

# **Development of an Application to Aid School**

## **Teachers in Planning an Off-Site Visit**

**Lisa Davie**

**September 2017**

**Dissertation submitted in partial fulfilment for the degree of  
Master of Science in Information Technology**

**Computing Science and Mathematics  
University of Stirling**

## **Abstract**

The aim of this project was to develop a mobile application which could be used by school teachers to simplify the process of planning an off-site school visit. Every time a teacher takes a class out of school grounds, whether for a visit to a local park or a residential trip overseas, there must be documentation in place to show that the risks associated with the trip have been properly evaluated and that reasonable control measures are in place to ensure the safety of the children. Furthermore, there is a growing body of evidence which suggests that opportunities to learn outside of the classroom have wide ranging benefits to children's education, health and well-being. As a result, outdoor learning is being actively encouraged in the curriculum in Scotland and the rest of the UK. Uncertainty around the guidelines for planning off-site visits and the paperwork involved have been identified as barriers to getting children outside for school lessons and this is the problem the project aims to tackle.

The main objective of the project was to build a solution that would make the process more accessible to teachers. The best practice guidelines for planning off-site visits are currently available in a 55-page document and it was hoped that by producing an app based on the document, the information could be made more approachable and interactive. In order to be accessible, the app needed to be available on both iOS and Android; it needed to use the best practice guidelines as its content and it needed to save the users time by helping them to complete the forms needed to plan a trip. It was decided to make local visits the focus for the application as these are the most common types of visits planned in schools.

Firstly, a cross-platform framework was selected for the development of the app, in order to satisfy the criteria that it should be developed for both iOS and Android. Secondly permission was gained from the Scottish Advisory Panel for Outdoor Education to use their materials in the app. Then the features of the app were developed incrementally under the guidance of an outdoor learning instructor with expertise in this field.

The result of the project is a cross-platform mobile application with features including step by step guides to take the user through the process of planning a trip; example forms to demonstrate how the paperwork should be completed and functionality to complete these forms within the app, then email them out to an address supplied by the user. The app is a self-contained application which works offline, with client-side form processing to produce the PDF documents. Feedback from teachers who have tested the app has been very positive and the Scottish Advisory Panel for Outdoor Education has shown an interest in taking the app on and are seeking funding to develop the project further.

## Attestation

I understand the nature of plagiarism, and I am aware of the University's policy on this.

I certify that this dissertation reports original work by me during my University project except for the following:

- The information provided in the app, and example documents were taken from the “Going Out There” website [1] and used with the permission of SAPOE. In addition, the forms generated by the app are adapted from the SAPOE generic forms.
- The jQuery Mobile JavaScript library, as discussed in section 5.5.1 was added to the project from the jQuery Mobile website [2].
- jsPDF and jsPDF AutoTable JavaScript libraries as discussed in section 5.5.4 are open source libraries included in the project from the GitHub website [3][4].
- Plugins added to the project from the Node Package Manager to implement specific functionality, as discussed in section 5.5 are as follows: Cordova-Plugin-camera [5], Cordova-Plugin-email [6] and Cordova-Open [7].
- The Cordova plugin APIs supplied by PhoneGap as discussed in section 5.5.6 were included by PhoneGap in the project files. Details of these can be found in the Cordova documentation [8].

Signature

Date

## **Acknowledgements**

The Author would like to thank the following people for their support and individual contributions to the project:

Dr Mario Kolberg for his expert guidance throughout this project.

Mike Harvey for proposing the project and all the help throughout.

The Scottish Advisory Panel for Outdoor Education for the use of their materials and their interest in the future of the project.

Kimberley Paton for providing a teacher's perspective.

My wonderful family, whose unwavering patience and support made this project possible.

# Table of Contents

Abstract .....	i
Attestation.....	ii
Acknowledgements .....	iii
Table of Contents.....	iv
List of Figures.....	vii
1 Introduction .....	1
1.1 Background and Context .....	1
1.2 Scope and Objectives.....	2
1.3 Key Achievements.....	2
1.4 Overview of Dissertation.....	3
2 State-of-The-Art .....	4
2.1 Existing Tools .....	4
2.1.1 Paper Checklists .....	4
2.1.2 Riskassessor.....	4
2.1.3 EVOLVE .....	7
2.1.4 Curriculum for Excellence App.....	9
2.1.5 Summary of Existing Tools .....	10
2.2 App Development and Cross Platform Frameworks .....	11
2.2.1 Microsoft Xamarin .....	13
2.2.2 PhoneGap .....	14
2.2.3 Titanium.....	16
2.2.4 Summary of Cross Platform Frameworks .....	17
2.3 The PhoneGap Toolkit .....	18
2.3.1 Selection of PhoneGap Build and Command Line Interface.....	19
2.4 The Agile Approach.....	20
2.4.1 Incremental development .....	20
2.4.2 Stakeholder Engagement .....	21
3 Requirements Engineering .....	22
3.1 User Requirements and Use Cases .....	22
3.2 Non-Functional Requirements.....	25
3.3 Analysis of the Problem .....	25
3.4 Acceptance criteria .....	27
4 Designing the Application .....	28
4.1 Content.....	28

4.2	Initial Designs.....	33
4.3	Further features.....	39
4.3.1	Generating Further Form Types.....	39
4.3.2	Acknowledging SAPOE.....	39
4.3.3	User Settings.....	40
4.3.4	Saved Documents .....	40
	Designing for PhoneGap .....	40
5	Implementation.....	42
5.1	Android vs iPhone Development.....	42
5.2	Application Architecture.....	43
5.3	Walk Through .....	44
5.3.1	Home Screen and Step by Step Guides .....	44
5.3.2	User Settings and About Screen .....	45
5.3.3	Completing a Risk Assessment.....	46
5.3.4	Completing a Visit Plan.....	48
5.4	Implementing Key Functionality.....	50
5.4.1	jQuery Mobile .....	51
5.4.2	Access to native E-mail functionality.....	52
5.4.3	Viewing PDFs.....	53
5.4.4	Generating PDFs .....	54
5.4.5	Access to the Device's camera .....	56
5.4.6	Standard Cordova Plugins .....	56
5.5	Visual Appearance .....	57
5.5.1	Icon and splash screen.....	58
5.6	Unsolved Problems.....	59
5.7	Internal Structure .....	59
6	Testing and Feedback .....	62
6.1	End User Feedback during Development .....	62
6.2	Functional Testing Against a set of Common Scenarios .....	64
6.3	User Feedback on the Final Product.....	65
6.4	Feedback from SAPOE .....	66
6.5	Feedback from the Client.....	67
6.6	Summary of testing.....	68
7	Conclusion.....	69
7.1	Evaluation of the App.....	69
7.2	Evaluation of PhoneGap.....	70
7.3	Evaluation of the Agile Approach.....	71

7.4 Further Work.....	72
7.5 Summary.....	73
8 References .....	74

## List of Figures

Figure 1.	Screenshots from Riskassessor .....	5
Figure 2.	Screenshot a PDF generated in Riskassessor.....	6
Figure 3.	Screenshot of a customised EVOLVE log in page .....	8
Figure 4.	Screenshots from the Curriculum for Excellence app .....	9
Figure 5.	Table comparing the tools available in the field .....	10
Figure 6.	Table comparing approaches to mobile app development .....	11
Figure 7.	Visual Studio IDE with sample Xamarin app .....	13
Figure 8.	Sample PhoneGap application .....	14
Figure 9.	Appcelerator IDE with sample Titanium app .....	16
Figure 10.	Summary of products available from PhoneGap .....	19
Figure 11.	Diagram illustrating the Agile Method .....	21
Figure 12.	Use Case 1: Initial Requirements.....	23
Figure 13.	Use Case 2: Additional Requirements .....	24
Figure 14.	Use Case 3: Final Requirements .....	24
Figure 15.	Analysis of the Requirements .....	25
Figure 16.	Flow Chart for planning a Day Visit.....	29
Figure 17.	Worked example of a Routine and Expected Risk Assessment .....	30
Figure 18.	Generic Visit Plan for “Routine Visits” .....	31
Figure 19.	Mapping SAPOE tools to features of the app.....	32
Figure 20.	Initial sketch of the app.....	33
Figure 21.	A Mock-Up User Interface .....	34
Figure 22.	Risk Assessment Design: Overview .....	35
Figure 23.	Risk Assessment Design: User Interface .....	36
Figure 24.	Risk Assessment Design: User Interface continued.....	37
Figure 25.	Risk Assessment Design: Handling the input .....	38
Figure 26.	Diagram illustrating how the tools relate to design .....	41
Figure 27.	Diagram explaining the iOS provisioning profile.....	42
Figure 28.	Diagram of the Application Architecture .....	43
Figure 29.	Example Risk Assessment generated by the App .....	47
Figure 30.	Routine Visit Plan generated by the app.....	49
Figure 31.	Day Visit Plan generated by the App .....	50
Figure 32.	Diagram illustrating how jQuery mobile was applied .....	52
Figure 33.	The Home Page Banner and App Icon.....	58
Figure 34.	Overview of the Application Structure .....	60



Figure 35. Project Files .....61  
Figure 36. Initial User Feedback.....63  
Figure 37. Example Scenario for Testing.....64

# 1 Introduction

This chapter gives a brief overview of the project, providing context, the scope of the project and the key achievements.

## 1.1 Background and Context

The idea for this project was proposed by an Outdoor Learning Instructor for West Lothian Council. Whilst working on a project to raise attainment of primary school children through outdoor learning, he found that the main barrier to getting children outside for school activities is uncertainty around the complex set of guidelines given to teachers regarding safe practice for off-site visits. Based on his knowledge of the guidelines, and understanding in this area, he proposed an app which would provide the relevant information in a simplified format given the requirements input by the user.

There is a substantial and growing body of evidence supporting the positive impact of outdoor learning on children[9][10][11]. These benefits include improvements to social development, health and educational attainment and in recent years projects such as the Natural Connections Project in England[9] or the Outdoor Connections programme in Scotland have been set up to look at these benefits and more importantly how to improve access to outdoor learning for all children. The benefits of outdoor learning have been recognised by the government in Scotland and in 2010 they published a report entitled Curriculum for Excellence through Outdoor Learning as guidance for schools regarding the benefits of outdoor learning and how to embed it in the curriculum[12]. It should be noted that “outdoor learning” encompasses a range of activities from residential trips, adventure activities and fieldwork to simply taking lessons out to the local environment such as woodland, parks and even the local urban environment. Attainment in subjects across the curriculum has been shown to benefit from being taught outdoors, if properly planned and implemented (Nicol et al, 2007 [11]).

While the benefits of Outdoor Learning are widely accepted, ensuring the safety of children in the process is paramount. In 1993 a canoeing tragedy resulted in the death of four teenagers on a school trip to Lyme Bay in England. Following this event, legislation came into force regulating all adventure activity centres working with children in the UK and groups were set up to oversee these kinds of activities. One side-effect of the increased regulations and the reaction to the tragedy was that some schools became averse to taking children on any kind of trip and so barriers to outdoor learning were formed (Grant, 2013 [13]). It is in this climate that the Scottish Advisory Panel for Outdoor Education (SAPOE) operates. The panel consists of representatives from local authorities throughout Scotland and aims to support the use of

outdoor learning as part of the curriculum in Scotland and to promote good practice in the planning and implementation of offsite visits[14]. They also oversee “Going Out There” the Scottish Government’s framework for safe practice in offsite visits. It is this material which this project seeks to clarify, by making it available to teachers in the form of an app which will take them through the process of planning an off-site visit.

## **1.2 Scope and Objectives**

The primary objectives of this project were as follows:

- To develop an app for teachers to simplify the process of planning a local off-site class visit.
- Use the guidelines available from the Scottish Advisory Panel for Outdoor Education in the “Going Out There” document.
- Develop the app for Android and iOS.
- Include functionality that will generate the completed documentation.
- Produce an app which is user-friendly and intuitive to use.
- Keep the app self-contained, so that it works offline.

Not included in the scope of the project were:

- Considerations about other types of visit, such as Residential or Adventure activities.
- Changing or amending the guidance provided.
- Details of specific establishment or local authority guidelines.
- Functionality relating to the visit approval process or any handling of forms after they have been completed by the app.

## **1.3 Key Achievements**

The app delivered at the end of the project exceeded the client’s expectations and has been demonstrated to a representative of the Scottish Advisory Panel for Outdoor Education, with a view to taking the project further. It has been successfully tested with teachers on both Android devices and on an iPhone, to very positive feedback. Implementation of client-side form processing and successfully accessing many of the native features of the device are among the top technical achievements of the project. In addition, the app succeeded in being user friendly and intuitive to use.

Furthermore, the author of this project has succeeded in developing technical and professional skills which will form the basis of a career in IT. These include becoming proficient in JavaScript, HTML and CSS, using a Command Line Interface, gaining an understanding of mobile devices running Android and iOS, the use of APIs in the form of PhoneGap plugins and experience of the software lifecycle and developing a product for a Client.

## **1.4 Overview of Dissertation**

**Chapter 1** provides an overview of the project, the context and motivations as well as the objectives and achievements of the project.

**Chapter 2** looks at existing solutions to similar problems and examines possible approaches to the project by comparing cross-platform frameworks and considering the Agile method.

**Chapter 3** explains how the requirements for the project were gathered and analysed through discussions with the client, leading to a set of Use Cases and Acceptance Criteria.

**Chapter 4** covers the design of the application based on the requirements set out in Chapter 3, examining design sketches that were made during the design process and consideration of other design decisions that were made.

**Chapter 5** looks at the implementation of the project with details of the architecture of the final product, a walk-through of its features and explanation of how this was achieved.

**Chapter 6** covers the testing that the application was put through, both functional and with end users and examines the feedback received from all relevant parties.

**Chapter 7** provides an evaluation of the product, recommendations for further work and a summary of the project.

**Chapter 8** is the references section.

## 2 State-of-The-Art

This chapter examines the tools already available in this field, to ensure that the project is adding something useful and to inform design decisions. It also aims to give context to the project by considering the technologies available and established approaches to solving the problem.

### 2.1 Existing Tools

This project aims to develop an app to help teachers manage risks associated with school visits. As this is a limited field, we will consider both existing tools for *general* risk assessment and also other types of app used in schools which serve a similar purpose to this one.

Existing risk assessment tools fall into two categories: simple, paper based checklists including those available from the Health and Safety Executive (HSE) and SAPOE and sophisticated software packages for the management and tracking of risk related issues such as Riskassessor and EVOLVE. A thorough understanding of the strengths and limitations of these will provide insights for this project.

#### 2.1.1 Paper Checklists

Paper based checklists are the main tool available from the advisory bodies for helping individuals identify and manage risks. For example, the HSE provides a checklist for classroom safety, which can be used identify risks in the classroom[15]. And similar checklists are available for a variety of different work place environments. Specifically, for off-site visits, SAPOE provides a toolkit on their website to supplement the guidance document. The tools are in the form of example assessments, generic forms and flow charts to outline the steps involved in planning a visit. In addition, some local authorities also include their own versions of flow charts and checklists within their policy documentation (for example: “Mid Lothian Council Guidelines for Educational Visits”[16]).

The main benefit of these paper checklists and flow charts are that they are simple to use, cost-effective and freely available. They may also help to clarify the guidance and save the user time compared to reading the entire guidance document. On the other hand, they are fairly generic in nature and not particularly interactive. According to the client, the uptake of the SAPOE toolkit is low, which he speculated may be due to a lack of awareness.

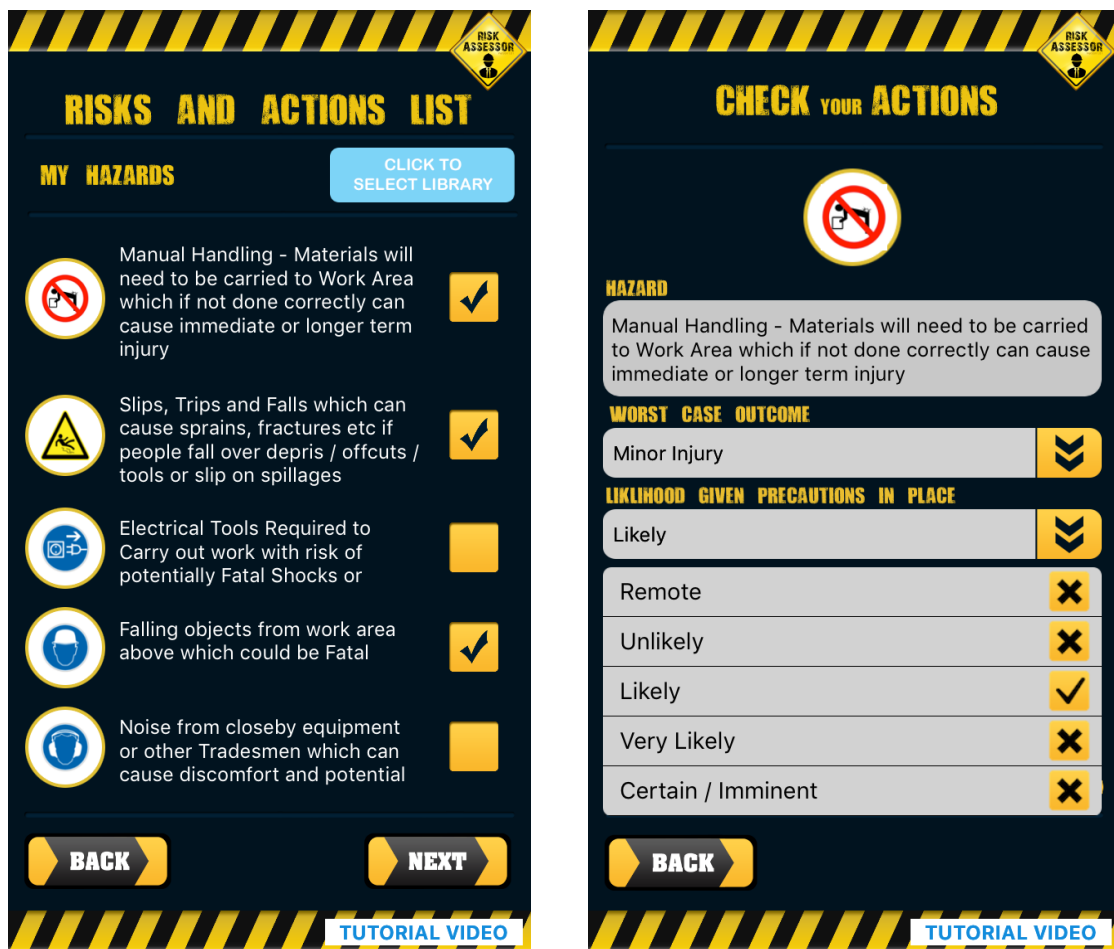
#### 2.1.2 Riskassessor

One app aimed specifically at managing health and safety in the work place is “Riskassessor”, a comprehensive risk assessment tool with cloud based storage of the resulting data. This is

available in the form of an online service and an app available in two formats - a paid version and a scaled down demo (Risk Assessor Lite). The Lite version does not have access to all the features of the full version, however it shows the basic functionality without charge, therefore it was downloaded and tested for the purposes of this project.

On installation, the app prompts the user to complete a setup step where they input their company details, the company logo and other settings such as the risk assessor's name, job title and digital signature. These details are used later to customise forms generated by the app. Once setup is complete, there are a range of options such as completing a risk assessment, producing a method statement and performing audits

**Figure 1. Screenshots from Riskassessor**



The risk assessment option takes the user through the steps of conducting the assessment, by prompting them to select options and enter details relating to the project. Once all selections are complete the app generates a PDF of the assessment.

