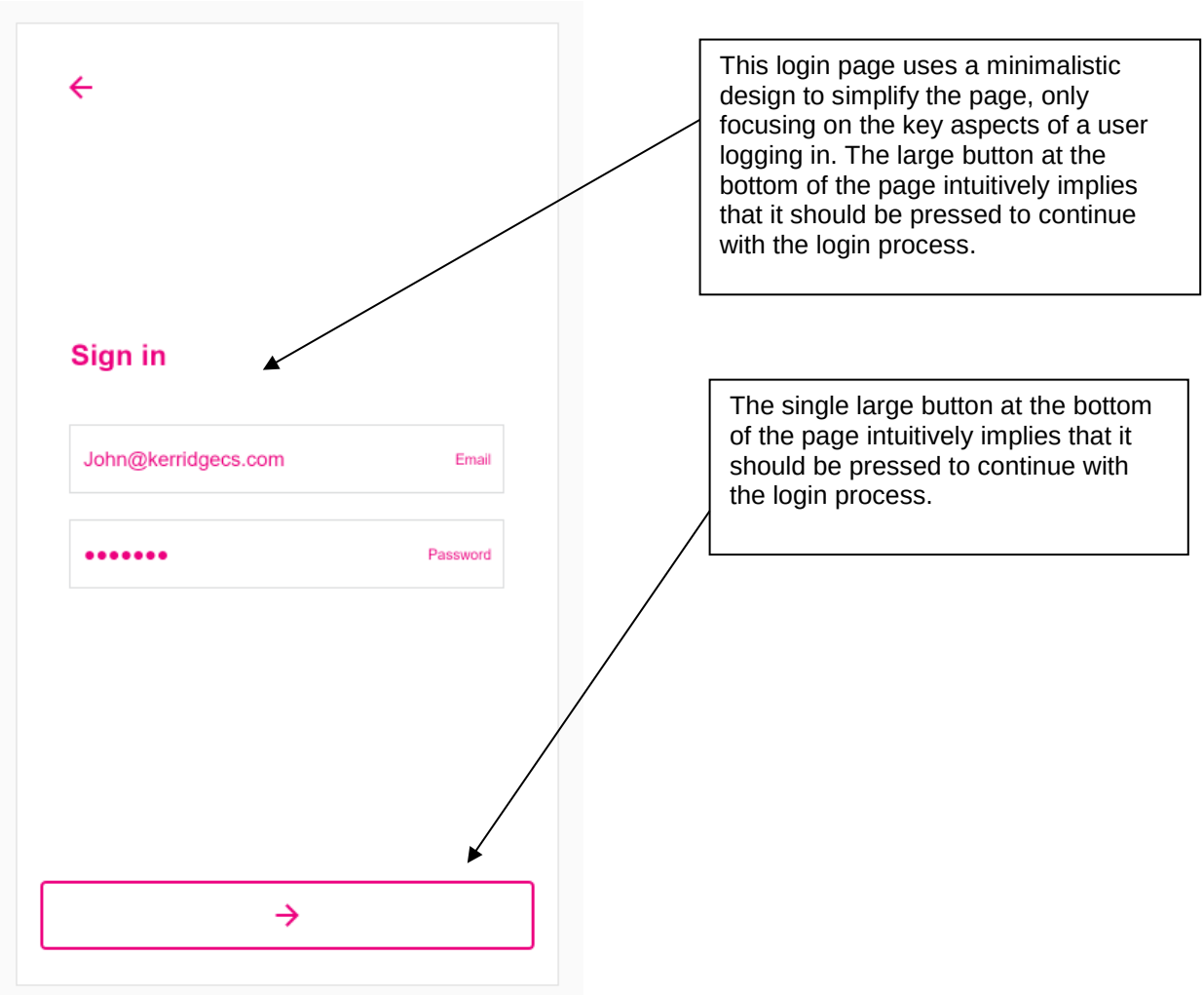


This sequence diagram shows the process of updating an existing note that has already been stored remotely created by the process shown previously. The client requests an existing note from the application which is then looked up remotely, if such a note exists then it is returned to the client to be displayed within the application.