**Figure 2. Screenshot of a PDF generated in Riskassessor**

Hazards Identified	Grid Ref	Controls in Place	Based on Existing Controls				Additional Control Measures Required	Residual Risk		
			Worst Case	Likelihood	Score	Rating		Likelihood	Score	Rating
Manual Handling - Materials will need to be carried to Work Area which if not done correctly can cause immediate or longer term injury		Use safety checked Trolley / Sack Truck	Minor Injury	Likely	6	Low				
Slips, Trips and Falls which can cause sprains, fractures etc if people fall over debris / offcuts / tools or slip on spillages		Area to be tidied, cleaned and dry prior to work date	Minor Injury	Likely	6	Low				

**Is a follow up assessment required? NO**

**Worst Case Outcome**

1 = No injury  
2 = Minor injury  
3 = Lost time injury  
4 = Severe injury  
5 = Fatality

**Likelihood**

1 = Remote  
2 = Unlikely  
3 = Likely  
4 = Very likely  
5 = Certain / Imminent

**Score Guide**

16 - 25 = High Risk  
9 - 15 = Medium Risk  
1-8 = Low Risk


It's important you understand this report. If you don't you should seek proper Risk Assessment Training.

Completed by: Me  
Position: Boss  
Signature:   
Date of Assessment: 30/08/17  
Page No: 1

Follow us:

[RiskAssessor](#)  
[RiskassessorApp](#)  
[www.riskassessor.net](http://www.riskassessor.net)

CREATED USING



[www.riskassessor.net](http://www.riskassessor.net)

BACK

ADD ANOTHER

EMAIL

SAVE

[TUTORIAL VIDEO](#)

In the full version of the app the user can save the PDF to the device, to the cloud or send it by email, however only the option to save to device is available in the Lite version. Once saved, the assessments can be “edited” by taking the user back through the questions and allowing them to change their selections before generating a new PDF.

User comments state that the app is simple and easy to use, good for sharing, managing and tracking assessments and many users said that it saved them time as it can be used by anyone and not just those trained in Health and Safety. However, users did not like the fact that you have to pay for additional features such as emailing the documents, in fact complaints around the costs associated with the app were the most common grievance. Some users reported technical issues, particularly around the app crashing; others did not like the fact that the Riskassessor logo is added to every document and that links to twitter and Facebook appear on several pages of the app; another user said they would like more guidance notes on the hazards, risks and control measures.

In the opinion of the author, the general look and feel of the app was professional, the options for the risk assessment were comprehensive and there was a library facility to add more custom options. The app effectively handled considerations such as adding a company logo (by uploading a file or accessing the camera on the device) and adding the assessor’s signature (by opening a window on which the user could “write” the signature on the screen). It also handles problematic issues, such as the layout of the information in the PDF when the assessment goes over more than one page (by effectively having a header and footer on each page).

The biggest weakness of the app was that there appear to be some bugs in the application, the Tutorial button takes the user to a video explaining how to use the app, but this caused the app to hang and then it was not possible to get back to the main application. The app was also difficult to navigate, once the user begins a risk assessment it is difficult to exit without going all the way to the end or back to the start and it is mandatory for the user to complete some parts of the assessment (such as project details) and it is not possible to leave these items blank if they are not applicable.

Overall, the risk assessment produced looked professional and this app would be well suited to a small or medium sized business that has a high requirement for health and safety measures but does not wish to invest in a bespoke risk assessment system. The app appears to have successfully simplified the process of carrying out a risk assessment and in so doing, made it more accessible to staff by allowing non-specialists to carry out these tasks. The cloud and email features allow the information to be shared, tracked and managed effectively. This app provides a good example of how the process of completing a risk assessment can be simplified and still customised to an individual organisation's needs. However, given that it is a professional application with associated costs, it was disappointing to find the technical issues had not been addressed.

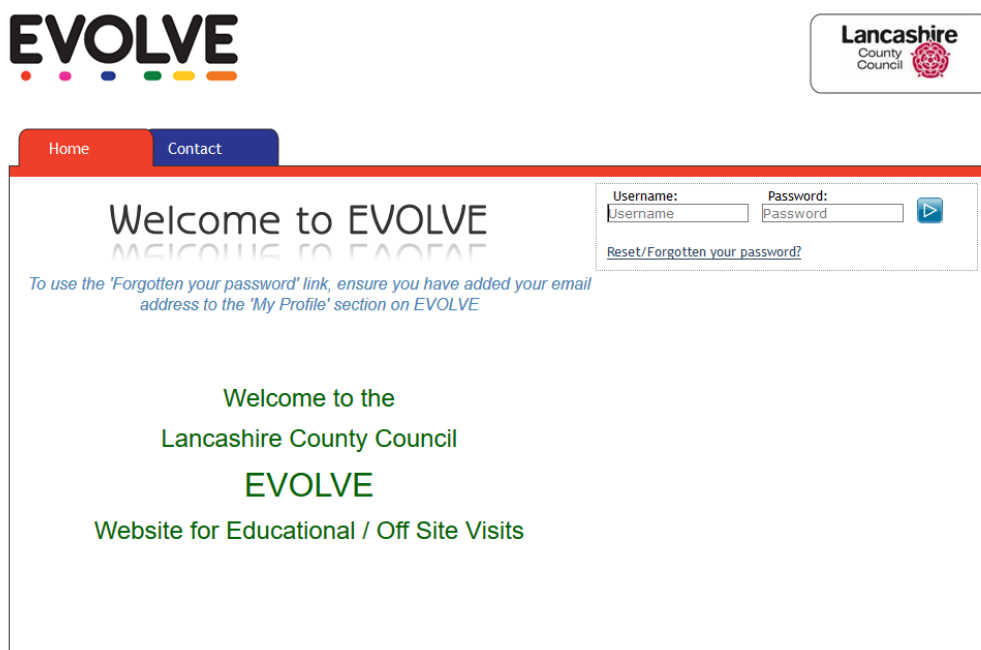
### **2.1.3 EVOLVE**

Probably the most relevant software to this project is a product called EVOLVE (Educational Visits Online Virtual Environment) owned by EduFOCUS. According to their website [17] EVOLVE is a tool for planning and managing school visits and clubs and offers a whole suite of services to manage educational trips end-to-end. All the services are paperless and available online, from the completion, submission and storage of forms, to communication with parents and generation of reports at the end of the year. As the software is aimed at schools and not the general public, it has not been possible to view it first hand or to find any user reviews for it. However, the following information was obtained through the company's promotional literature [18]:

- It is a standalone web based product and does not require software to be installed on the school's computers.
- It is completely customisable to local authority or establishment guidelines and it can integrate with local systems (school or local authority) for visit approvals.
- It includes facilities for searching, viewing, managing and analysing the data.



**Figure 3. Screenshot of a customised EVOLVE log in page**



(Source: Lancashire County Council's EVOLVE log in page [19])

It should be remembered that the information gathered is mostly promotional material and therefore an accurate and unbiased picture may not be possible. However, based on what is available, EVOLVE appears to provide everything a school needs to manage trips and remove much of the uncertainty from planning an off-site visit, by both standardising the process and offering a service tailored to a given establishment. If software such as this were used by every local authority, it would be unlikely that this project would be able to add anything new.

One obvious drawback of the system is the charges associated with it (a set-up fee and annual subscription, although no exact prices are quoted on the website), which may be a barrier for some councils taking it up. As the responsibility for the way trips are managed lies with local authorities, the service is only available in those areas where the local authority has implemented EVOLVE as their system for handling off-site visits, or to independent schools who are not under local authority control. According to a government website [20], there are a total of 32 local authorities in Scotland, of which only 11 are listed on the eduFOCUS website as EVOLVE customers.

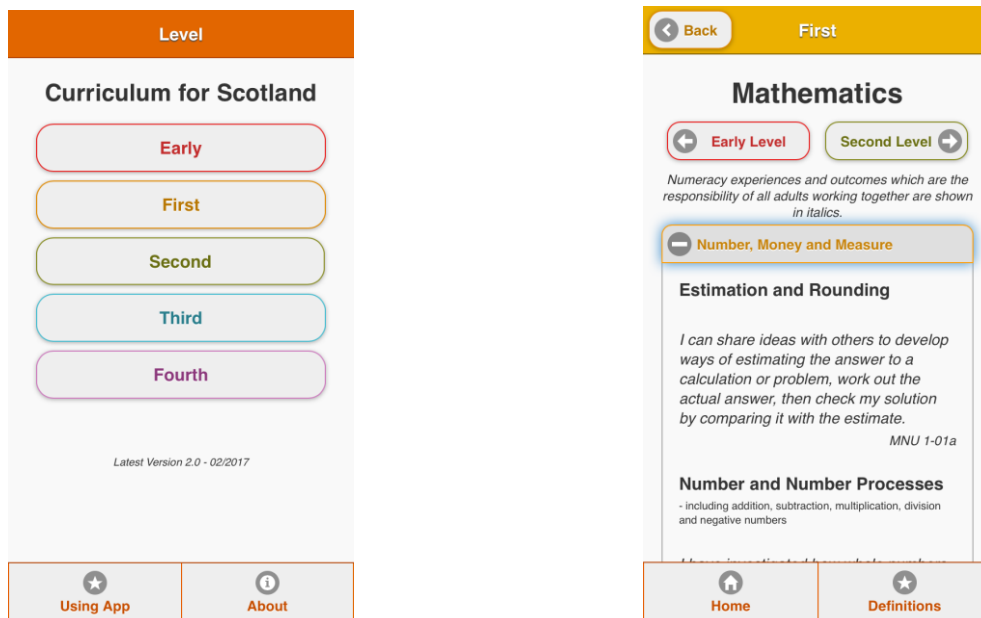
In conclusion, EVOLVE appears to be a very good solution for the management of all aspects of school visits, where it is available. However, as availability is dependent on the local authority, there leaves a significant market for a solution which any teacher can access, regardless of their establishment or local authority.

## 2.1.4 Curriculum for Excellence App

Looking at other types of application designed to support teaching staff, a significant category is applications for managing communication between school and home, such as “Parentmail” [21], or “ClassDojo” [22]. These types of application tend to be adopted at an establishment wide level for staff to use. Beyond these, teachers are free to choose from the many apps available to support specific subjects in their classroom, or for planning and managing their lessons.

One example of such an application is the Curriculum for Excellence app, which was cited by the client at the start of the project as an example of what they were looking for. The app is available on both Android and iOS and was developed using the PhoneGap cross-platform framework (pers. comm.). The app was developed by a teacher with the aim of providing a quick reference guide for the Experiences and Outcomes found in the Curriculum for Excellence document, based on his own experience of difficulties accessing the materials in meetings. According to the developer’s website [23], the content of the app is taken directly from the Education Scotland website and does not alter the wording, it just aims to provide this information in a portable and easily navigable format. The app has 8 User ratings on Google Play and the users have commented that it is straightforward, easy to use and thoughtfully laid out, however the fact that there are so few reviews suggests that the uptake of the app may not be very high.

**Figure 4. Screenshots from the Curriculum for Excellence app**



Tabbed options allow the user to navigate to the Curriculum for Excellence “Experiences and Outcomes” which are used in lesson planning and report writing.

In the author’s opinion, the app does what it set out to do, in that it is very straightforward to navigate and there are clear advantages to having a large document like this on your phone rather than in a bulky document, especially if you need to refer to it often. The Curriculum for Excellence particularly lends itself to being set out in this way as the Experiences and Outcomes are usually brief statements rather than lengthy explanations. The visual layout and design is simple, but consistent throughout and on both Android and iOS there are no obvious bugs or technical issues. This app lacks the features and functionality of more complicated apps like Riskassessor, but its clean design and reliability are its selling points.

### 2.1.5 Summary of Existing Tools

**Figure 5. Table comparing the tools available in the field**

Product Name	Format	Advantages	Disadvantages	Cost
HSE checklist	Paper	Easily accessible Cost effective	Generic and basic	Free
SAPOE toolkit	Paper	Easily accessible Cost effective	Generic and basic	Free
Riskassessor	App	Easy to use Customisable Simplifies the process Saves time	Expensive Technical issues Too much branding	£59.99 + Subscription fees
EVOLVE	Online	End-to-end service Fully customisable Hosted remotely	Availability is dependent on local authority	Annual subscription
Curriculum for Excellence app	App	Straightforward Well laid out Reliable	Limited functionality	£1.99

On the basis of the research outlined above, there is nothing currently available which does specifically what this project aims to achieve. To be useful, the app will need to go further than the paper based flow charts already available but it should not be so complicated that it incurs the costs associated with the bigger and more sophisticated systems. It should be

straightforward to use and save the users time and where possible the app should be customisable to individual user's requirements. Furthermore, it is important that any features it offers work well and user experience should be considered for every design decision.

## 2.2 App Development and Cross Platform Frameworks

For a number of years, the market for mobile applications has been growing and developing. According to Gartner [24], global sales of smart phones are still rising so the trend is likely to continue for the foreseeable future. It is not surprising then, that a number of different approaches to app development have emerged. One factor driving this is that initially there were multiple mobile device platforms, each with its own SDK, programming language and distribution methods (Ohrt & Turau [25]), although Android and iOS currently dominate the market, this still means that to reach the majority of consumers developers must develop their app for at least two completely independent platforms. As a result, there are strong business pressures which make cross-platform app development an attractive proposition.

Although many approaches to app development exist, the options most relevant to this project are native, cross-platform native and hybrid web apps. A summary of the advantages and disadvantages of each will be considered here:

**Figure 6. Table comparing approaches to mobile app development**

	Native	Cross platform Native	Hybrid web app
Programming language	iOS: Swift / Objective C Android: Java	Varies by framework	JavaScript, HTML, CSS
Advantages	Fast, reliable, responsive apps. Native user interface. Access the full functionality of the device	Significant code reuse. Close to look and feel of a native app Test environments available with some frameworks	Access to some native features Easier to learn Save time and money Best suited for prototyping.
Disadvantages	Separate codebase for each platform Takes twice as long, or requires more developers. Expensive	Steep learning curve for some frameworks May still need to use some native code	Often inferior performance to Native apps Less professional user interface

	Native	Cross platform Native	Hybrid web app
Examples	Native iOS app	Xamarin	PhoneGap
	Native Android app	Titanium	Canvas

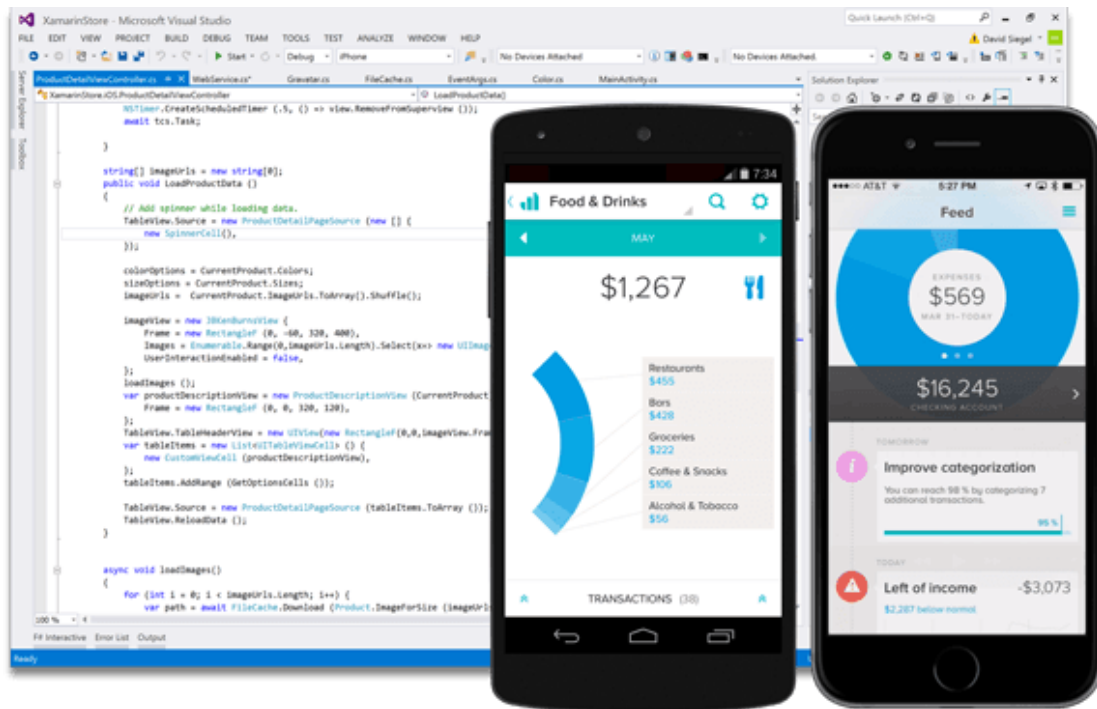
(Note this table contains information from “Native, Web or Hybrid Apps? What's the Difference?” [26])

Developers are faced with a choice, developing an app natively for each platform is widely agreed to produce the best results in terms of user experience, but it is time consuming and expensive on resources, which is a significant drawback in a fast-changing market. Cross-platform apps are quicker and more cost effective to develop and produce a consistent appearance across platforms, however this comes at the cost of design, reliability and user experience. One of the primary aims of this project was to develop an app that would be available to both Android and iPhone users, so it was essential that a cross-platform framework be used as time constraints did not allow for two native apps to be developed independently.

As the benefits of being able to develop once are so high, there are many tools available for cross platform development. For the purposes of this project, research was focussed on the most established options, in the hope that these would be more reliable and better supported than the newer and less well tested alternatives.

## 2.2.1 Microsoft Xamarin

Figure 7. Visual Studio IDE with sample Xamarin app



(Source: Xamarin website [27])

Xamarin aims to produce apps with a look and feel close to native

According to their literature, Xamarin offers a comprehensive package, for developing native apps for the main platforms iOS, Android, Mac and Windows using C# as the programming language. The “Test Cloud” feature provides automated testing for over 2000 devices and technical support is available. Xamarin apps are developed in the Visual Studio IDE. Windows apps can be built in Visual Studio directly using C# code but without using any additional Xamarin tools. However, Xamarin is required to compile the code into an iOS native app or an integrated .NET application for Android. The business level licence is expensive but Xamarin is included free with Visual Studio for individual developers and open source projects [31]. The main disadvantage of Xamarin is that the learning curve is steep if you do not have prior experience of C#, the platform APIs and the Xamarin tool chain.

The following strengths and weaknesses are based on analysis of user reviews on the G2 Crowd website [28].

### Positive points:

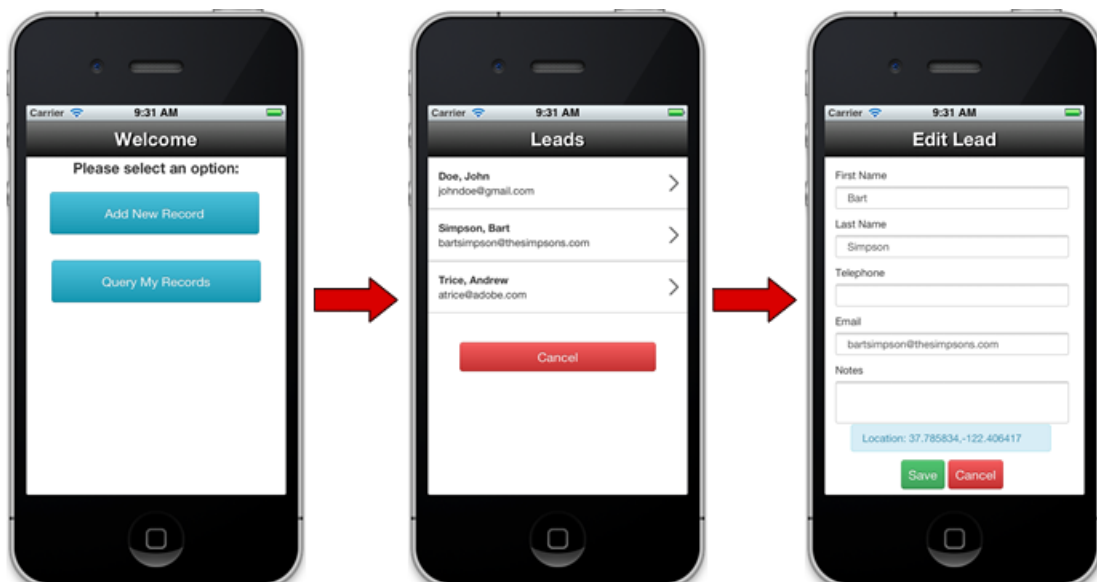
- The fact that you can develop on Windows using Visual Studio which is a popular and easy to use IDE.
- Good debugging tools available.
- Xamarin is well supported with tutorials and for anyone with experience of C# it is quick and easy to learn.

### Negative points:

- Visual Studio takes up a lot of memory (approximately 7Gb) and the apps themselves are often large as they include a lot of libraries;
- Development requires knowledge of the structure of native apps and you cannot access every native feature;
- The developer needs access to a Mac in order to compile the application for iOS
- There are constraints on the free tier of Xamarin which make it necessary for many developers to upgrade to the paid version in order to get access to the features they need.

## 2.2.2 PhoneGap

**Figure 8. Sample PhoneGap application**



(Source: PhoneGap website [32])

The use of buttons, widgets and forms are common features of PhoneGap applications due to the use of UI libraries to style HTML content.

PhoneGap is a product owned by Adobe, and “Cordova” is the open-source engine which provides the core API mapping JavaScript to native code (Ionic Blog[42]). Cordova exists as a separate entity from PhoneGap for licencing purposes and Adobe offers additional tools that tie into other Adobe services under the PhoneGap brand. Therefore, it is possible to use either PhoneGap or Cordova tools to develop the app and the process and results are very similar but research here has focussed on PhoneGap due to the additional features available.

Apps are coded in JavaScript, HTML and CSS and then run in a web view wrapped in native code. Native features can be accessed through plugins which allow communication with native APIs through JavaScript. It is possible to use JavaScript libraries, such as UI libraries, to improve the design. The main benefits of PhoneGap are the speed and low cost of development, but as with all hybrid web apps, it is not suited to large or complicated applications and user experience is often inferior to other options, which developers need keep in mind when making design decisions. Again, based on analysis of user reviews from G2 Crowd [29], PhoneGap has the following strengths and weaknesses:

**Positive points:**

- PhoneGap is a good option for getting up and running quickly.
- The documentation and community support are good and there are plenty of supplementary plugins developed by the PhoneGap community.
- The option to build in the cloud means developers can avoid using native SDKs and the associated costs and effort associated with maintaining these.
- The desktop version of PhoneGap is useful for developers who are not familiar with a command line interface.
- It uses standardised web APIs and it is possible to start development for iOS without access to a Mac.

**Negative points:**

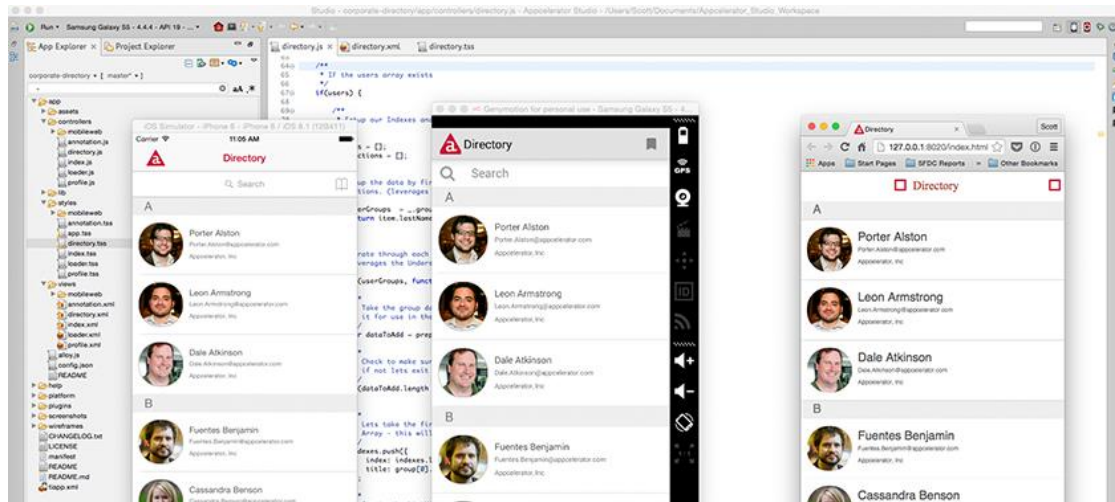
- There are too many options (PhoneGap or Cordova, Build in the cloud, Desktop version, command line interface), which can lead to confusion and they are not all interchangeable.
- The option to build in the cloud is only available for Android, iOS and windows and not for niche platforms.
- Extending native functionality is not easy and you cannot make changes to native code if you are using the cloud option as everything is handled remotely.



- It is not as robust as other frameworks and it is not suited to apps which require a lot of graphics.

### 2.2.3 Titanium

Figure 9. Appcelerator IDE with sample Titanium app



(Source: Appcelerator website [33])

Titanium combines HTML content with native UI objects to give a more native look and feel than HTML on its own.

Owned by Appcelerator, Titanium uses JavaScript HTML and CSS to deliver native apps. As with the other options, Titanium offers access to the device's native features and 60 – 90% code reuse. It works by compiling native objects into native code and running business logic in JavaScript at runtime. According to Appcelerator's own blog [34], this approach gives the app a native look and feel but requires more adaptations as some native features will be available on one platform but not the other. The IDE includes a drag and drop palette for visually designing the app and there are also comprehensive analysis tools available with the professional bundle. The basic package is available free to individual developers. The advantage of Titanium is that the developer has better access to native features, whilst coding in JavaScript and HTML, however like PhoneGap it is not well suited to large or complicated apps. Unlike PhoneGap, a good level of familiarity with the mobile environments is necessary and although code can be reused, developers are required to design, develop and test for each platform separately. Based on G2 Crowd user reviews [30], the following strengths and weaknesses were identified:

**Positive points:**

- Appcelerator Studio is a good IDE with a simple to use interface, which is easy to install, learn and get started with.
- They like the fact that Titanium makes use of native elements on the device.
- The tool is evolving with improvements in each new release.
- It is well suited for mock-ups, proof of concept and simple applications.

**Negative points:**

- Technical issues such as memory leakage problems which need to be dealt with manually and missing features which should be part of the main toolkit.
- Limited technical support, in particular community support appears to be diminishing.
- Developers need to use the native SDKs therefore there are issues around updating code when moving to a new version of the SDK.
- It is not possible to use UI libraries to enhance the appearance of the app.

**2.2.4 Summary of Cross Platform Frameworks**

Framework	Advantages	Disadvantages
Xamarin	Powerful tool with extensive features  C# is a common programming language  Visual Studio is a popular IDE	Not well suited to beginners  Limited functionality available in the free tool
PhoneGap	Fast development time  No need to install lots of software or native SDKs  Good documentation and support	Less robust than other options  Not suited to complex apps
Titanium	Simple to use  Fast development time	Technical issues  Limited support

In summary, no single framework has yet cornered the market of cross platform development. Each tool has its strengths and weaknesses and each is suited to different types of project, depending on the skills of the developer and the goals of the project. In the opinion of the author, Xamarin looks like a professional quality tool which would be suited to a business environment where developers with the necessary skills could be employed to make the most of its features. Titanium tries to offer the native advantages of Xamarin with the coding advantages of PhoneGap, but does not appear to do so very successfully, with a lower level of code reuse and users reporting technical issues. Comments about the lack of support were also worrying. PhoneGap does not offer as professional a product as Xamarin, however it does appear to have a good community base and to offer advantages of speed and ease of use which are well suited to a student project.

### **2.3 The PhoneGap Toolkit**

Based on this research, the PhoneGap cross-platform framework was selected for the following reasons:

- It is a well-established framework, its strengths and weaknesses are well understood and there are tutorials, documentation, blogs and forums available to support it.
- It does not require in depth knowledge of mobile environments and the learning curve is not too steep for anyone with experience of HTML and JavaScript. This was a consideration due to the timescale of the project.
- It has APIs for accessing the device's native functionalities, which opens possibilities for how the app can develop.
- PhoneGap is open source and does not add additional charges for using its services. This was a consideration for the long-term maintenance and future of the app.
- The Curriculum for Excellence app was developed using PhoneGap, which supported the case that the aims of this project would be achievable in PhoneGap.

PhoneGap offers several different tools to build an app, at the start of the project, it was a case of trial and error to determine which option would be most suitable.

**Figure 10. Summary of products available from PhoneGap**

Tool	Description
PhoneGap Build	<p>Upload the app files to the cloud Build service and the app is compiled there.</p> <p>Fast compilation time without needing to install and maintain native SDKs.</p> <p>Easy to share the app with collaborators and testers via the website.</p>
Cordova CLI	<p>Allows access to the Cordova functionality but not additional PhoneGap features.</p>
PhoneGap CLI	<p>According to PhoneGap literature “the most powerful and flexible way to use PhoneGap”.</p> <p>The CLI is written in Node.js (an open source framework for running JavaScript on a server), it can be used on any platform and gives the developer most control over their environment.</p> <p>Can be used in conjunction with PhoneGap Build or native SDKs to build the app.</p>
Desktop App	<p>The simpler alternative to the CLI, for users who prefer to use a GUI.</p> <p>Uses the same libraries as the CLI and allows the developer to create apps and serve them to connected devices via the Developer mobile app.</p> <p>Used in conjunction with the native SDK to build the app.</p>
Developer Mobile App	<p>Used in conjunction with the Desktop app, saves the developer time by allowing previews on a mobile device without the need for compiling, signing keys or installation of the app during the development stage.</p>

(Source: PhoneGap [39][40])

### 2.3.1 Selection of PhoneGap Build and the Command Line Interface

The PhoneGap literature promotes the Desktop app as the quick and easy way to get up and running with PhoneGap, but the drawback of this approach is that it is necessary to install the SDKs for each platform you are targeting in order to actually build the app for deployment.

Also, the config.xml file it generates is not compatible with the alternative method, PhoneGap Build so once you commit to one method, the two options are not interchangeable.

PhoneGap Build allows the user to upload the app files to the cloud and the app is built there, without the need to install and maintain the native SDKs. The app files can then be downloaded on to any device from the website. This was the approach used in this project as the author found the Build tool to be quick and easy to use. Initially files were compressed and uploaded manually, but as the project progressed and the technical challenges increased, the PhoneGap CLI was installed to access the full range of features on offer. The CLI can be used in conjunction with PhoneGap Build as well as native SDKs, so the introduction of the CLI after the project started was seamless and did not clash with any of the work already completed.

## **2.4 The Agile Approach**

It was the intention from the outset that an Agile approach would be taken in this project. Agile is an umbrella term covering a variety of methodologies each with subtly different practices, such as Scrum, Kanban, XP etc, but these all share some features in common and it is these common features which were adopted, rather than following a prescribed methodology. Given that this is an individual student project rather than a business setting it should be considered that Agile was the inspiration for the approach used rather than a strict application of it.

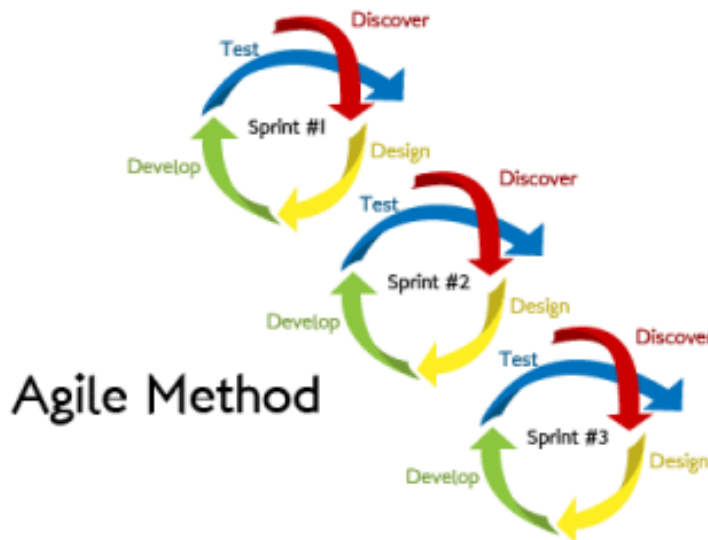
Broadly speaking, Agile is lightweight, iterative and adaptable with a focus on Client/User input throughout the process. The development lifecycle is typically split up into phases or ‘sprints’ with the focus on developing an increment by the end of each sprint. These principles are suited to this project because they are designed for developing software quickly with a flexible approach to requirements. The textbook alternatives to Agile are the “heavy weight” approaches such the Waterfall model. While these models help us understand the stages of the software lifecycle, their rigidity make them impractical in most real-life situations and therefore these alternatives will not be considered here.

### **2.4.1 Incremental development**

Given that time was a critical consideration in the project, it was agreed that it would be better to get each increment of the app working well before moving on to the next, rather than having a broad scope that was poorly implemented. Furthermore, at the first meeting with the client it became clear that the scope of the project was fairly open-ended, and therefore a flexible and adaptable approach would be necessary. The initial project proposal split the development up

into four sprints, each of two weeks, with meetings planned at the beginning of each sprint. This schedule was broadly followed in the project, with additional meetings added where necessary and additional time at the end for wrapping up the project.

**Figure 11. Diagram illustrating the Agile Method**



(Adapted from: “Agile Software Development Basics and fundamentals – CodeProject” [35]).

#### **2.4.2 Stakeholder Engagement**

Three major stakeholders in the project were identified from the outset. These were Mike Harvey at the Low Port centre who acted as the Client; the project supervisor, who acted as the main technical stakeholder and teaching staff were identified as end users. Each of these stakeholders were engaged at regular intervals throughout the duration of the project, to give feedback on work done and to agree next steps. The views of these stakeholders were also sought at the end of the project to evaluate the success of the app.

As the project developed it became apparent that SAPOE could also be seen as a stakeholder in the project. Although input from SAPOE was not pursued at regular intervals during the development stages, their permission was gained at the outset to use the materials from their website and e-mail contact was established with the Communications Manager to clarify any questions arising concerning the materials. Furthermore, the final app was demonstrated to a member of the committee and a slot in their next meeting was also offered for us to present to the panel.

### **3 Requirements Engineering**

This chapter looks at how the requirements for the project were gathered, interpreted and translated into a design for the application. Understanding these requirements is a critical step in the software engineering process for ensuring the product meets the client's expectations. The information gathered here provides a set of acceptance criteria against which the final product can be evaluated.

#### **3.1 User Requirements and Use Cases**

The User requirements were elicited in the form of discussions with the client and with the project supervisor. Where possible the opinions of teaching staff were sought and these fed into the discussions. In addition, the concrete materials that were used in the discussions were the Curriculum for Excellence App as a model example and the "Going Out There" toolkit as the basis for the content, so these both influenced the direction of the project. The design process was iterative, each stage building on the previous one, and once the client was happy with the work the next requirements were discussed.

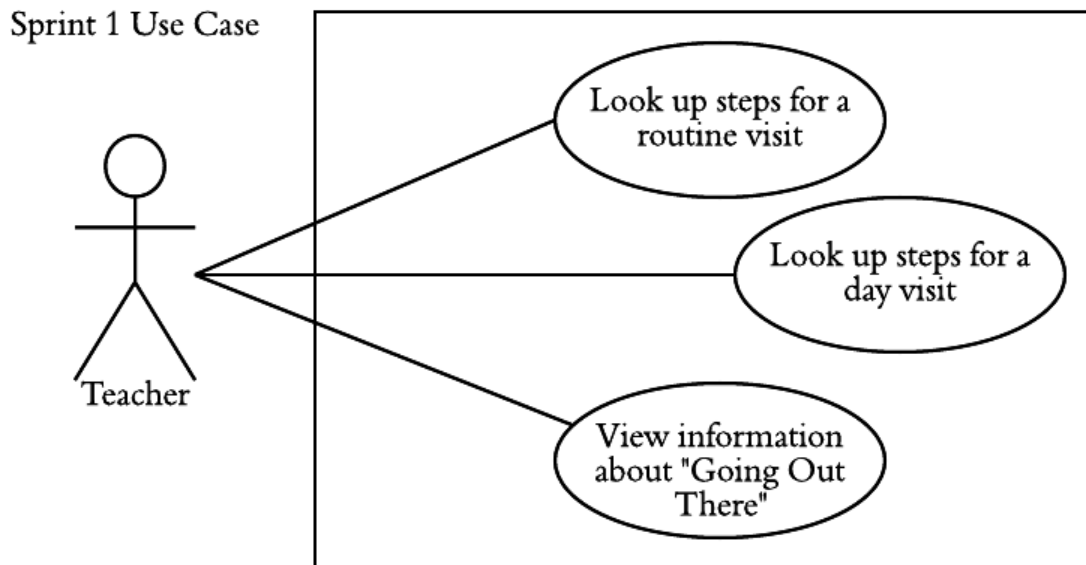
##### **Sprint 1**

The initial requirements gathered to support a minimum viable product were outlined in the early meetings. These were as follows:

- The app should be available on both iOS and Android.
- It should provide an easy to follow format for teaching staff to access the relevant guidance for planning an off-site school trip.
- The content should be taken from the "Going Out There" documentation and credit given to SAPOE as the source of the information.
- It should be "simple, interactive and dynamic", for example give tabbed options leading to the relevant information, as in the Curriculum for Excellence app.
- Low maintenance in the long term.

These points were refined to give three use cases for the initial design of the app, as illustrated in Figure 12.

**Figure 12. Use Case 1: Initial Requirements**



**Sprint 2 and 3:**

The requirements from Sprint 1 were implemented and demonstrated to the client. The resulting discussions lead to the establishment of requirements for sprint 2. These requirements turned out to be the most technically challenging and time consuming to achieve, therefore these were completed over the course of sprints 2 and 3, with meetings continuing over this time to review the work and refine ideas.

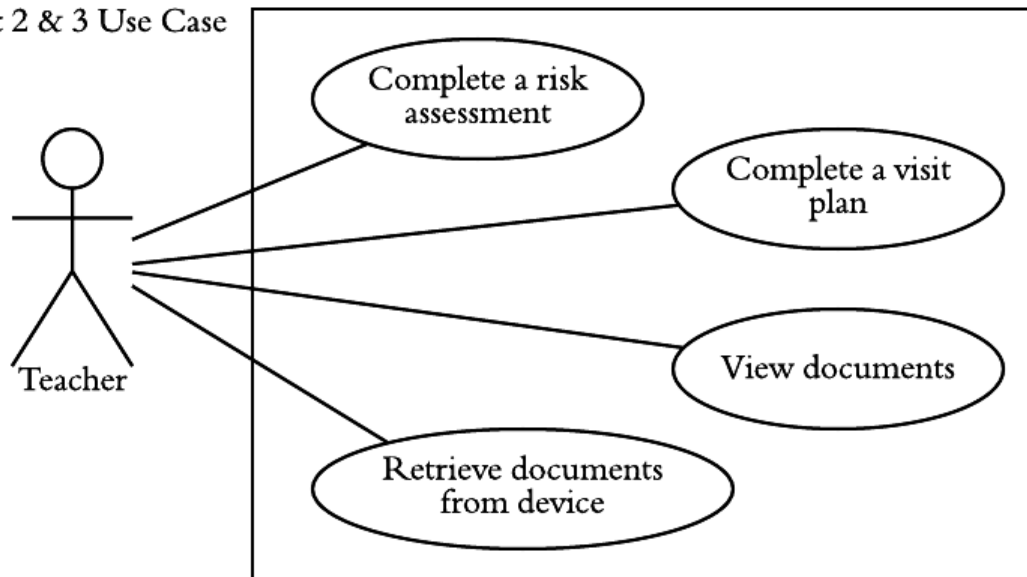
- Include example forms to show how the documents should be filled out.
- Focus to be on local visits rather than all visit types, therefore options should be available for “Routine and Expected Visits” and “Day Visits”.
- Working title changed from “Going Out There” to “Get Local”.
- Add more interactive features, for example functionality to complete, generate and print off the relevant forms.
- The app should work off-line

These points were used to construct four further use cases as detailed in Figure 13.