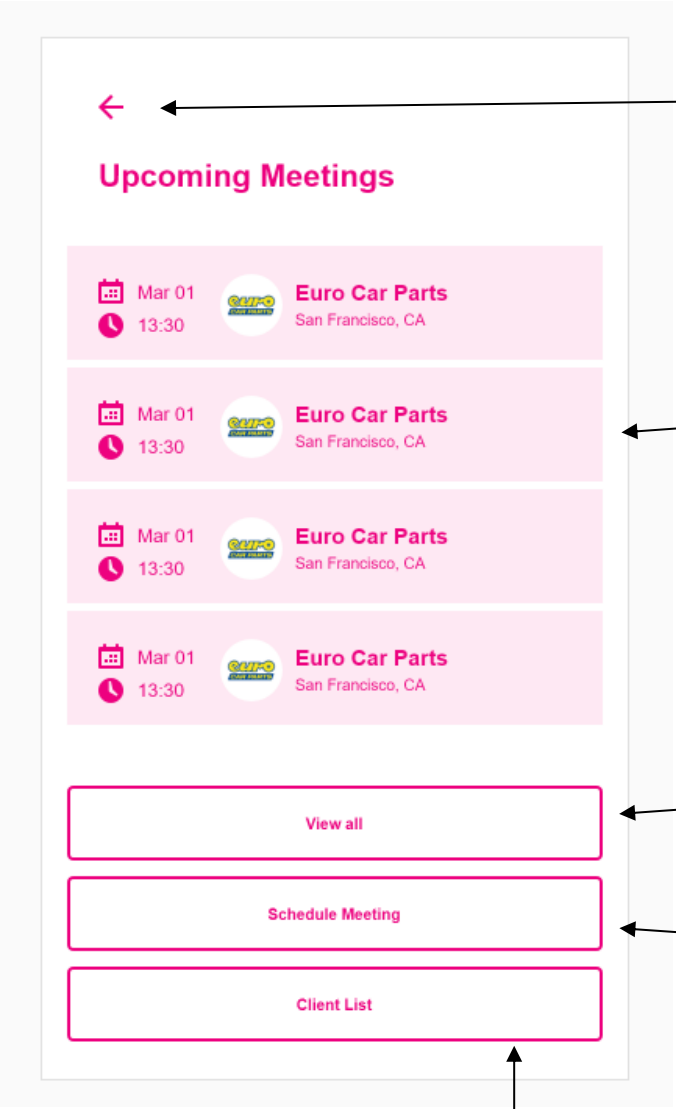
Then the user can elect to modify or expand on the note, this will trigger a similar process to the previous example whereby the note will be updated on the remote database, and if there is any issue with the process an error message will be returned to the user.



9.7 GUI, Human Interface Views – Describe how the user accesses functionality



After a successful login, the User will reach this homepage which displays the closest upcoming meetings:



The back button will allow the user to return to the login page in case there is a different account they wish to switch to, or simply to logout for security reasons.

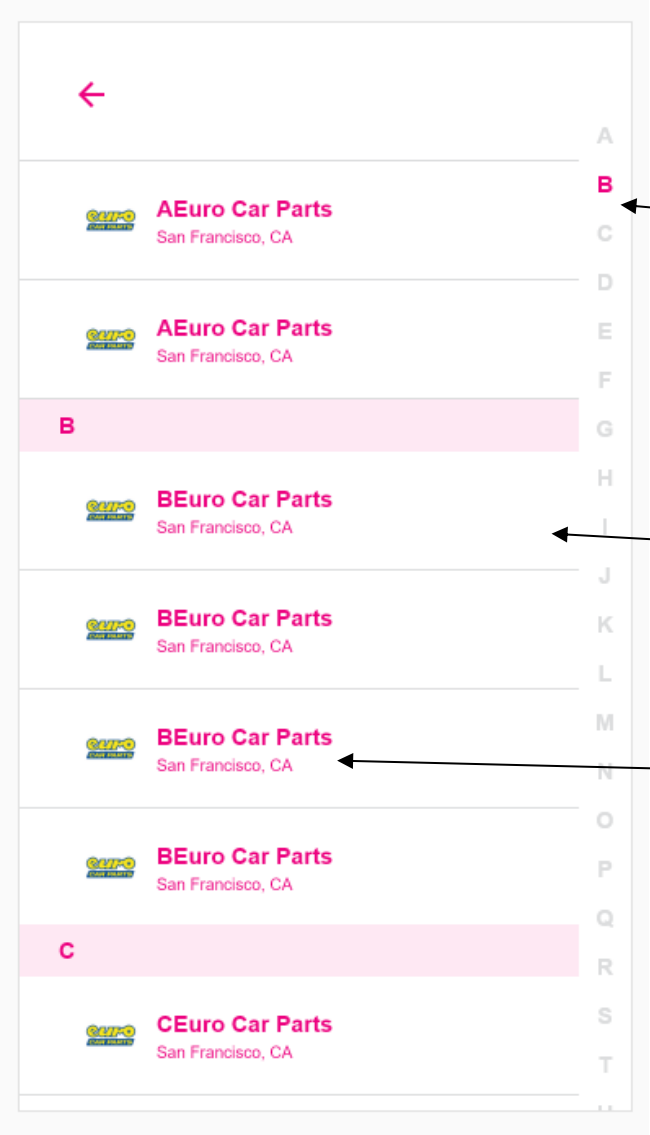
This central section displays the upcoming meetings, organised chronologically. Each meeting is spread out into a uniform grid, following material design aspects.