**Figure 13. Use Case 2: Additional Requirements**

Sprint 2 & 3 Use Case



**Sprint 4:**

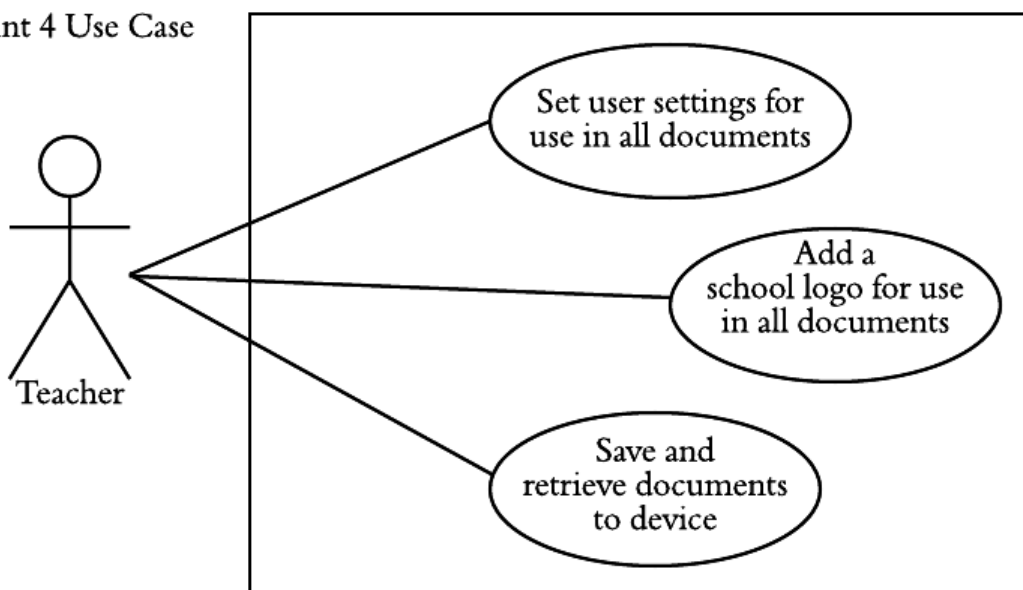
The result of the work in sprints 2 and 3 gave the app all its essential functionality. The final sprint added features to make the app more professional, polished and complete.

- Add User Settings which can be customised.
- Allow the User to add their own establishment's logo to documentation.
- Provide a way for saving, retrieving and deleting documents.

These final points were used to define three final use cases as stated below.

**Figure 14. Use Case 3: Final Requirements**

Sprint 4 Use Case



### 3.2 Non-Functional Requirements

While the User Requirements mainly focussed on the functional requirements of the app, consideration was also given to the look and feel of the app. As highlighted in the review of existing tools in section 2.1, these factors make a big difference to user experience and ultimately the success of a product. The following factors were identified as important considerations:

- Ease of use: the app should be easy to navigate and intuitive to use, without the need for additional instructions.
- Responsiveness: there should be minimal delays when using the app.
- Minimal user permissions: Users should not have to give excessive permissions to install the app, this can be very off putting to the user and is unnecessary.
- Appealing visual design: the visual design should be clean, consistent and nice to look at.

### 3.3 Analysis of the Problem

Each of the requirements outlined has implications for the design of the app. As already discussed, a cross-platform framework was selected in order to satisfy the requirement that the app be available on iOS and Android. Cross-platform frameworks come with their own limitations, such as an inferior user interface and implications for User Experience. Therefore, additional consideration needed to be given to minimising these problems.

**Figure 15. Analysis of the Requirements**

Requirement	Design Implication
Available on iOS and Android	Use a cross-platform framework and a UI library to enhance the User Interface
Provide the “Going Out There” material in an easy to follow format	The materials provided in the “Going Out There” toolkit should be mapped to features in the app.
“Simple, interactive and dynamic”	Use GUI features to optimise how the information is accessed and displayed.

Requirement	Design Implication
Include example forms	Convert forms to a format that will display well on a mobile device and provide an appropriate method for displaying documents in this format.
Low maintenance	<p>Issues of updating and maintaining materials should be considered when making design decisions.</p> <p>Where possible, avoid using tools that will incur fees in the future.</p>
Complete, generate and print off forms	<p>Requires a user interface for input of the information.</p> <p>Will need a method for processing the information and generating the documents.</p> <p>Requires a straightforward way to get the documents off the device.</p>
The app should work off-line	All form processing should be performed using client-side technologies only and everything must be coded into the app.
Add user settings that can be customised	Will need a method for storing the settings within the device.
Allow the User to add their own logo to forms	Requires a straightforward way for the user to add a logo, for example uploading a file or taking a photograph.
Provide a way for saving, retrieving and deleting documents.	<p>Requires a location on the device with read/write access for storing and retrieving the forms.</p> <p>Will need a user interface for the User to access the files.</p>
Ease of use	The user interface should be logical and straightforward and the experience of the user should be considered in the design.

Requirement	Design Implication
Responsiveness	This has implications for the size of the app, how links are handled between pages and how functions are designed.
Minimal user permissions	Do not include any unnecessary features that will require the user to give permissions at installation
Appealing visual design	Keep content to only what is necessary and apply themes and styles thoughtfully. The appearance should enhance the experience of the app.

### 3.4 Acceptance criteria

The statement of acceptance criteria helps to ensure that the resulting product genuinely reflects what the client was looking for at the outset. Based on the analysis above the following checklist of features will be used to assess the final product:

- 1) The app is available on Android and iOS
- 2) The user can view guides for planning a local visit
- 3) The user can view examples of how the form should be filled out
- 4) The user can complete the forms they require using the app
- 5) The user can preview and access the forms once completed
- 6) The user can customise the app with user settings
- 7) The user can save and retrieve previous documents

## 4 Designing the Application

The user requirements and their analysis fed directly into the design of the application. This chapter focuses on the process of converting the requirements into a design for the app.

### 4.1 Content

As already stated, the material provided for the content of the app was the “Going Out There” document and associated toolkit and it was decided that the app should focus on visits to the local area as these are the most common types of visit planned by schools. The following definitions are provided to clarify the nature of the material and the wording used:

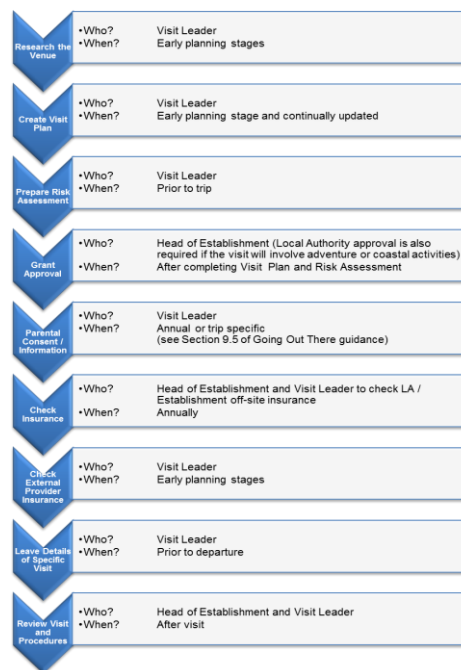
- **Risk Assessment** is the document detailing potential hazards associated with a trip and control measures in place to mitigate those risks. The assessment should be carried out for each venue prior to any visit.
- **Visit Plan** is the document detailing the aims and objectives of the visit, how the visit will be carried out, and steps that have been taken to organise the visit. It must give enough detail for the head of establishment to approve the visit.
- **Parental Consent Letter** is a letter to parents giving details of the trip and asking for their consent for their child to participate. It should be noted that most local authorities have a system in place for handling parental consent.
- **Routine and Expected Visit** is a visit to a local venue involving easily managed activities. These visits happen on a regular basis and are completed within normal school hours. For example, a weekly trip to a swimming pool or the use of a local park.
- **Day Visit** is a visit that requires additional planning and is a less frequent occurrence, such as an annual outing to a museum or a class trip to the theatre.
- **Flow Charts** are part of the SAPOE tool kit to walk the user through the steps of planning a specific type of trip (i.e. a routine and expected or a day visit), as discussed in section 2.1.1 below.
- **Generic Forms** are included in the tool kit in the form of a risk assessment, visit plan and parental consent letter without any details completed. These can be viewed as a template.

- **Example Forms** are also part of the toolkit and provide the user with a worked example of a specific risk assessment, visit plan and parental consent letter. As a model of how these forms are intended to be completed.

The flow charts, the generic forms and the example forms were mapped to features of the app. Note that as most local authorities have systems in place for gaining parental consent (for example, in West Lothian a one-off form is sent out at the start of the school year to obtain parental consent for the duration of that year), it was decided that functionality to generate a parental consent letter would not be required in the app. However, the example parental consent letters are included in the app as guide, should a teacher need to send out additional information about a given trip.

Below are examples of the forms, all available from the Going Out There website [1]. For each type of visit (Routine and Expected and Day Visit), the tool kit includes a flow chart, generic forms for risk assessment, visit plans and parental consent and example forms of each of these.

**Figure 16. Flow Chart for planning a Day Visit**



The flow chart gives each step in the process of planning a Day Visit with details of who should have responsibility for this step and when it should be carried out. This information was a starting point for the app and was used as a basis for the Step by Step guides, in conjunction with more detailed guidance from the main Going Out There document to clarify each step.

**Figure 17. Worked example of a Routine and Expected Risk Assessment**

DESCRIPTION OF TASK / ACTIVITY		Loch View Primary School		
LOCATION		Range of activities		
		Green Loch Park		
Item	What are the hazards?	What are you already doing, i.e. what Control Measures are already in place?	Further Actions Required	Implementation Plan for Further Actions
1	Issue whilst walking to venue <ul style="list-style-type: none"> <li>Crossing Green Rd</li> </ul>	<ul style="list-style-type: none"> <li>Adequate staff ratio for group</li> <li>Staff are familiar with the route - use pedestrian crossing on Green Road</li> <li>Group has been briefed about how to cross in groups with nominated staff member</li> <li>Emergency procedures in place</li> </ul>	Staff should wear high vis vests – these need to be bought	JJ by August 2013
2	Issue at the venue <ul style="list-style-type: none"> <li>Glass on ground</li> <li>Falling in pond</li> </ul>	<ul style="list-style-type: none"> <li>Adequate staff ratio</li> <li>Staff have prior knowledge of the venue</li> <li>All new staff familiarised with venue</li> <li>Staff will check for glass at the bench area on arrival</li> <li>Pupils will be supervised at all times around the pond</li> <li>If the pond banks are muddy and slippery staff have a Plan B available</li> <li>Emergency procedures in place</li> </ul>		
3	Adverse weather	<ul style="list-style-type: none"> <li>Weather forecast will be obtained if necessary</li> <li>Clothing to be checked before leaving</li> <li>Staff will Carry spare clothing when necessary</li> <li>Staff will have a Plan B available</li> <li>Emergency procedures in place</li> </ul>		
<b>Prepared by:</b>		Whole staff team	<b>Date:</b>	12.9.12
			<b>Date for review:</b>	12.9.13
<b>Checked and Approved by:</b>		J. Newton (Head of Establishment)	<b>Date:</b>	12.9.12

The example documents show how the forms should be filled in for a *specific* trip. In this case, a risk assessment for a routine visit to a local park identifies potential hazards with walking to the venue, hazards at the venue and considerations about the weather. It gives details of how these hazards are being controlled and plans for further actions. The examples are designed to give teachers an idea of the details expected on the forms, if they are unsure.

**Figure 18. Generic Visit Plan for “Routine Visits”**

- This is a generic Visit Plan and provides suggestions for what should be included in a Visit Plan.
- The purpose of a Visit Plan is to record any decisions and training that have been carried out.
  - It should give sufficient information for the Head of Establishment to approve the visit.
  - See the sample specific Visit Plan for how this generic document can be used in practice

ROUTINE VISIT TO .....

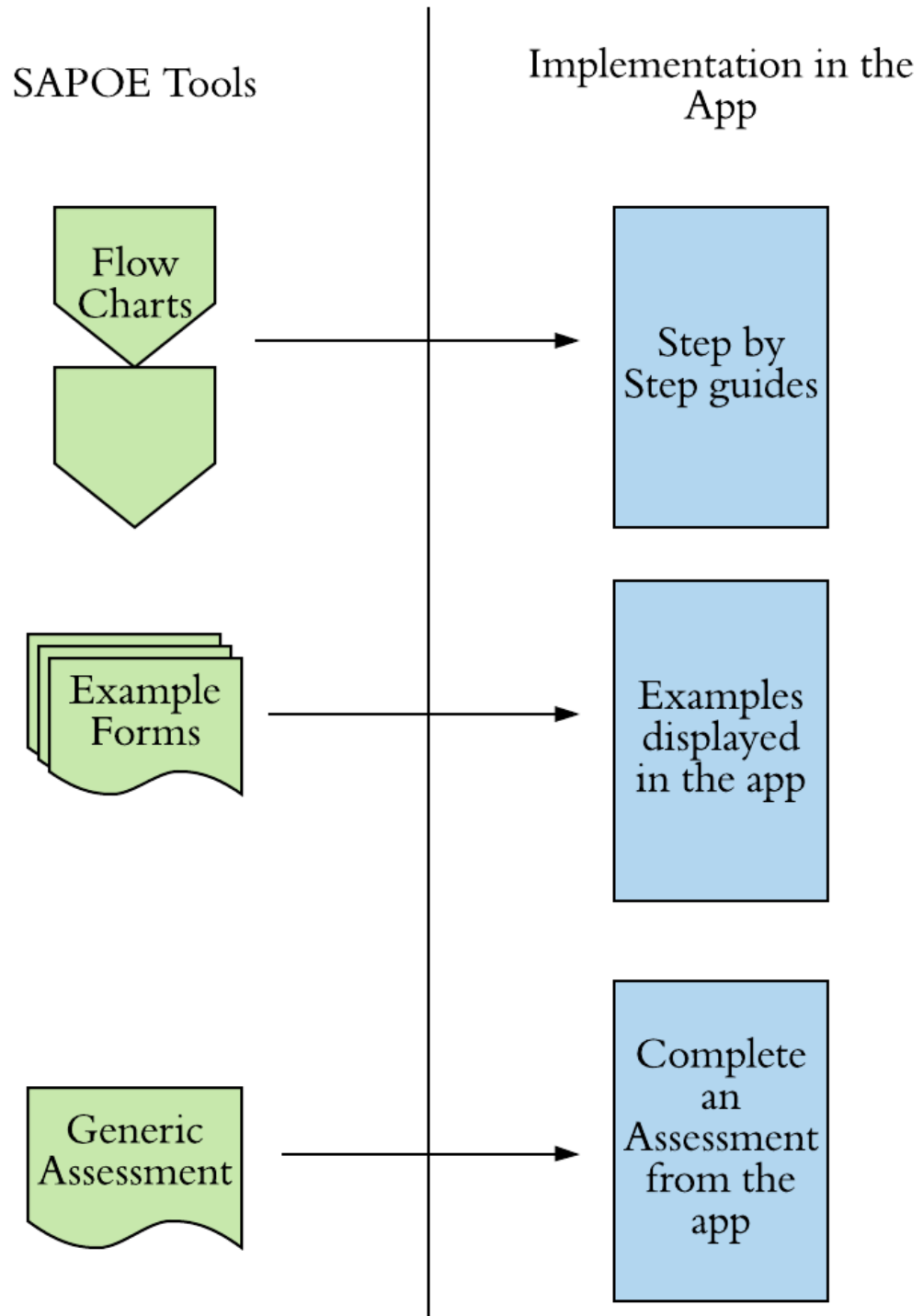
ITEMS TO BE PLANNED		Typical examples of detail to be provided
1	Information to parents/consent	<ul style="list-style-type: none"> <li>• provide information in induction material</li> <li>• give verbal reminder to group before visit</li> </ul>
2	Staff visit to venue	<ul style="list-style-type: none"> <li>• preparatory visit to site</li> <li>• carry out assessment of hazards at venue</li> </ul>
3	Staff/participant ratios	<ul style="list-style-type: none"> <li>• establish any factors affecting the ratio</li> <li>• ratios agreed by staff team</li> </ul>
4	Getting to the venue	<ul style="list-style-type: none"> <li>• how travel by vehicle will be managed</li> <li>• how travel on foot will be managed</li> </ul>
5	Equipment required	<ul style="list-style-type: none"> <li>• group personal clothing</li> <li>• list equipment to be taken by leader</li> </ul>
6	Managing the activity	<ul style="list-style-type: none"> <li>• supervision arrangements</li> </ul>
7	Medical needs	<ul style="list-style-type: none"> <li>• gather medical details of participants</li> <li>• carry medicines as necessary</li> </ul>
8	Weather	<ul style="list-style-type: none"> <li>• obtain forecast</li> <li>• change plan if necessary</li> </ul>
9	Emergency procedures	<ul style="list-style-type: none"> <li>• leave list of participants left at establishment</li> <li>• arrange contact person at base to be available</li> </ul>
10	External provider (if applicable)	<ul style="list-style-type: none"> <li>• discuss programme to be supplied</li> <li>• check provider's insurance</li> </ul>
11	Risk Assessment completed	<ul style="list-style-type: none"> <li>• see generic and specific Risk Assessments</li> <li>• review Risk Assessment after visit if necessary</li> </ul>
Add further items as required		List details for specific activity
Routine Visit approved by Head of Establishment		Signature
		Date
		Date for review

The generic forms give the details of what type of information should be included on the forms, without giving specific details. Although similar to the example forms they do not consider any details such as the location or the circumstances of the visit and just give general information about what should be included. For example the staff/participant ratio is listed as “ratios agreed by staff team”. Whereas on an example Visit Plan a literal ratio would be given such as 1:7.

The design and content of the app drew heavily on the Toolkit forms. Firstly, it was important that the information in the app was a true reflection of Going Out There, so as to give the user accurate advice and the format of the toolkit was also used to inspire the design. Figure 19 illustrates how each of the tools was mapped to features of the app.



**Figure 19. Mapping SAPOE tools to features of the app**



The Flow Charts were adapted into step-by-step guides (with additional text taken from the “Going Out There” document to clarify each step). The Example forms were added to the

guides to clarify how the forms should be completed and the generic forms were used as the basis for completing a risk assessment or visit plan functionality of the app.

## 4.2 Initial Designs

Before programming started, preliminary discussions were expanded to designs on paper to give a visual idea of what the app should look like and how it should behave. This was particularly useful when discussions were vague, as it helped to identify more specific details of the design. As such, design sketches were mostly used at the start of the process and to conceptualise new features of the app, rather than for expanding or refining existing features.