If the user wishes to look at meetings further into the future, or to view a specific date, this button can be pressed to take the User to the calendar page.

This button can be used to set up a meeting with any client, by specifying a time and date.

If the user wishes to browse through a list of all the clients, they may select this button to take them to a page like a contacts page on any smartphone.

This page displays the list of all clients available to the salesperson, and from this point can open a view of the client's profile page and can then view previous meetings with that client to look over any notes created within the meetings.

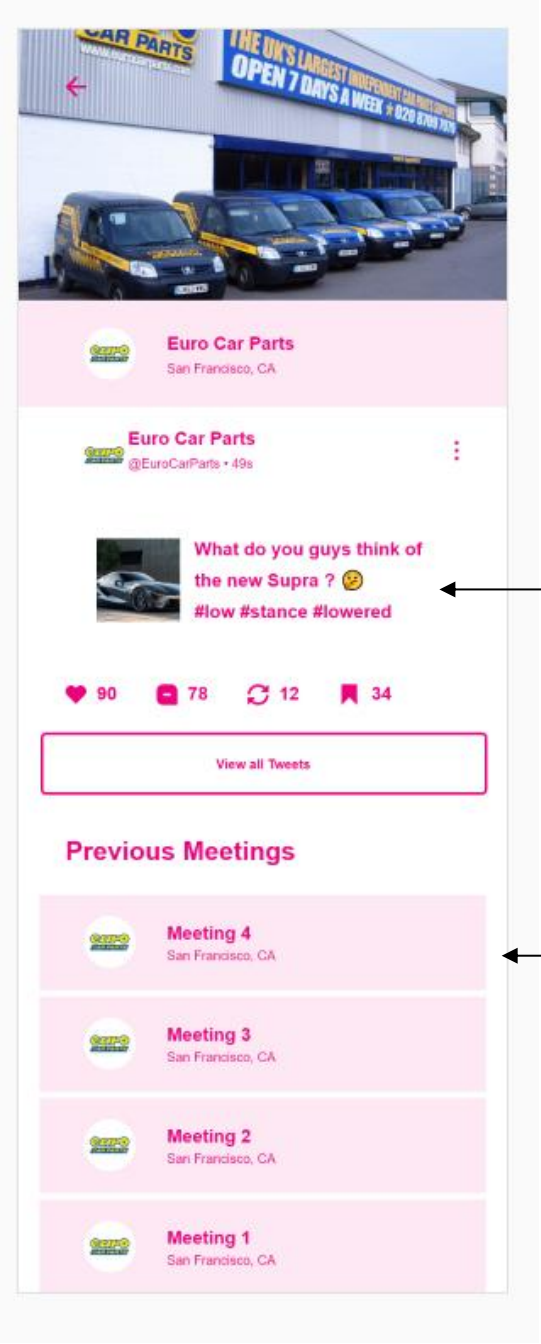


This alphabetical scrollbar can be used for quick navigation to certain positions in the sorted list of clients. Sliding across it will cause the scrollbar to move along with the user's input, taking inspiration from material design practices.

For smaller scale navigation through the list, the user can scroll on the list of clients directly, which will also react intuitively to the user sliding their finger up/down exactly like the scrollbar.

Selecting any specific client will take the user directly to their profile page.

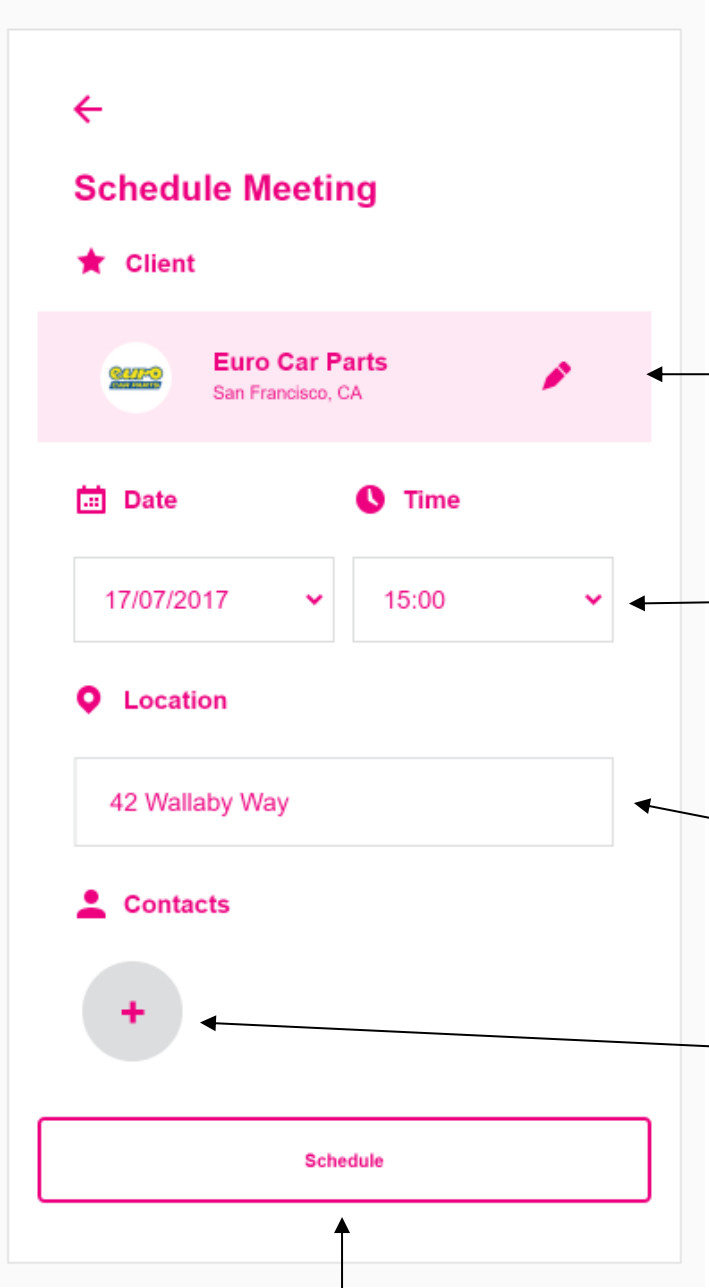
This page is where the client list will navigate to if the user has selected a specific client to view. Which allows the user to get an insight into their Client as well as view previous meetings.



Each client will have a snapshot of their twitter feed inserted into their profile page, which provides the user with a view of the company's current status, which could be useful during sales negotiations. This preview will always display the latest tweet produced by the company.

All previous meetings with this client will be viewable from this page, selecting any of these will take the user to the page for that meeting so they can view any specific details. This list is infinitely scrollable and follows the same material design spacing rules as previous lists within the app.

This page is for when the user wishes to create a meeting with a client, which will then be saved as an upcoming meeting in the user's calendar.



This section allows the user to select the Client from the list that was displayed on previous pages. This can be edited throughout any point of the meeting creation process. A simple pencil icon is used to denote a possible edit. This section is highlighted to emphasise importance.