**Figure 20.** Initial sketch of the app

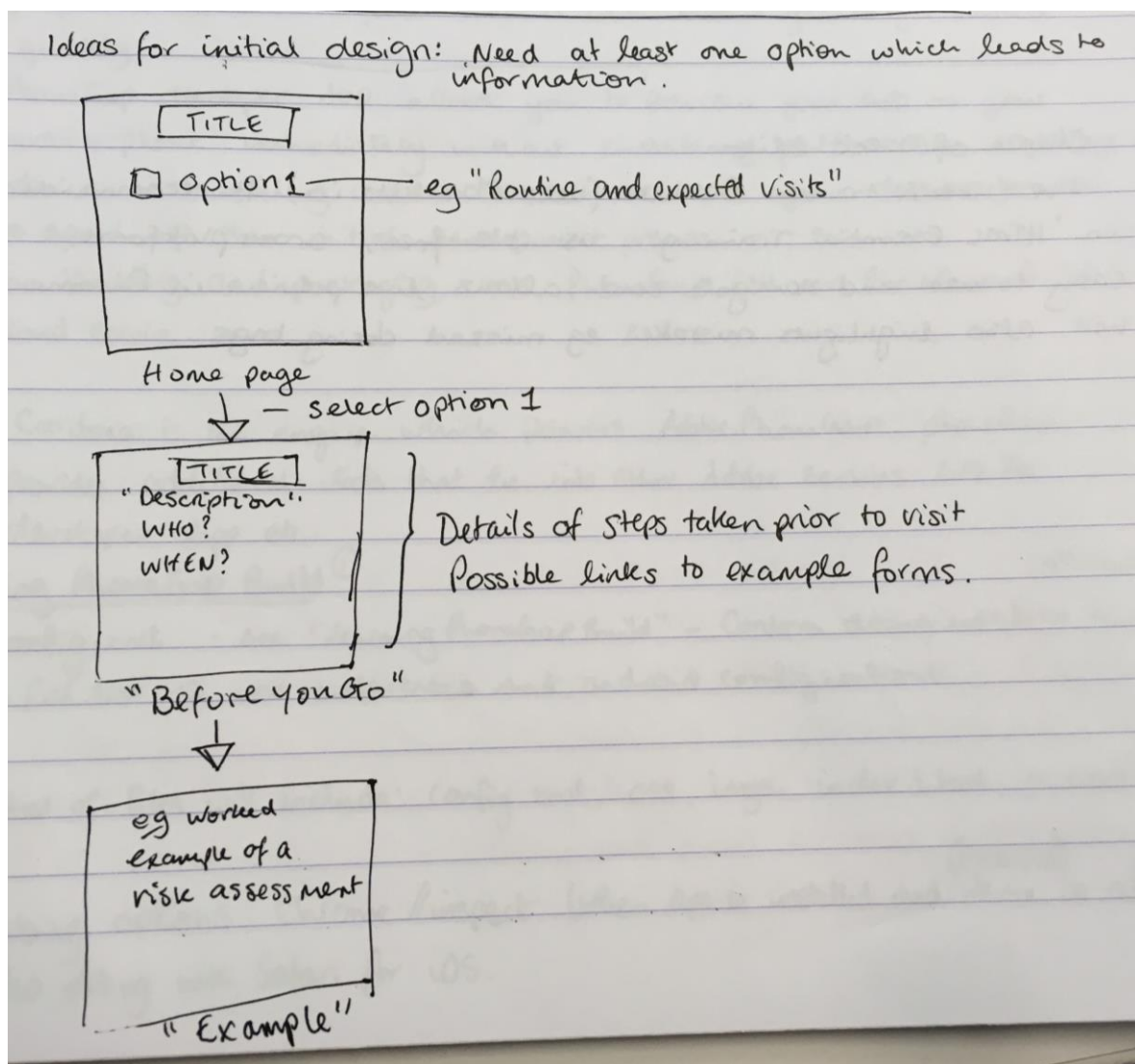
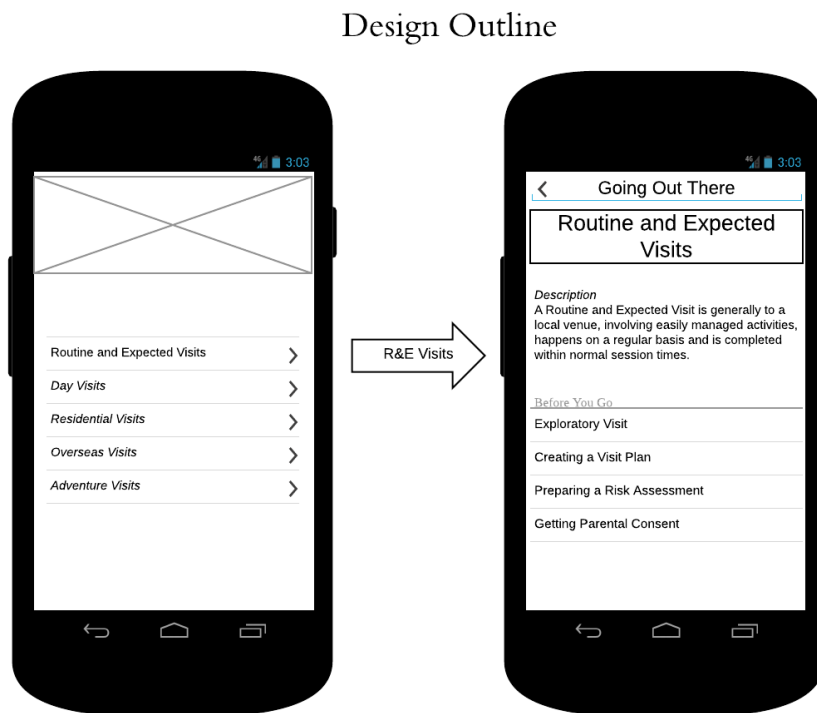


Figure 20 shows a sketch drawn up after the first design meeting with the client, where the client asked for a page with at least one option, which when selected would lead to more information about that option. The sketch led to the mock-up of a user interface Figure 21, which was used to establish a shared understanding with the client before programming began.

**Figure 21. A Mock-Up User Interface**

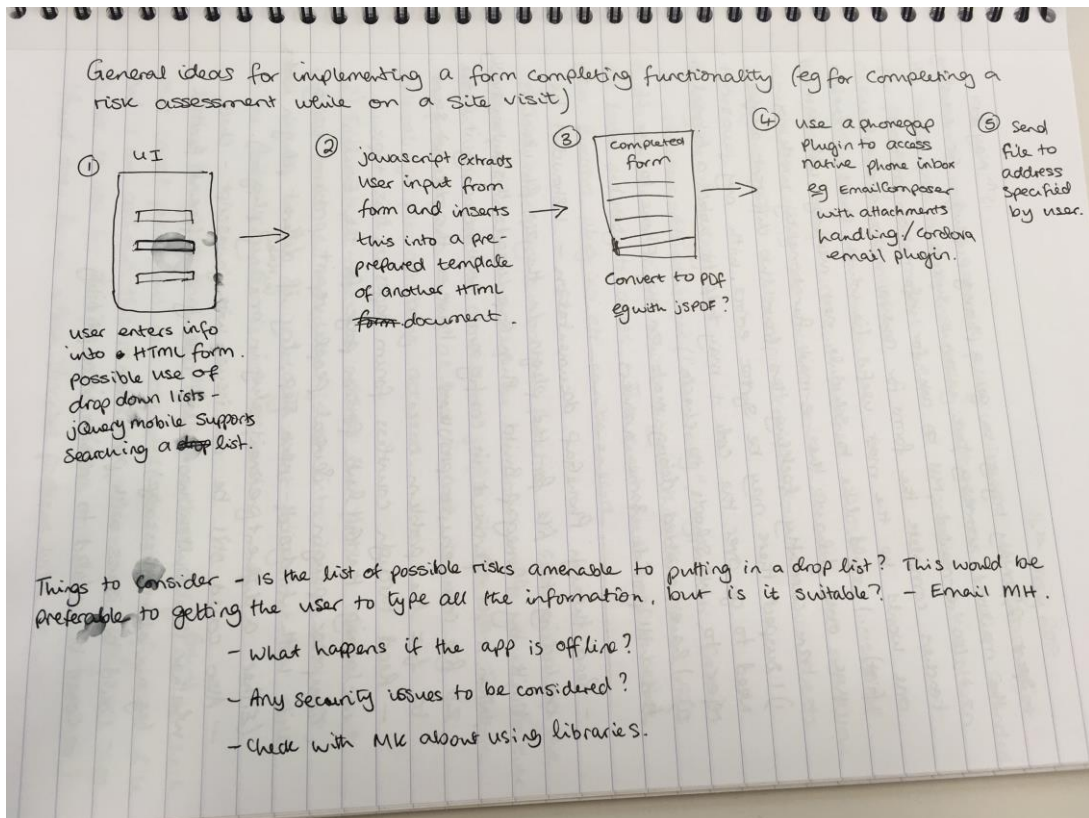


Based on these designs, the first iteration of the app was produced, consisting of a home page displaying options and linking to a second page with information taken from the SAPOE documentation.

Sketching designs on paper was not only used to plan how the app should look but also for thinking about how functions of the app should be implemented. When the functionality to complete a risk assessment was identified as a requirement, it was useful to explore the problem from different angles to ensure all considerations were taken into account. These included:

- A high-level overview of how the feature would work (Figure 22).
- A design of the user interface for the user to input their requirements (Figure 23 and Figure 24).
- Consideration for how the user input would be handled (Figure 25).

**Figure 22. Risk Assessment Design: Overview**



The design considers what steps are required to implement a form-filling feature of the app, to get the user's input, convert it into a document and get the document off the user's phone.

- 1) A user interface is required
- 2) Use JavaScript to process the user input
- 3) Convert the input into a PDF format
- 4) Access the phone's email app
- 5) Send the information by email

Breaking the process down into steps allowed the author to consider what would be required to implement this and ideas for tools to use are included where appropriate, such as the use of HTML forms for the user input, use of jsPDF library to convert the document to PDF and the use of a plugin to access email on the device. It also prompted questions detailed at the bottom of the page to be followed up later, such as the potential use of a drop list to enter the user input and consideration of how the app would behave offline.

Figure 23. Risk Assessment Design: User Interface

The image shows a hand-drawn form design on lined paper. At the top, it is titled 'Form Design' and 'P1.'. The main heading is 'Complete a Risk Assessment'. Below this, the form is divided into several sections: 1. 'Select visit type' with two radio button options: 'Routine + Expected' and 'Day visit'. 2. 'Visit location' with a single-line text box. 3. 'Hazards (select all that apply)' with four square checkboxes. 4. 'Additional hazards' with two multi-line text boxes. 5. At the bottom, two buttons labeled 'Reset' and 'Next'. Brackets on the right side of the form group these elements into categories: 'Radio buttons' for the first section, 'Text box' for the second, 'Check boxes' for the third, 'Text box' for the fourth, and 'Buttons' for the fifth.

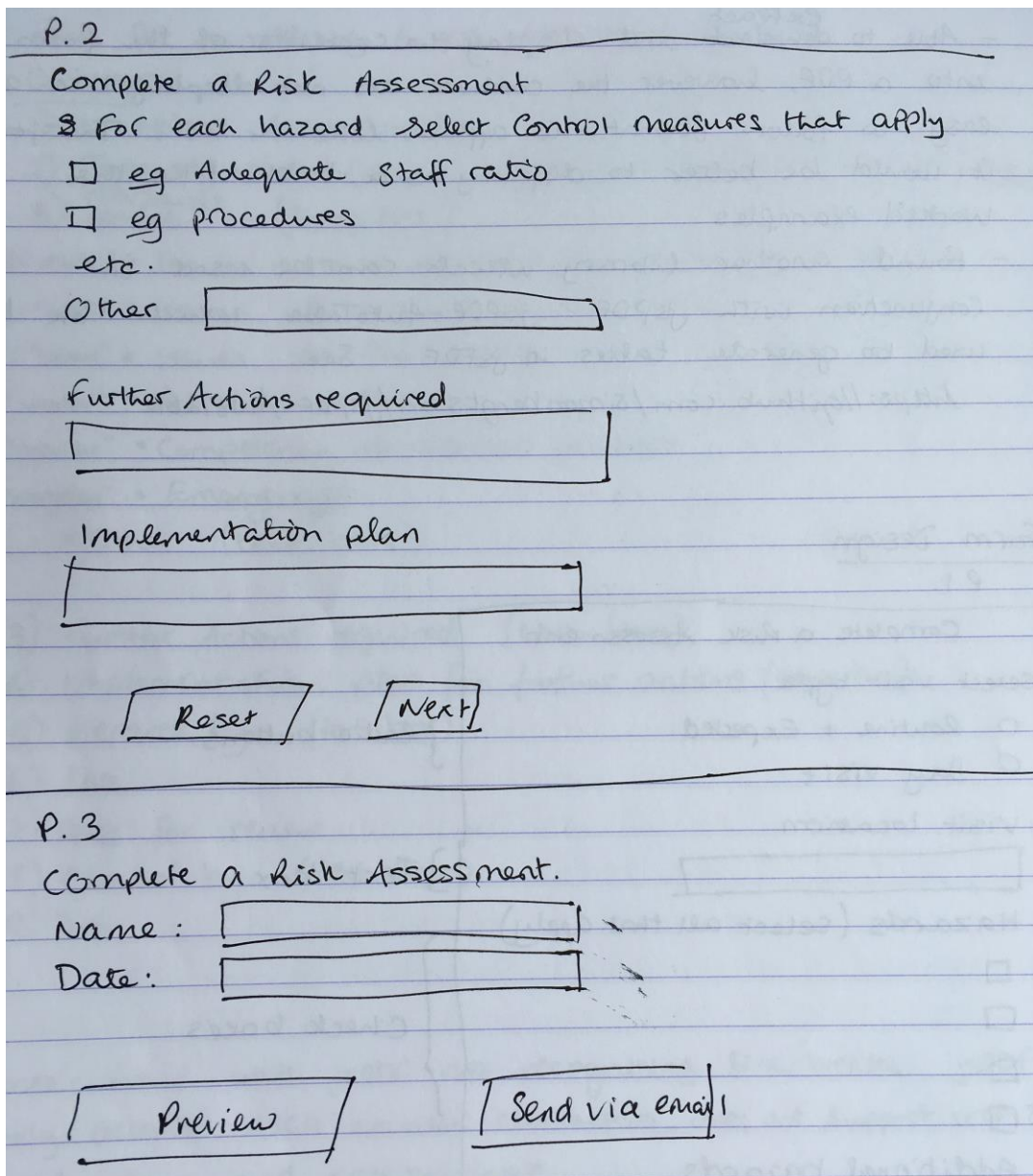
The design for the user interface identifies the use of HTML forms to get the user's input. The use of forms seemed to be an obvious choice for this step as they are designed for collecting user input and they offer a choice of elements for that purpose. The aim was for the user to enter the information quickly and easily and to minimise the amount of information that needed to be typed, as mobile device screens are not suited to excessive typing.

The risk assessment forms for a Day Visit and a Routine and Expected Visit are very similar, there are just more potential hazards and controls associated with a Day Visit. For this reason, it was decided both types of risk assessment could be covered by the same form on the app. Therefore, the first question in the feature prompts the user to select the visit type. Radio buttons were chosen for this purpose as they ensure that one option alone is selected. The visit location was designed as a text box since the location is very specific and not something that can be selected from a list. The hazards are in the form of a list with check boxes so that as

many or as few of these as apply can be selected by the user. The hazards (which are not listed in the diagram) were gathered from the SAPOE toolkit forms and there is an option on the diagram for the user to input extra hazards that are not listed, using text boxes. Initially the author considered putting the hazards into a searchable list for the user to select from, but it was found that the hazards in the toolkit were fairly limited in number and a list with check boxes for selection was deemed a more suitable option.

Finally, it was assumed that in the very least a button to reset and a button to continue to the next page should be available. The “next” button is actually an HTML link styled to look like a button. This styling was used to make the app more user friendly as a plain HTML link would not sit well with the look and feel of the app.

**Figure 24. Risk Assessment Design: User Interface continued**



The sketches for page 2 and 3 of the app continue with the theme of check boxes and text boxes to allow the user to input their information, with as little typing as possible. They do not consider every detail of the design, but allowed the author to take a high-level overview of what would be required. They were used to consider what kind of information the user might need to enter and how the user should input that information. The designs identify:

- A three-page architecture for the risk assessment form.
- Use of radio buttons, check boxes, text boxes and buttons for user input.
- Some of the detail of the kind of information required, such as hazards and control measures.

**Figure 25. Risk Assessment Design: Handling the input**

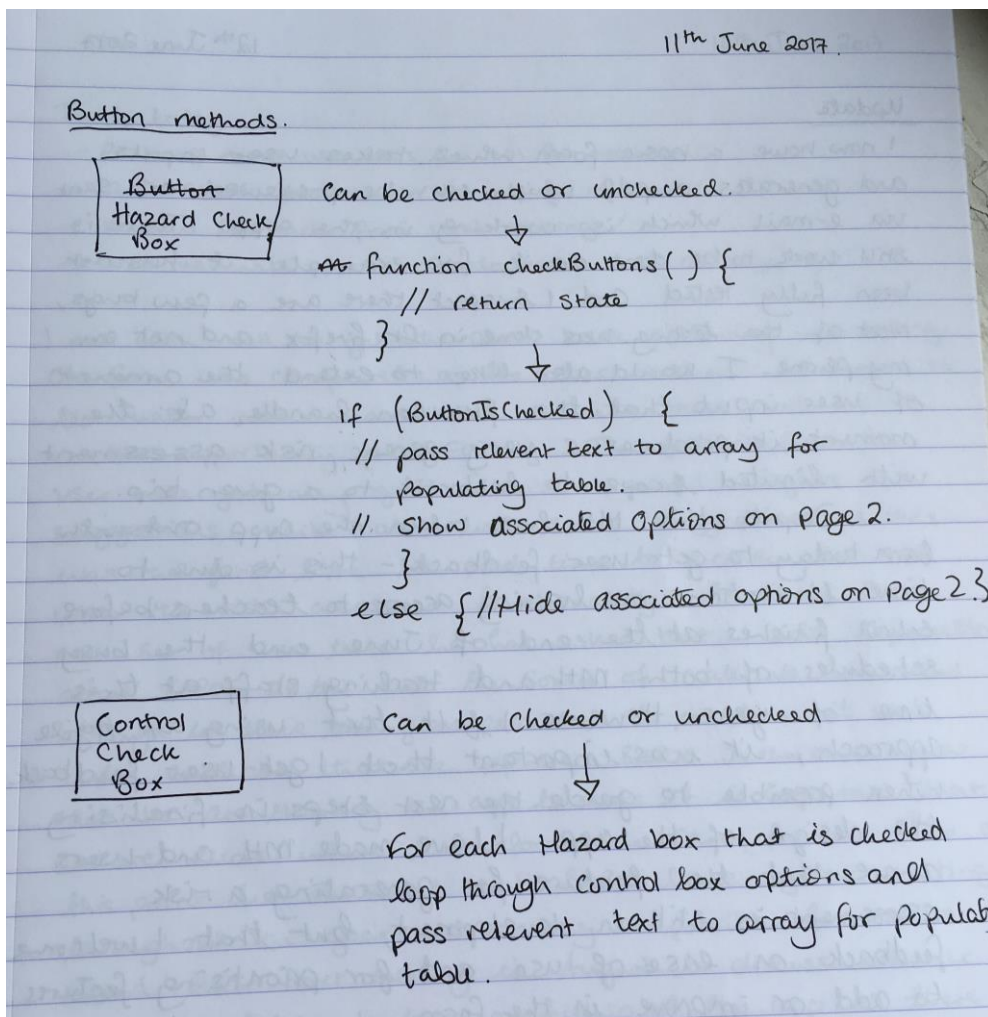


Figure 25 considers how to approach the logic used to process the information from the forms. The author had prior experience using PHP to process form input but not using JavaScript for this purpose, so particular consideration was given to how this could be achieved. In the inter-

ests of an uncluttered and user-friendly interface, the aim was to design the forms such that only the control options for hazards selected on page 1 would be displayed on page 2 (i.e. the content displayed on page 2 would change dynamically based on the selections made on the first page), this functionality is considered on the first half of the page. The second half considers how to ensure the control options are paired with the appropriate hazard when displaying them on the resulting PDF.

The functionality to complete a visit plan built on the design of the risk assessment feature and as such diagrams were deemed unnecessary for the development of this feature.

### **4.3 Further features**

Further features that were identified in the requirements analysis include the ability to generate other relevant forms as well as the Risk Assessment, acknowledgement to SAPOE as the source of the materials, User Settings, the ability to add an establishment logo and the ability to save and retrieve documents. Although no diagrams were produced to aid the design of these features, the decisions behind their design will be briefly discussed here.

#### **4.3.1 Generating Further Form Types**

When the form generating feature was first discussed with the client, priority was given to generating a risk assessment as this was deemed the most useful of the forms to generate from the end user's point of view. Once this functionality was working, the same approach was applied to the Visit Plans, with two main differences from the Risk Assessment. Firstly, the two types of visit plan were considered too different to accommodate in a single form and secondly, all the information is entered into text boxes as the nature of the Visit Plan is more specific and not so amenable to selecting options from a list. Therefore, two separate forms were designed, one for Routine Visit Plans and one for Day Visit Plans. Finally, the parental consent letters were considered but because of the way many local authorities handle parental consent (forms are issued to parents on an annual and not a visit by visit basis), it was decided that the functionality to generate a parental consent letter would be redundant in most circumstances and was not pursued.

#### **4.3.2 Acknowledging SAPOE**

The use of the SAPOE toolkit was granted at the outset of the project under the condition that adequate credit would be given in the app. Aside from this understanding, there are further benefits to citing the source of the material – for the credibility of the app and in the interest of making the app more useful by providing a link to those materials. Different approaches to



providing the acknowledgement were considered, such as a pop-up message at the start of the app, however it was felt that the best solution would be to include an “About” page which would summarise the information and provide a link to Going Out There. It should be noted that this is the approach used in the Curriculum for Excellence app and the author felt that this was a neat approach to providing the user with more information in an unobtrusive way.

### **4.3.3 User Settings**

User settings were proposed as a way to help save the user time by allowing them to enter information once, however only two pieces of information were identified as being suitable for this feature – the user’s name and the name of their establishment. These are the only two features that are entered into all types of form and that are likely to remain the same from one assessment to the next. When the functionality to add a logo was also added as a requirement the logical place to access this was the user settings, with the theory that the user would set these up at the beginning and then forget about them. In the Riskassessor app discussed in Section 2.1.2, the user is required to enter user settings before they can use the app, but this approach risks annoying the user and it was hoped that “User Settings” should be a sufficiently unambiguous title to allow the user to decide whether they wanted to use this feature or not.

### **4.3.4 Saved Documents**

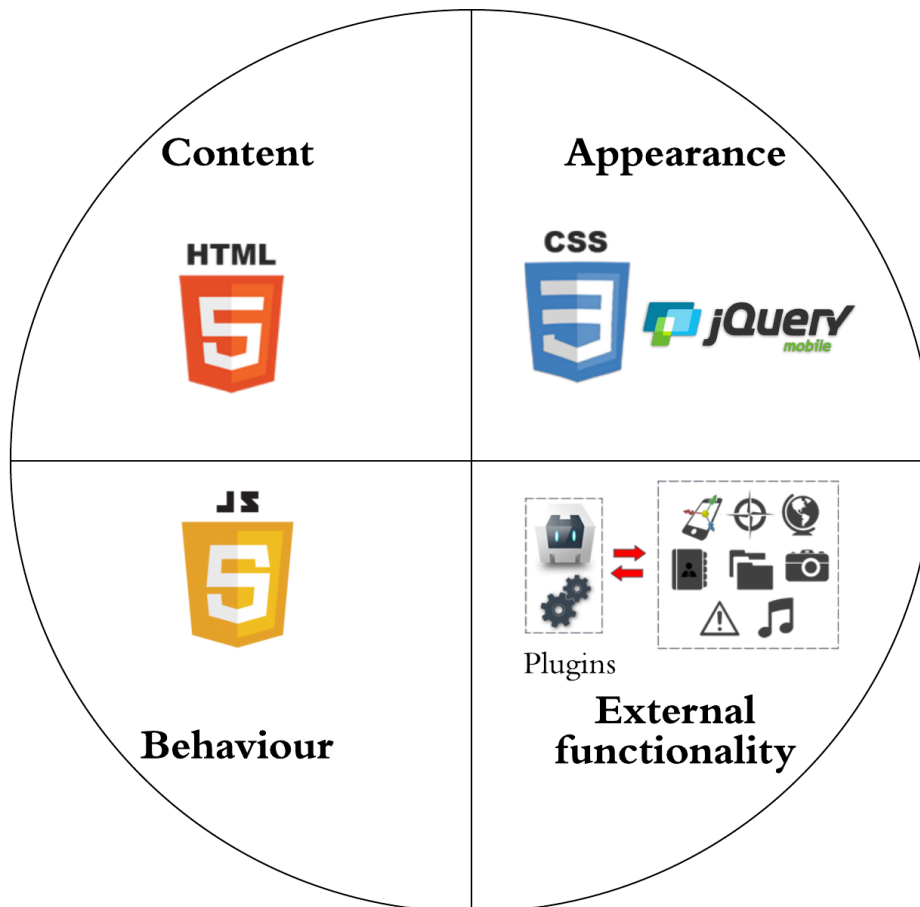
The design for saved documents posed a problem, which unfortunately was not solved by the end of the project. By the time this feature was proposed, everything was already in place to save and retrieve documents from the device’s storage. Therefore, it appeared to be a straightforward feature to implement. The main issue was a design problem, in that a suitable user interface for viewing, retrieving and deleting documents, was not successfully implemented in time for testing to start, although given more time to research this, a suitable option most probably would have been possible. This issue is discussed more fully in section 5.7 on Un-solved Problems.

## **Designing for PhoneGap**

The design was understandably influenced by the tools used to make the app. The application is essentially a web app so the approach was more similar to designing a website than a native application. For example, the content, appearance and navigation between the pages of the app were handled by HTML, CSS and jQuery Mobile. This combination was used for all interface requirements and made it possible to apply consistent styling throughout the app. The downside of using these tools is that they limit the level of detail available to the developer. For example, only limited changes to the appearance (such as colour scheme, font and styling)

are available and it is not possible to design bespoke interface features. Furthermore, the developer is relying on the browser to render the features as intended, which leads to differences in appearance from one device to another and issues if any of the features are not supported by a browser. An example of these differences is the way the browsers handle text boxes. In Chrome on an Android device, voice recognition is available for entering text but this is not offered automatically in Safari at present. Another example is the way in which pop-up messages are displayed. The upshot of this is that when designing the user -interface the developer has to appreciate that they can design things in a general way but the specific details are sometimes out of their hands.

**Figure 26.** Diagram illustrating how the tools relate to design



(Adapted from “PhoneGap explained visually” [41])

## 5 Implementation

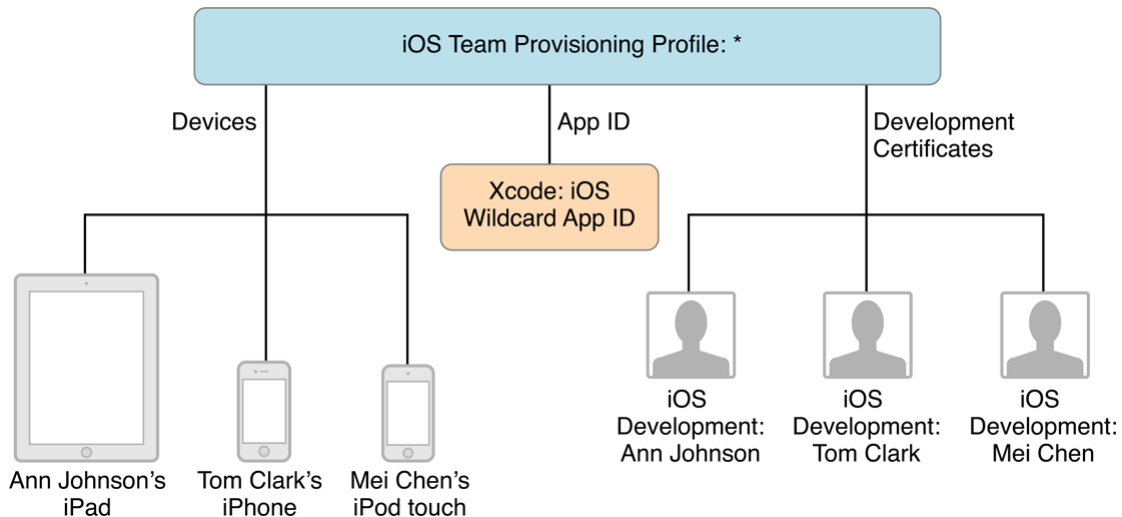
This chapter examines how the app was implemented, which technologies were used, how the app works and explanation of the final product. But first an explanation of how the app was developed for iOS and Android.

### 5.1 Android vs iPhone Development

The PhoneGap Build service packages the files into an app remotely and makes the files available for download as an ipa file for iOS or an APK file for Android. The app can readily be downloaded on to multiple Android devices but due to Apple's licensing, downloading onto iOS is more complicated.

In order to put the app onto any iOS device the developer needs a signing key which is linked with the app in PhoneGap Build. To generate this signing key, the developer first needs to set up a provisioning profile with Apple, which encompasses details of the specific developer, the specific target device and a Mac used in development. Only with these details in place can the app be downloaded on to an iOS device and it must be the one specified in the provisioning profile.

**Figure 27.** Diagram explaining the iOS provisioning profile

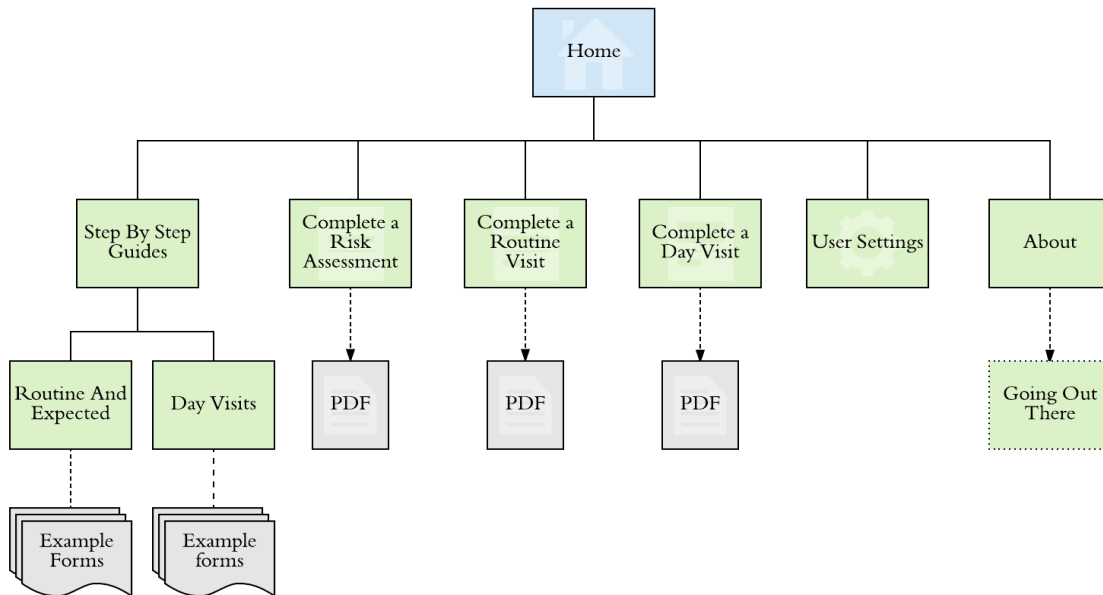


(Source: Apple Developer Guide [44])

For convenience, the development of the app was conducted on the author's personal Windows laptop and as a result the initial stages of development were carried out on Windows and tested on Android devices. By the end of sprint 3 it became more critical to test on iOS as well as Android, so a Mac was borrowed in order to create a provisioning profile and from that

point onward the app was developed and tested for both iOS and Android. The devices available for developing and testing the app were a Fusion 5 phone running Android version 5.1 (Lollipop), a Nexus tablet running Android version 5.0 and an iPhone SE running iOS 10.3.3.

## 5.2 Application Architecture



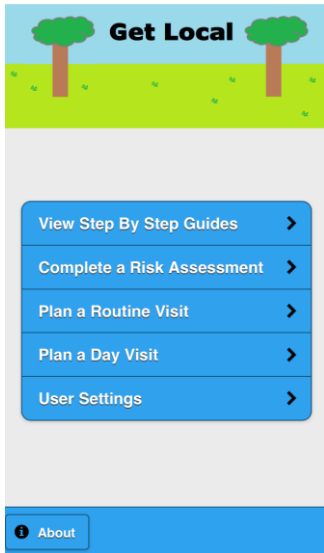
The Home page at the top is the first page of the app which appears when the app is opened. From here there are buttons to navigate to:

- 1) Step by step guides – the feature developed from the SAPOE flow charts
- 2) Complete a Risk Assessment – the function for generating a risk assessment.
- 3) Complete a Routine Visit – the function for generating a Routine Visit plan.
- 4) Complete a Day Visit – the function for generating a Day Visit plan.
- 5) User Settings – settings the user can specify such as the school logo.
- 6) The About page – information about SAPOE and a link to the Going Out There website.

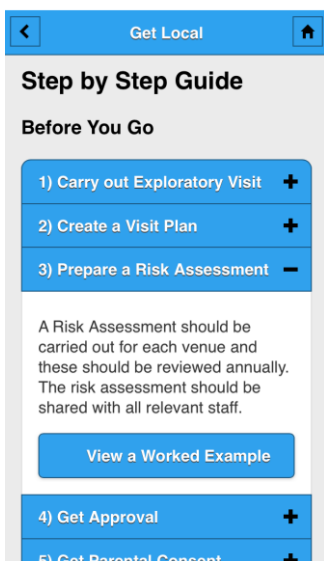
The next layer of the diagram illustrates where the user can navigate to from these pages. The dotted lines indicate that these features are handled outside of the app – in the case of the example forms, the PDFs are opened in an external PDF viewer, from which the user can return to the app. In the case of the forms generated by the app, these can be opened in an external PDF viewer or emailed from the device and in the case of the link to Going Out There, the website is opened in an in-app browser (provided the device has internet access).

## 5.3 Walk Through

### 5.4 Home Screen and Step by Step Guides.

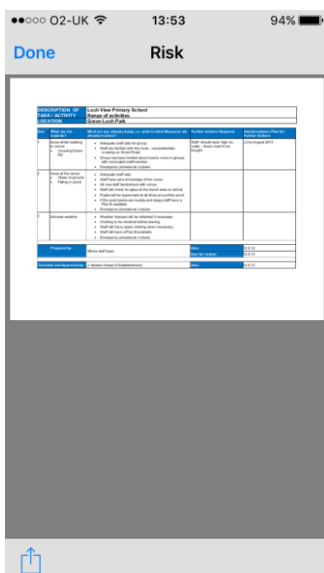


The Home screen is the first screen displayed to the user on opening the app. From here the user selects from the options available to navigate to the function they require. All other pages in the app link back to this page, its chief purpose is navigation around the app.



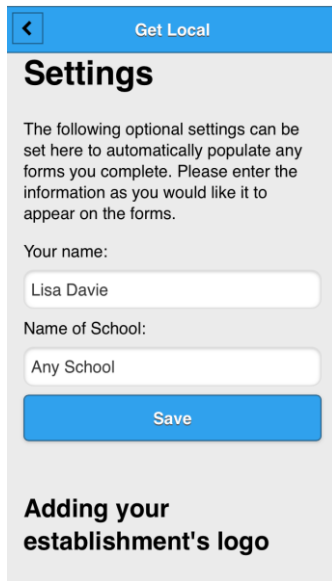
Two Step by Step Guides are available, one for Routine Visits and one for Day Visits. This screenshot shows how the guides are laid out. The steps from the SAPOE flowchart have been styled as collapsible blocks which can be expanded to give more information taken from the Going Out There document. This approach was used to keep the design as streamlined and uncluttered as possible. Where appropriate there are links to example documents which are viewed in an external PDF viewer.

Here is an example document displayed in the PDF viewer, the transition between the app and



the PDF viewer is seamless, so the user may be unaware that they have left the app. The display of example documents in this way was chosen because it allows the user to zoom in to the document, which is essential to read the information on a small screen.

## 5.4.1 User Settings and About Screen



**Settings**

The following optional settings can be set here to automatically populate any forms you complete. Please enter the information as you would like it to appear on the forms.

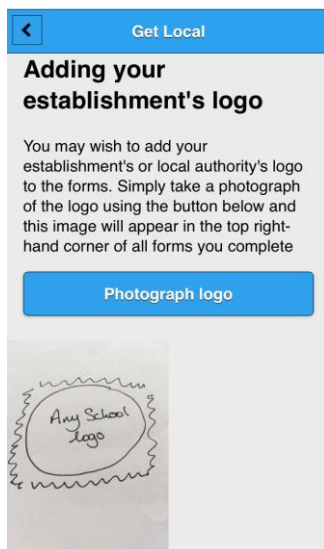
Your name:  
Lisa Davie

Name of School:  
Any School

Save

**Adding your establishment's logo**


The User Settings allow the user to set their name and the name of their school once and this information will be used to pre-populate these fields in any forms completed using the app.



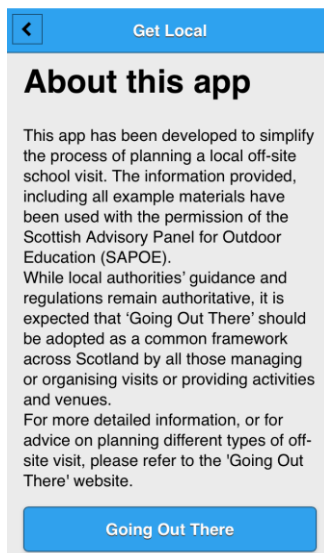
**Adding your establishment's logo**

You may wish to add your establishment's or local authority's logo to the forms. Simply take a photograph of the logo using the button below and this image will appear in the top right-hand corner of all forms you complete

Photograph logo



Also under User Settings is the option to add a logo for display on all documents. This function was added with the intention of making the forms more customisable and it was decided that the easiest way for the user to get the logo into the app would be to take a picture. Uploading a file was also considered, but this would be more complicated for the user and also make it more difficult to control the dimensions and the format of the image. If no logo is added, the default behaviour is to generate forms without a logo.



**About this app**

This app has been developed to simplify the process of planning a local off-site school visit. The information provided, including all example materials have been used with the permission of the Scottish Advisory Panel for Outdoor Education (SAPOE). While local authorities' guidance and regulations remain authoritative, it is expected that 'Going Out There' should be adopted as a common framework across Scotland by all those managing or organising visits or providing activities and venues. For more detailed information, or for advice on planning different types of off-site visit, please refer to the 'Going Out There' website.