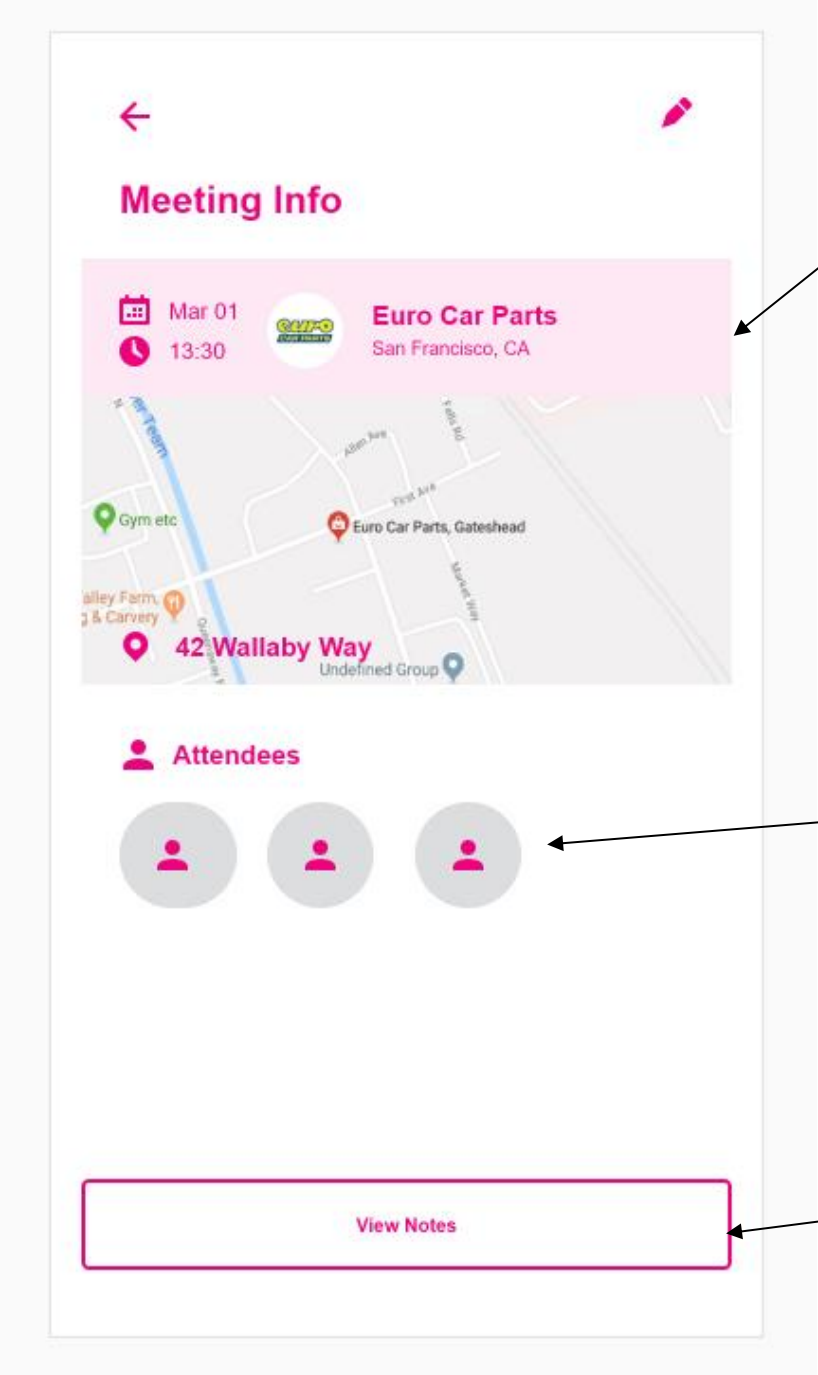
The user can then specify date and time of the meeting in two separate sections. Keeping time and date separate allows the user to change one without needing to reconfirm the other, which is much more intuitive for quick changes during creation.

The location of the meeting can also be edited, this location will default to the main location related to the client.

Adding additional contacts to the meeting can be performed by the user in the event of multiple people intending to be present to the meeting.

Once the user is finished with the creation of the meeting, they will need to press the large, minimalistic button at the bottom of the page to finalise this. This button is the largest on the page by a large margin which is intentional as it being the easiest button to see makes the process of scheduling a meeting intuitive.

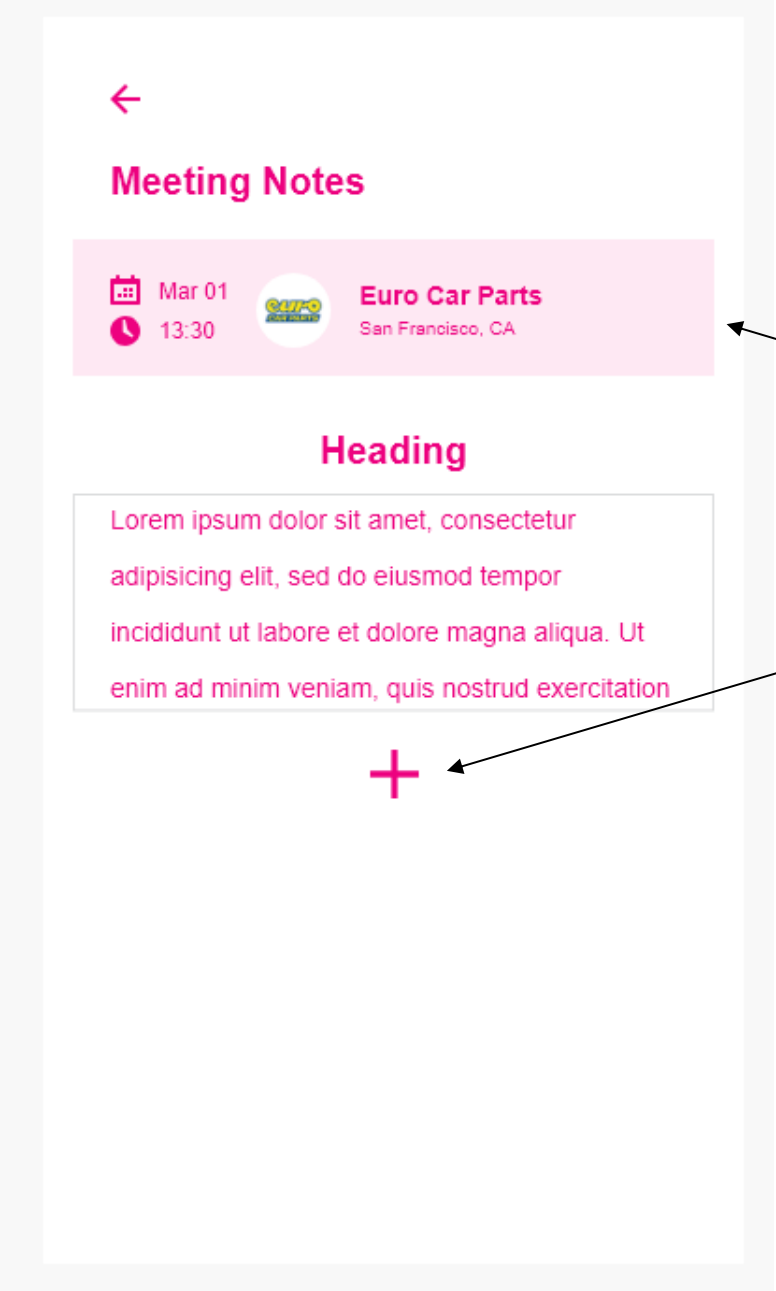
This section is what the user will see when they are currently within a meeting, or when they are viewing a previous meeting through the client's profile page.



Using a geolocation API, the location of the meeting will be displayed, as well as the client and date of the meeting as a heading above said location.

A list of attendees will be displayed in the bottom half of the page, these attendees will be the list of people added during the creation of the meeting.

The user can select the button at the bottom of the page to either edit notes in a current meeting, or view notes from a past meeting for reference, this button will open a new page dedicated to meeting notes.



If the user decides to enter notes, they will transition to this new page for creating meeting notes and revisiting previous notes.

This page displays the same meeting information as the previous page in a uniform format, to help the user keep track of the meeting details.

Articles of text for the meeting notes are added at the will of the user via a simple plus icon, with the application allowing the user to scroll through previous sections of text once enough notes have been written to necessitate this.

Users will additionally be able to view sales data relating to the client in each meeting, as well as being able to modify currently active order that the client requests during the meeting.