Going Out There

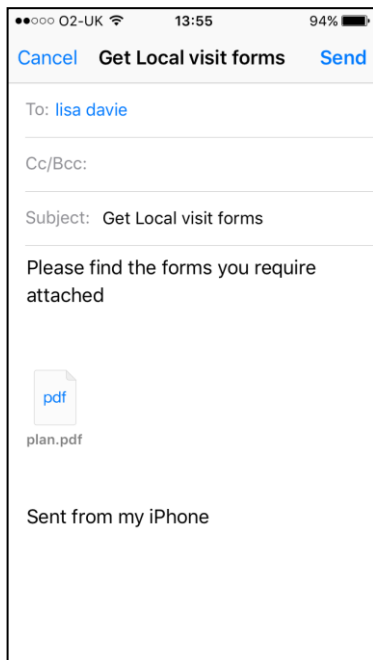
The About page was added to ensure adequate credit was given to the source of the information and to let the user know where they can get more information by including a link to the Going Out There website. The wording of the About section was developed in consultation with SAPOE.

## 5.4.2 Completing a Risk Assessment

The first page of the Risk Assessment asks the user to input details of the trip and select any hazards that apply, including the opportunity to enter their own risks. Note both these screenshots are from the same page, which can be scrolled up and down on a small screen, the header and footer are fixed so that the user can access the navigation buttons contained in them at any point.

The second page (left) asks the user to select details of control measures in place for each of the hazards selected on page 1, while the third page (right) gives the user the opportunity to add any further actions required. The Submit button processes the information on the form and opens the document in the PDF viewer for the user to preview. If any of the selections

need to be changed the user can exit the preview and go back and alter their selections before submitting a second time.



Once the User is happy with the document they can press send and an email application on the phone will open with the document attached. The user just needs to input the address they wish to send the document to. Once the user sends or cancels the email, they are returned to the app. If the device does not have an internet connection at the time, the email remains in the outbox of the email application.

Risk Assessment				
Name of School:	Any School			
Visit Type:	Routine and Expected Visit			
Visit Location:	Local Park			
Assessment prepared by:	Lisa Davie			
Date prepared:	20/09/17			
Visit Reference:	RA123			
Item	Potential Hazards	Control Measures	Further Actions Required	Implementation Plan for Further Actions
1	Issues with walking to the venue.	Adequate staff ratio. Ensure staff are familiar with the route. Set clear procedures for walking as a group.	Buy high vis jackets for staff to wear during outing	To be actioned by LD by 15/09/17
2	Issues at the venue.	Adequate staff ratio. Ensure staff have prior knowledge of the venue. Carry out a dynamic risk assessment.		
3	Adverse weather.	Obtain weather forecast if necessary.		
4	River running through the park	Advise all participants of the danger		
		Sign	Date	Review Due
Prepared by:				
Approved by:				

**Figure 29. Example Risk Assessment generated by the App**

The design of the form was based on the risk assessments provided in the toolkit, with similar layout and colour scheme applied.



### 5.4.3 Completing a Visit Plan

**Complete a Visit Plan for a Routine Visit**

Name of School:  
Any Schools

Visit Location:  
The Park

Prepared By:  
Lisa Davie

Reference Number:  
1234

The purpose of the Visit Plan is to record any decisions and training that has been carried out. It should give

**Give details for each item listed:**

Information to parents/Consent  
Obtained annually

Staff Visit to Venue  
Completed 26/08/17

Staff/Participant ratios  
1:8 - a total of 4 adults will be required

Getting to the venue  
On foot via the high street

Equipment required

The Routine Visit Plan and the Day Visit Plan functionality are handled using different forms because substantially more detail is required for a Day Visit Plan. However, both forms look very similar and the design is based on the Risk Assessment form for consistency. The main difference is that a Visit Plan requires more specific information and therefore text boxes are used for much of the user input.

Departure time:  
09:30

Return time:  
16:30

Reference Number:

The purpose of the Visit Plan is to record any decisions and training that

Clear Done

Weather

Emergency Procedures

External Provider (if applicable)

Risk Assessment Completed

Submit

Send


Reset

The relevant data type was set on the text box element and if supported by the browser features like the clock (above left) are automatically implemented to help the user input the

information. The date data type was not used for the input of the date, due to the many different formats that can be applied to dates and for this reason it was decided to let the user input the date manually in their preferred format. Once the form is completed the handling of the forms is exactly the same as the Risk Assessment. The submit button triggers the processing of the information and results in a preview, the send button opens an email with the document attached and the reset button resets the text boxes back to empty.

**Figure 30. Routine Visit Plan generated by the app**

### Visit Plan: Routine and Expected Visit



Name of School:	Any School
Visit to:	The Park
Plan prepared by:	Lisa Davie
Visit Reference:	VP1234


	Items to be Planned	Specific Details
1	Information to Parents/Consent	Obtained annually
2	Staff Visit to Venue	Carried out 23/08/17
3	Staff/Participant ratios	1:8, three adults will be required to supervise this trip
4	Getting to the Venue	On foot via the high street (approx. 10 minutes walk)
5	Equipment Required	Note books and pencils
6	Managing the Activity	Each adult will work with a group of 8 to complete planned tasks
7	Medical Needs	None reported
8	Weather	This trip will only go ahead in dry weather
9	Emergency Procedures	School procedures apply
10	External Provider (if applicable)	NA
11	Risk Assessment completed	This trip is covered by assessment RA123

Signed	Date
Prepared by:	
Approved by:	

As with the Risk Assessment, the design of the Visit Plans was modelled on the forms provided in the SAPOE toolkit.

**Figure 31. Day Visit Plan generated by the App**

### Visit Plan: Day Visit



Name of School:	Any School
Day Visit to:	Natural History Museum
Date:	29/09/17
Visit Leader:	Lisa Davie
Departure time:	09:30
Return time:	16:30
Visit Reference:	VP321

	Items to be Planned	Specific Details
1	Aims and Objectives of Visit	To tie in with our dinosaur theme by conducting activities planned around the dinosaur exhibits
2	Information to Parents/Consent	Covered by annual consent form. In addition an email will be sent out to all parents with details of the trip, timings and lunch requirements
3	Staff/Participant ratios	1:6 two members of staff and two parent helpers will be required
4	Venue	Natural History Museum
5	Travel/Transport	Local Coach Company has been provisionally booked for that date.
6	Participants	P4
7	Equipment required	Activity sheets, clipboards and pencils to be provided by the venue.
8	Managing the activity	Each adult will work with a group of 6 children.
9	Medical Needs	None recorded for this class
10	Weather	NA, this will be an indoor activity.
11	First Aid provision	A first aid kit will be carried by the visit leader. One first aider will be accompanying the group and first aid provision is available at the venue
12	Emergency Procedures	School emergency procedures apply.
13	External Provider	Natural History Museum has provided a suitable programme of activities and confirmed appropriate insurance is in place.
14	Risk Assessment Completed	This trip is covered by RA321

Signed	Date
Prepared by:	
Approved by:	

This concludes the section demonstrating the walk through of the application. The next section examines how this was achieved and why each technology was selected.

## 5.5 Implementing Key Functionality

As previously discussed, the content of the app was written in HTML, with CSS, JavaScript and jQuery mobile used to enhance the way the user interacts with the information. The more complicated features of the app, required the use of plugins to access native features of the

phone and external libraries to generate the PDFs in the browser. Selecting the right technologies was time consuming, the more options available the longer it took to evaluate the options and the use of community developed technology was particularly challenging due to poor documentation and support.

PhoneGap allows the use of plugin APIs to access native features of the device that would not otherwise be available to a web app. These plugins are in the form of external files which can be added to the project via the command line. A selection of standard Cordova Plugins are automatically included in the application folders when a new PhoneGap project is started, such as plugins for accessing the device orientation, geolocation and network information. It is up to the developer to remove any plugins they do not need.

In addition to these, there are many community developed, open source plugins available via the Node Package Manager repository (NPM). There is a great deal of variability in the quality of these plugins, level of documentation, and there is often no technical support for them, but they do allow developers to extend the features of their project far beyond the standard plugins alone and in a non-chargeable way.

As JavaScript is the programming language of PhoneGap, a further way to extend the app is by using JavaScript libraries. In particular, the use of UI libraries to enhance the appearance of the app seems to be standard practice, as the non-native appearance of PhoneGap applications is a recognised drawback.

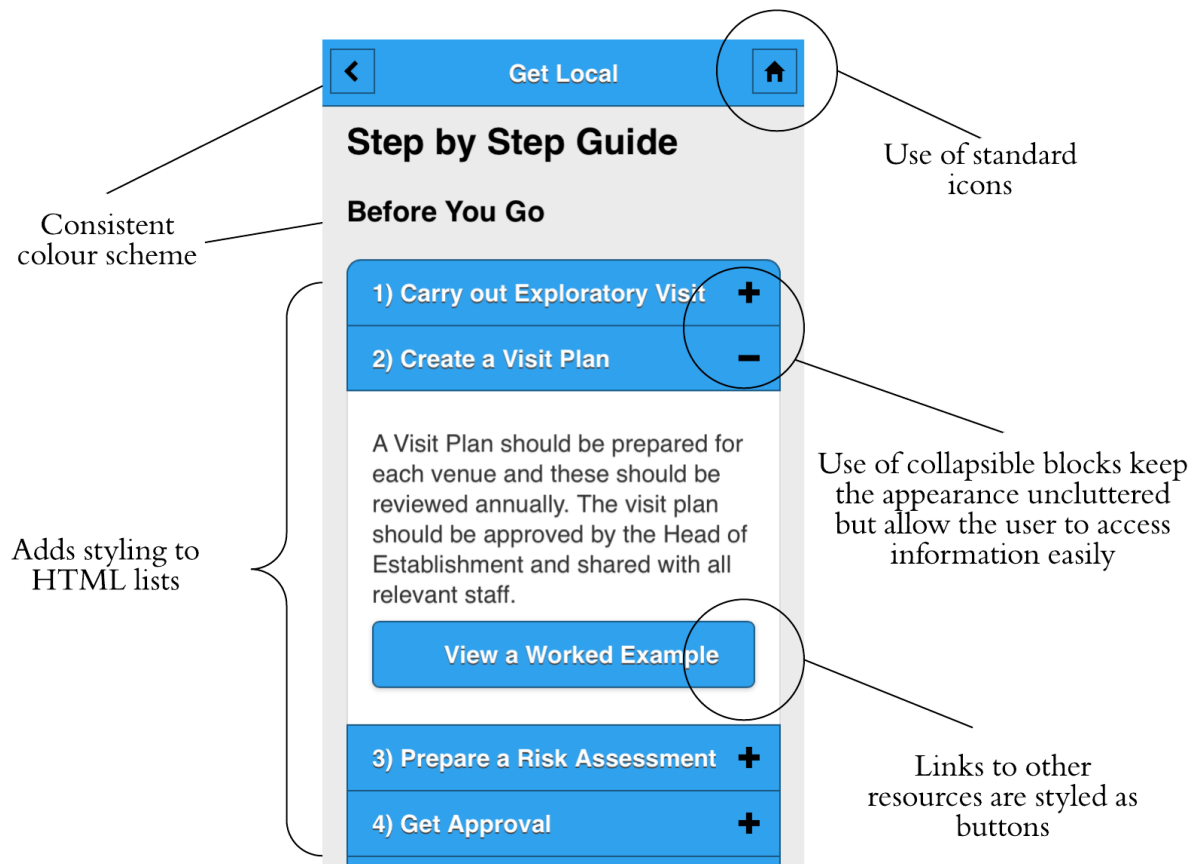
### **5.5.1 jQuery Mobile**

jQuery Mobile is a JavaScript library based on the well-established jQuery library, it was selected for use in this project because:

- It has good documentation, tutorials and community support.
- It is commonly supported by browsers for enhancing pages to give a consistent look and feel, but is still configurable using CSS.
- It is lightweight and modular so the developer only includes what they need.
- It is possible to include custom themes using the jQuery theme roller.

The basic features of the app, such as the content, visual appearance and layout were achieved using just HTML enhanced with CSS and jQuery mobile. In particular these tools were used to give a consistent style throughout the app and provide user friendly navigation of the material through buttons, icons and collapsible blocks. The main aim was to provide a familiar looking interface that would be intuitive to use.

**Figure 32. Diagram illustrating how jQuery mobile was applied**



The more complicated features of the app, required additional tools to implement them. The next section examines how these features were achieved and the challenges faced.

### **5.5.2 Access to native E-mail functionality**

As discussed in the requirements analysis, in order to offer the form-filling functionality there needed to be a straightforward method for getting the forms off the device. In discussions with the project supervisor, it was agreed that e-mailing the forms from the device would be the best option, as e-mail is widely available on smart phones; users are familiar with it and it gives the user control over where the form is sent. The implementation of this functionality was prioritised over the functionality to generate the documents, because it was identified as being potentially quite challenging.

There is no standard Cordova plugin for e-mail access, but there are open source community developed options available. These are maintained and documented on GitHub, which makes the code for the plugins available for other developers to branch and develop their own version, with the result that for most plugins there are multiple options available. The main requirement when selecting an e-mail plugin was that it should open an email on the user's

device, an attachment could be added in the form of a PDF and the email would successfully send.

A trial and error approach was applied to this and to other plugins in this project, until a solution that worked was found. Adding the first plugin was a steep learning curve and took approximately 10 days to implement. The main issues were:

- The quality of the documentation available – some documentation was ambiguous or incomplete and forum advice was often contradictory.
- Technical issues with the plugins – as they are community developed, they are less rigorously tested and robust than professional products.
- Level of experience – it was difficult to tell when the problem was associated with the plugin or when it was caused by a mistake made by the author. In many cases it turned out to be a combination of factors.
- Lack of useful debug information to point to what was causing the problem.

The problem was eventually solved using a newer fork of the email composer plugin which had been updated to deal with Android issues arising in the previous version. In addition, the plugin was added via the CLI, as some of the issues may have been caused by adding the plugins incorrectly. The process of working through the problem, although frustrating did provide important lessons which were applied to future difficulties, such as the value of perseverance and different ways to approach and break down the problem.

### **5.5.3 Viewing PDFs**

Viewing PDFs was identified as a requirement in order for the user to view example forms. Other options were briefly considered, such as displaying the original word documents or converting to HTML format, but neither of these worked in practice. The main advantage of using a PDF viewer is that the format remains consistent and is not dependent on the browser and most importantly, the user is able to pinch and zoom. Zoom in capability is disabled in PhoneGap apps and it was vital that the user could zoom in on example material in order to read the detail. Several options were identified for viewing a PDF:

- Safari includes PDF viewer functionality in iOS, however pinch and zoom is automatically disabled and there is no such option on Android.
- Google document viewer allows you to view PDFs online, however this approach did not satisfy the requirement that the app should be self-contained as it requires access to the internet. Additionally, the presentation of the documents was inferior to other PDF viewers.

- Six community plugins were identified which allow the app to access any PDF viewer installed on the device and each one was trialled. There were technical issues, with some of them causing the build to fail, therefore these were discounted. Efforts were concentrated on the two options which showed the most promise: Cordova-plugins-document-viewer and Cordova-plugins-open.

The document-viewer option worked as hoped, however it required the user to download a specific PDF viewer (Sitewaerts pdf viewer) and redirects the user to the app store if they do not have it installed. This was not considered as an acceptable option, as most users would not want to install this PDF viewer in order to use the app, so this option was also discounted.

Cordova-plugins-open was eventually implemented as the PDF viewer, however it took additional programming to make it work. Initially, the PDF viewer attempted to open the document but failed. After extensive research, it was discovered that an issue around access rights in newer versions of Android was preventing the plugin from accessing documents in the asset folder of the app. A work-around was implemented by using the Cordova file transfer plugin to move the files to an external directory of the app after it loads, where they can be accessed by the PDF reader.

This approach to the problem paid off later in the project when the requirement to generate documents was added. These needed to be saved in an external directory with read/write access, so the knowledge of the file systems of Android and iOS were applied here.

#### **5.5.4 Generating PDFs**

Functionality to generate PDFs was required for the form filling capability. One possibility was to take the information entered by the user, process it remotely on a server and return the resulting document. As discussed, one of the aims of the project was for the app to be self-contained, so the use of a remote server was not pursued.

The processing of the information provided by the user was all possible using JavaScript. For each form, the submit button calls a function to process the user input for that form type. In the case of the Risk Assessment, the function loops through the selections made by the user using `document.getElementById` for each form element and stores the information in a variable, or in the case of the Hazards selected, stores them in an array. For each Hazard selected a second function loops through the Control Measures and Further Action plans associated with each hazard and enters these into arrays. These arrays store the information input by the user ready for populating the document.

In order to actually generate the document, additional code was required and fortunately there is a JavaScript library available which will generate a PDF in the browser. jsPDF is an

open source JavaScript library for generating PDFs[3]. Like the community based plugins, the code for jsPDF is managed through GitHub and is open to community development, but unlike the majority of the plugins, it has a professional consultancy service behind it, called Parralax[43]. As a result of this, the quality of the product and the documentation available was of a much higher standard. Once the library has been added to the project the basic code to generate a PDF is as follows:

```
var doc = new jsPDF()           //Create a new pdf
doc.text('Hello world!', 10, 10) //Add text at 10mm x 10mm
doc.save('a4.pdf')              //Save the PDF as 'a4.pdf'
```

(Note: code taken directly from jsPDF GitHub page [3] with comments added by the author.)

There are options for setting the page format, document orientation and the units used, colour schemes, font and font size and the co-ordinates of any text. It should be noted that the doc.save() function opens the PDF in a PDF reader when running the library in a desktop browser, but not when running on a mobile device. For the purpose of this project the documents were saved in the external storage folder of the device and opened from there using a separate function.

jsPDF provided a simple but powerful tool for generating a PDF, however more work was needed to convert the user input into a useable form. The next tool added to the project was “jsPDF AutoTable”, a supplementary library to jsPDF, also available through GitHub [4]. This library provides a way for formatting the material into tables, with extensive options for setting colour, layout and styles to your tables.

The code for generating a simple table is as follows:

```
var doc = new jsPDF
var column = ["one", "two", "three"]
var row = [[ "A", "B", "C"], ["D", "E", "F"], etc]
doc.autotable(column, row, 10,10)
doc.save('table.pdf')
```

(Adapted from the AutoTable GitHub page [4]).

jsPDF AutoTable accepts columns in the form of an array of column titles and rows in a 2-dimensional array of row details. As a consequence, arrays were chosen to handle much of the information input by the user as these could be passed directly to jsPDF AutoTable to provide the content in each row of the table. JavaScript was also used to dynamically change the dis-



play of the Risk Assessment form, so that it only showed relevant options based on the choices the user had already selected.

No other credible alternatives were found that matched the quality and functionality of this approach. The choice to include these two libraries has enabled the app to produce quality PDFs of the forms within the browser, without the need to use a remote server.

### **5.5.5 Access to the Device's camera**

Access to the device's camera to add a logo to the User Settings is via a Cordova standard plugin and as such was straight forward to use with the documentation available. The `camera.getPicture` function opens the device camera, allows the user to take a picture, then closes the camera, restoring the application. The return value is sent to the success call back function for the developer to determine how it is used and there are options for determining the nature of the return value (e.g., the dimensions and quality of the image; the format as a file URI or base 64 encoded string).