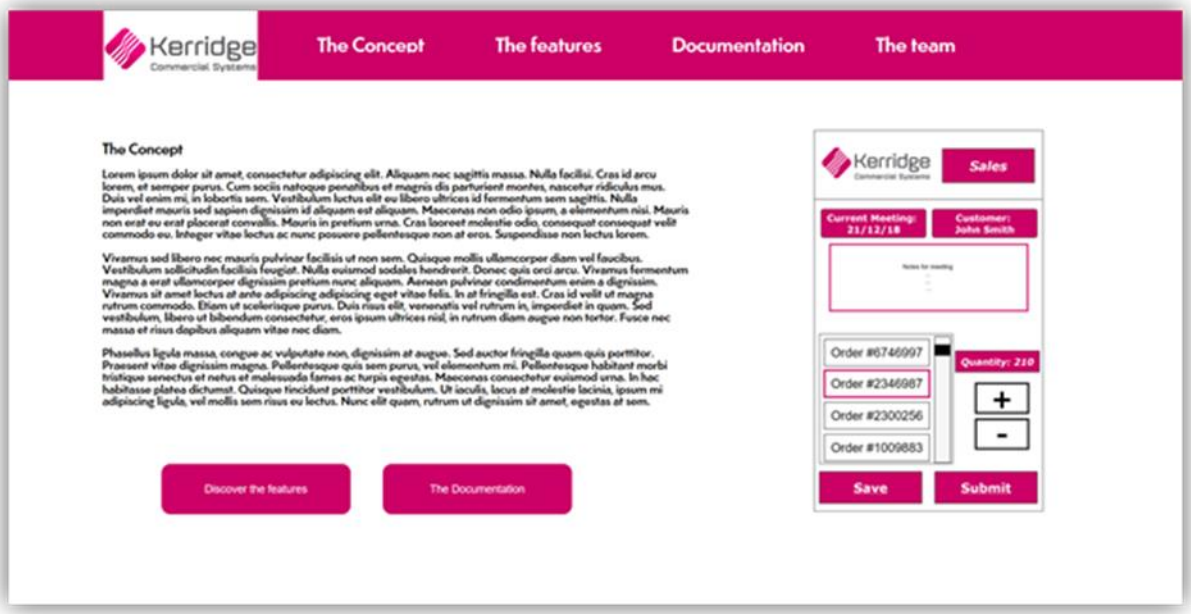
A graph displaying sales figures will be created based on orders from within the database, allowing the user to visualise a client's current financial investment in their products

Numerical figures for quantities of specific product types are displayed at the bottom of the page and will be scrollable in the event of many different products being ordered. Selecting a specific product will highlight the row.

Another large yet simple button is at the bottom of the page to simply the process for the user if they wish to modify current orders.

9.8 Website Design

As aforementioned, a website is to be built alongside the mobile application.



This is the first prototype of the marketing website design. It is being used to showcase the app and as a database for the documentation. The design retakes KCS' codes; as the colours for the navigation bar and the buttons.

The spacing has been taken into consideration ensuring that the content is evenly distributed across the pages. Each page includes relevant information and the design means that no overlapping of content can occur, and the page looks clean and is easy to read.

This is an advertising website using almost only front-end. It has been planned to include some animations or actions that play based on some user action.

Furthermore, the team has decided that a good idea is to include all of the user's notes from the application on the website so that they can be viewed when the salesperson is at a computer or another device. This provides an ease of access to the user so that they can copy their notes easily without having to do it through the application itself.

10 Test Plan

To be tested	Expected result	Actual result	Comments
Download the app.	The android app is downloaded and opens without error.		
Run the app.	The app opens, past to the splash screen and proceeds to the login screen without error.		

Correctness validation for email, the user enters an email address into the email field of the login screen.	After deselecting the email field (either by clicking the password field or elsewhere), the app should validate that the format of the email is correct, if incorrect display an error.		
Login process, the user provides correct email and password, and presses submit.	Credentials are validated against information stored in the database or local storage, if successful, then the user is taken to the home screen of the application. Otherwise, a relevant error is displayed to the user about what went wrong.		
The home screen is displayed after login, showing upcoming meetings and other options for the user.	A list of upcoming meetings should be displayed to the salesperson using the app; this list should be accurate to the upcoming meetings the salesperson has scheduled. Buttons to view all meetings, view all clients and schedule a new meeting should be visible		
Selecting a meeting.	The meeting information page relevant to the ID of the selected meeting should be displayed, by either fetching information from local storage or the remote database.		
Meeting information displayed is correct.	The meeting display card should show accurate information for the meeting date, time, company, location and company logo.		

Viewing all clients.	The user should be displayed a list of all the clients listed in an alphabetically ordered phonebook style format, the user should be able to select any of these clients to view the client information page. The list of clients will be fetched from either the remote database or the local storage.		
All clients are displayed in client list.	All client's companies in the system should be displayed to the salesperson		
Displaying individual client page.	After clicking a link to view an individual client page, the client page should be displayed with relevant information to the client selected. This will involve either displaying information from taken from the remote database or from local storage.		
Client information displayed is correct.	The client page should display accurate information for the company name, location, company logo, previous meetings and the twitter feed for the company selected. The information for this should be taken from the remote database or local storage.		
User can schedule meeting from within.	User should be able to schedule a meeting with a client company for a given time with a given company contact, this meeting should then be added to the list of meetings		

	associated with that salesperson.		
Company in schedule meeting is prefilled.	When the user selects to schedule a meeting from the client page, the client section of the meeting schedule should be prefilled based on the client page that the user scheduled the meeting from.		
User cannot schedule meeting for a date/time that is in the past.	The selection for dates that are in the past should be greyed out or not visible at all, the backend should also not allow for incorrect dates to be submitted.		
Correct contacts for the client are shown when arranging a meeting.	Contacts for a specific client are shown which the user of the application can select to add them to the meeting they are creating		
Meeting page, when user selects a specific meeting, the information about that meeting should be displayed.	The information relating the meeting should be displayed on a new page, including the date, time, client, contact, location, order graphics and the option for the client to make notes.		
Information on meeting page is accurate.	Information on the meeting page should be relevant to the meeting selected, all information on that page should relate to the client with no erroneous data. This information will be fetched from the remote database or local storage.		
Client order information page displays correctly.	Order information should be displayed to		

	the user, including the order number, product name, product images and product quantity.		
Calendar displays correct date.	When the calendar page is loaded to display all the meetings, the correct month should be displayed on the calendar to allow the salesperson to have a quick overview of upcoming meetings.		
Calendar highlights dates that contain meetings.	The calendar should highlight dates using colour to show which days have meetings on, salespeople should then be able to click on a highlighted date and a list of meetings on that date will be displayed.		
Edit meeting details.	Salesperson should be able to edit the details of a meeting; these updated details should be reflected in the data on the application and remote database.		
Correct contacts for client are shown on the meeting creation page.	The application should display the correct contacts for a specific client when scheduling a meeting.		
The location shown on the map for a meeting is correct.	The application should display the correct location (i.e. the country, city, street).		
Notes can be created for a meeting.	Salesperson should see the note has been created within their application. This means it should also have updated on the remote database.		

Existing notes can be updated by the salesperson.	The salesperson should be able to modify an existing note and see this change replicated on all places that the salesperson can view notes.		
API call for getting client list works correctly.	The call should return a JSON formatted list of all the clients that the salesperson has access to.		
API call for getting upcoming meetings works correctly.	The call should return a JSON formatted list of the upcoming meetings for a specific salesperson. This data should be correct checked against database to be correct. This call should return a limited number of upcoming meetings specified by the caller.		
API call to get meeting notes works correctly.	This call should return a JSON formatted response containing notes relating to a specific meeting ID.		
API call to create/update meeting note works correctly.	This API call should accept JSON encoded data from the application and update the database with the information provided following validation checks.		
Sales data is displayed correctly within the application graphs	The sales data page for a specific client should show accurate and up to date graphics representing the monthly orders by the client.		

Contribution Matrix for the Specification

Key
C = Create
M = Modify
R = Review

Section	Conor	Nathan	Ben	Jason	Matthew	Nick	Raven	Cédric	Ross
Background and Analysis – including Research Analysis, Analysis Process and Project Purpose	C, M, R	R	R	R	R	R	R	R	R
Roles and Deliverables – Including Team Roles and Responsibilities and Deliverables	M, R	R	M, R	R	R	R	R	C,M,R	R
Project Plan	R	M, R	R	R	R	C, M, R	R	R	R
Hardware and Software Resources	M, R	M, R	R	R	M, R	R	R	R	R
References	C, M, R	R	R	R	R	R	R	R	R
Definition of Terms	C, M, R	M, R	M, R	R	R	R	R	R	R
Solution Requirements – Including Functional and Non-Functional Requirements	M, R	M, R	R	C, M, R	R	R	R	R	R
Other Considerations – Including Assumptions, Constraints and Dependencies	M, R	R	R	R	C, M, R	R	R	R	C, R
Overview of Software	M, R	R	M, R	R	R	M, R	R	R	R
Modifications	M, R	R	R	R	R	M, R	R	R	R
System Architecture	R	R	C, M, R	R	R	R	R	R	R
Overview of Functionality	R	R	C, M, R	R	R	R	R	R	M
Package and Class Diagrams	R	R	M, R	C	R	R	R	R	R, M
Dynamic Behaviour	R	R	M, R		R	R	M	R	C, M, R
GUI and Human Interface Views	M, R	R	R	R	R	C, M, R	R	R	R
Website Design	M, R	R	R	R	M, R	R	R	C, M, R	R
Test Plan	R	R	C, M, R	R	R	R	R	R	R