The main technical issues came with passing the image to jsPDF in order to add it to documents. jsPDF is aimed at web developers rather than for mobile web apps and finding support for mobile related problems was often challenging. Problems arose because:

- The functionality to add an image is less well documented or supported than the basic functionality of the library.
- There are two methods for adding an image, passing the image as a base 64 encoded string or using the file URI. The second option was not well publicised and this led to a great deal of time being spent trying to add the image as a string unnecessarily.
- There were also issues around the asynchronous loading of the image in the browser which caused the image not to be displayed if handled incorrectly, which the author did not appreciate at the outset.

All of these issues needed to be addressed before the image was successfully added to the forms.

### **5.5.6 Standard Cordova Plugins**

In addition to the plugins added to the project for their specific functionality, the project initially included a set of standard plugins provided by PhoneGap as standard. Once the development of the app was complete all unnecessary plugins were removed, so as to keep the size of the app to a minimum by not including unnecessary files and because several of these features require the user to grant permissions, which may seem intrusive and excessive if they

are not required for the functioning of the app. The following table details the standard Cordova plugins which were retained in the project and briefly outlines the reasons that these were kept:

**Figure 33. Table of Standard Plugins included in the Project**

Plugin Name	Reason
Cordova-Plugin-File	Implements a file API for allowing read/write access to files on the device. This was used to access local storage on the device for saving user settings.
Cordova-Plugin-File-Transfer	Allows upload and download of files to the device. This API was used for transferring PDFs and saving new forms to the devices external storage.
Cordova-Plugin-Device	Used to access information about the device. Specifically, the device.platform function was used to return the platform of the device, to determine which file path to follow for file storage.
Cordova-Plugin-Inappbrowser	Opens a browser for the user to view resources on the web, without leaving the app. This function was used to include the link to the Going Out There website for the user to access further information
Cordova-Plugin-Splashscreen	Required to display and hide splash screens during application launch.
Cordova-Plugin-Statusbar	This plugin gives access to the status bar of the device and was included in order to remove the status bar entirely from the app. This allowed the header on each page to be streamlined, enhancing the back button so that it functioned more reliably on a small screen.

## 5.6 Visual Appearance

User experience was taken into consideration when designing the visual layout of pages. For example:

- the author endeavoured to give the app a clean, uncluttered appearance and also avoid excessive tabbing.
- Navigation of the app was designed such that the user can get back to the Home screen from any other screen in the app by using a back button or the home button.
- To make the app intuitive to use, options are designed to be self-explanatory and nested in a logical way.
- Colour scheme and styling was used to give the app a consistent look and feel throughout.

The inclusion of a banner on the home screen was initially to illustrate to the client what was possible in terms of the visual design. Although basic, the client liked the banner as the graphics are in keeping with other apps aimed at school teachers and so it was kept. If the app had been developed for an organisation the banner could be used to display the logo of the organisation, however in this project the banner was used to make the home page more visibly interesting and a part of the banner was reused as the app icon, which gave a visual theme to the app.

**Figure 34. The Home Page Banner and App Icon**



### 5.6.1 Icon and splash screen

It is necessary for the app to have a unique icon and splash screen, the icon gives the user a quick and easy way of identifying the app and if designed well, it can give the app a polished and professional finish. As discussed above, the icon was taken from the banner displayed on the Home page of the app, as a consistent and recognisable image that the user will associate with the app.

The Splash screen is a launch screen used to give the impression that the app is fast and responsive by appearing briefly between the app opening and the first page. Most splash screens are not very complicated and according to Apple's developer guidelines [45]:

*'The launch screen is not an opportunity for artistic expression. It is solely intended to enhance the perception of your app as quick to launch and immediately ready for use'*

Therefore, a very basic splash screen was designed, merely showing a green screen to tie in with the theme of outdoor learning and local green space.

## 5.7 Unsolved Problems

A further feature which was planned for the app was to save documents to the device and allow users to go back and modify past assessments, for updating them or planning a similar trip. PDFs are not editable by design, but a solution to this was considered, whereby the user's selections could be stored alongside the PDF and these selections would be used to populate the HTML form, then the user would be taken back through the form with the option change the selections. Although this would require a fair amount of code, it is likely to be a viable way of approaching the problem. Furthermore, the framework for saving the documentation is already in place in the file system of the device, as all documents are saved before they are displayed. The main problem encountered in implementing the feature was the design of the user interface for accessing the files.

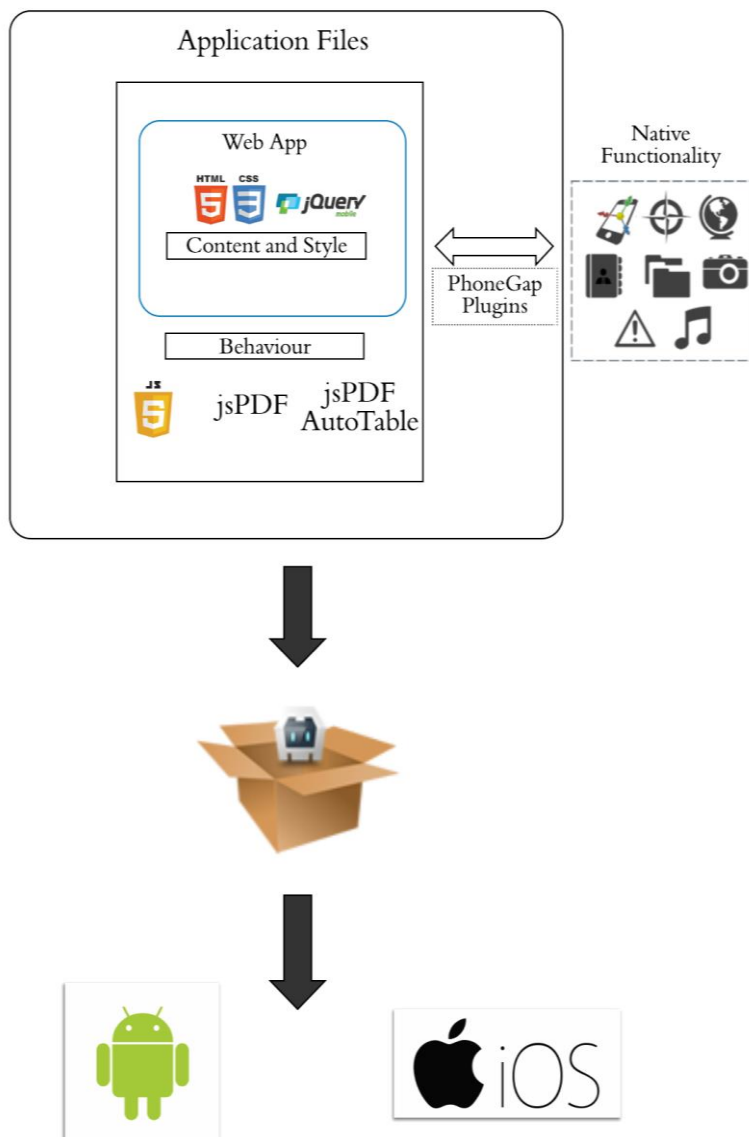
Initial attempts involved writing a piece of code which dynamically created a button on a "Saved Documents" page when a document was saved. A function was also created to link the button to the document so that it would be viewed when the button was pressed. However, two problems arose with this approach. Firstly, this only allowed the user to *view* the document, any other interaction such as emailing the document required the creation of another button and this had the potential of causing the user interface to look untidy, with a long list of buttons. Furthermore, when subsequent documents were saved, there were issues creating the elements to display these additional buttons and they were not styled correctly on the page.

All of this was taking place at the point in the project when it was important to start getting user feedback. As the amount of code required to implement the Saved Documents feature properly was expected to be significant, the decision was taken to concentrate on getting adequate user feedback and on writing up the project rather than pursuing this feature any further.

## 5.8 Internal Structure

As detailed in the State of the Art in section 2.2.2, PhoneGap works by creating a web application, with access to native APIs, wrapped in native code to produce a binary application archive. This can be installed on to devices in the same way you would install a native app. Figure 35 provides a diagrammatic explanation of this, which was adapted from the blog "PhoneGap Explained Visually"[41].

**Figure 35. Overview of the Application Structure**



The application files are the HTML, CSS and JavaScript files created by the developer, that determine what the app looks like, how it behaves and all the factors that make the app unique. These are referred to by PhoneGap as the project's web assets. On top of these are any plugins added to the project which are accessed via JavaScript and these will communicate with the native features of the device once the app is installed. When the project is built, the web assets and the plugins are packaged up with auxiliary PhoneGap files to produce the relevant application binaries for whichever platforms are being targeted. As a result, a PhoneGap project folder will contain several files for carrying out these functions. Below is a brief overview of the major files or folders included in this project and their purpose:

**Figure 36. Project Files**

Folder / File	Purpose
Config.xml	Contains all the information for configuring the app, e.g. App name, ID and author as well as security settings, plugins to include and platform specifications.
iOS / Android folders	Platform specific files for implementing the app.
Plugins	A folder containing the plugin files added to the project via the config.xml. Any plugins listed in the config file are fetched and added to the project at build time.
Asset Files	Purpose
www/CSS	The external CSS files which specify the styling to be applied to the index.html document.
www/Imgs	Any images used in the app, in this project the banner for the home screen is located here
www/js	A folder containing the JavaScript libraries and the external JavaScript file (app.js) for the functions called in index.html. These functions were written specifically for the project by the author and control the behaviour of the app.
www/PDFs	A folder created to hold the example documents in the form of PDFs. Note that these files are moved to device storage outside of the app when the app loads, in order to be accessible by the PDF reader.
www/index	A single HTML page with different sections marked up as pages covering every page of the application. The single file approach allows smooth page transitions.

## 6 Testing and Feedback

Testing of the app happened in the following ways:

- Continual functional testing of the app, as each feature was added, with any bugs fixed as they were found.
- Demonstration to the client at regular intervals
- End User Feedback during the development process
- Functional Testing against a set of User stories at the end of development, to check the app behaved as expected in a set of common scenarios.
- Feedback from major stakeholders gathered at the end of the process

### 6.1 End User Feedback during Development

End user feedback was sought during the development process as it was recognised as part of the Agile approach that this would be one way of ensuring the final product would be really usable. This feedback was gathered in two ways – informally through discussions (via the client or through personal contacts) and formally through end user testing in conjunction with a questionnaire. This was performed in June when a working model of the app with most of the main features was first available.

Participants were recruited by the client, by asking teachers staying at the Low Port Centre on residential trips if they would be willing to test the app on an Android phone provided to them for the purpose and complete a questionnaire on their experience.

The questionnaire gave the user 3 tasks to complete, reflecting common scenarios which the user might need to perform, then they were asked to give feedback on whether they were able to complete the task, any error messages they encountered and to rate how easy the task was to perform. No guidance was given on how to complete the tasks. Furthermore, they were asked if there were any additional features they would like to see, to rate their overall experience and for additional comments.

In total, the client approached teachers from four groups staying at the Centre over a two-week period in June. The timing of the testing was not ideal as the end of the school year is a notoriously busy time for teachers. Although teachers from all the groups showed an interest in testing the app, time constraints and other factors meant that only one teacher was able to successfully carry out a test and complete a questionnaire. The results of that feedback were as follows:

**Figure 37. Initial User Feedback**

Task	Feedback
1) Look up what to do <i>after</i> completing a Routine and Expected Visit	Completed No error messages. Ease of use: 5/5 – very easy.
2) Choose a type of visit then find a worked example of a Risk Assessment for that visit.	Completed No error messages. Ease of use: 5/5 – very easy.
3) Complete a Risk Assessment and send it via Email	Completed Received an error message stating “PDF not formatted” and only 1 of 4 pages were displayed Ease of use: 5/5 and “intuitive to use”

Overall experience was rated as 5/5 for quality of information, ease of use and visual design and further comments stated that the app was:

- User friendly.
- The step by step guides are detailed and clear.
- Most considerations taken into account in the worked examples.
- Risk assessment produced by the app was succinct and to the point.
- Suggestion that the school name, date and person filling in the form should be added.

Based on this feedback, the form filling features were further developed to include the school name, date and name of the person completing the form and the Risk Assessment included a step for the user to add additional risks and controls.

The error message that the user received when previewing the form was caused by the app trying to write over the previous PDF and open it before this step was complete. This happened when a user went back to change some of their selections and then tried to preview the document a second time. As a result of the feedback, the way the files are saved was changed, so that when any alterations are made to a document, it is saved under a different file name and this removed the problem.



Although it would have been beneficial to get more users to test the app while it was still in development, the start of the school holidays in July meant that the client was not in contact with any more teachers for the rest of the development period. However, the user feedback from just one end user was incredibly useful and allowed at least some improvements to be identified.

## 6.2 Functional Testing Against a set of Common Scenarios

Three sets of scenarios were formulated as common user interactions with the app. The order of the steps was important, to test how the app behaves under different circumstances (e.g. with user settings, without user settings, the first time it is used and subsequent times), to check that it continues to behave as expected. Although three scenarios cannot cover every eventuality, the most common uses of the app were identified and tested. This approach was adopted in an attempt to be methodical in the testing and to ensure that no obvious bugs were overlooked. An example on one of these scenarios is given in Figure 38.

**Figure 38. Example Scenario for Testing.**

Scenario	Successfully completed
1) Set User Name, School Name and Logo	Yes
2) View example documents for routine and expected visits	Yes
3) Complete a risk assessment for a routine and expected visit and preview the document	Yes
4) Go back and change some of the selections then preview the document	Yes
5) Email the document to any email address	Yes
6) Complete a routine visit plan and preview the document	Yes
7) Go back and change some selections then preview the document	Yes
8) Email the document to any email address	Yes

### 6.3 User Feedback on the Final Product

At the end of the project, a total of 12 teachers were approached to see if they would be willing to test the app and provide a paragraph of feedback based on their experience, highlighting the strengths and weaknesses of the app. These teachers were identified as colleagues of the client, contacts of a representative from SAPOE and personal contacts of the author. A cross section of nursery, primary and secondary level teachers were approached over two different local authority areas. Again, timing was not ideal as this took place at the end of August and start of September in the first few weeks of the new school year, when many teachers are quite busy.

A total of 4 teachers responded and again they were provided with a mobile phone for the purposes of testing the app. The following positive comments were received:

- Easy, quick and very user friendly.
- Makes you think through the stages of planning a trip and I like that it generates a risk assessment and forms.
- Easy to change your selections when generating a risk assessment.
- Easy to use and offers a wide range of options to satisfy my school's requirements.
- The Visit Planner is well thought through and offers the options I need.
- I would find this useful as a class teacher.

The following critical feedback was received:

- It should cover Child Protection considerations in the Step by Step guides and Risk Assessment.
- Risk Assessment feature could be expanded to cover more scenarios.
- It would be good to customise the Risk Assessment to the same forms used by my local authority.
- It would be useful as a desktop app or webpage as I complete these forms at my desk. However, it would be useful to store the completed forms in the app to have them portable and readily available on a trip.

There are some drawbacks in the way that the testing was carried out. Only receiving feedback from four respondents was a little disappointing and a greater number of respondents

would have provided a more rigorous result. Also, the testers all tested the app on a phone provided to them as the app was only downloadable to Android phones. Apple does offer a tool called “Test Flight” which allows the app to be distributed for testing purposes, however Apple approval of the app is required in order to use Test Flight and as such time did not permit for this tool to be used. In an ideal situation, the app should have been tested on multiple devices and information about how it performed on these devices gathered and analysed to identify any functional problems, in addition to the feedback on the users’ experience.

Furthermore, only one of the respondents reported using the app whilst actually planning a trip, the others did not have any trips in the planning stages at the time of testing and therefore their feedback was in some ways hypothetical rather than the result of using the app in a genuine setting.

#### **6.4 Feedback from SAPOE**

It was decided early on that the logical owners of the app at the end of the project would be SAPOE, given that it was their material which had been used. In the very least the author wanted to get SAPOE’s approval of the app before it was distributed for any widespread use. The option to present the app at the next panel meeting was discussed, but as a first step the app was demonstrated to one of the members of SAPOE.

The representative was very interested in the work that had been carried out in the project and was pleased that we had chosen to use the Going Out There materials rather than the potentially easier option of focussing on local practices. Overall, she was impressed with the capabilities of the app, seeing real potential for its future application.

Nonetheless, one issue that requires further consideration is that Going Out There provides best practice guidelines, but the responsibility for how these are implemented lies with the local authorities. While it was hoped that practices would become standardised across Scotland, at present only a selection of local authorities use Going Out There as their framework, while many other authorities continue to provide their own sets of guidelines, systems and forms, which makes it difficult to provide a “one size fits all” solution.

Despite differences in local authority practice, the representative thought it would be possible to develop the app further into a companion to the website or potentially to tailor the app to any local authority who was interested in using it. She agreed to investigate these possibilities further at the next panel meeting and would seek funding for the further development of the app from Education Scotland.

## 6.5 Feedback from the Client

The opinion of the client at the end of any software engineering project is an important measure of its success. The feedback received from the client is provided here in its entirety:



## Outdoor Learning Team

### “Get Local” Outdoor App: Lisa Davie, Stirling University

Project Details			
Lead Officer(s)	Mike Harvey/Lisa Davie(Stirling University)	Start	Sept 2016
Name	Get Local App development	End	June 2017
Frequency	N/A	Duration	Sch Yr
Delivery	Develop a mobile application which allows school teachers or school management team to plan class excursions and produce risk assessments.		
Inclusivity	Teachers & All Ed Users		
Cost	N/A		

### Report

#### Summary

In our discussions with teachers and educational staff they often complain about the endless paperwork and “barriers” to getting children outdoors into the local community and greenspaces. Much of this relates to the complex “risk assessment” paperwork and the formal documentation required by the local authority health and safety teams.

This App, developed by Lisa Davie at Stirling University computer department, very neatly solves these problems in a well-designed and reliable little package.

It is quite simply a excellent and functional tool for local authority education professionals.

#### Key Project Points:

- Lisa listened very carefully to our requirements and designed an App that met these requirements. She did what we asked and exceeded our expectations.
- At all the meetings we had, Lisa was always professional, on-time and followed-up the action points in a timely manner.
- The simple and functional content navigation within the App is excellent. In other words, a complex and confusing process is now straightforward.
- Risk assessments are generated very simply and quickly.
- The App is fast and early “bugs” were identified and corrected.
- Our Staff and the teachers we spoke to thought it was an excellent tool: everyone wanted to know where they could download it!

We hope the excellent work on this project continues with Education Scotland and SAPOE (the Scottish Advisory Panel For Outdoor Education) who are currently investigating its application across all local authorities within Scotland.

## **6.6 Summary of Testing**

The app was tested in a variety of ways, to ensure it functioned as expected and to ensure that it met the requirements of the end user. Feedback was obtained from each of the stakeholders in the project and was generally very positive. The feedback from teachers, included some important suggestions for how the app could be improved, which will help inform the section on Further Work. All this feedback will be taken into consideration in the evaluation of the project in the following chapter.








## 7 Conclusion

This chapter evaluates the successes and failings of the project and looks at further work that could extend this work in the future.

### 7.1 Evaluation of the App

While the project is complete, the app still has the potential for further development. As it is, the app is not ready for distribution to the general public because the feedback suggests there is still more to do, it has not yet been approved by SAPOE and has not received adequate testing (either by end users or on sufficient different devices). However, it was never explicitly stated at the outset that the app should be ready to distribute at the end of the project, although this was obviously the long-term aim for the application. In terms of the project criteria the app will be evaluated against what was set out to do at the beginning.

The following list of acceptance criteria were identified during the requirements analysis in section 3.4, the criteria which were met in the project are highlighted below.

- |  |   |
|--|---|
| 1) The app is available on Android and iOS                         |  |
| 2) The user can view guides for planning a local visit             |  |
| 3) The user can view examples of how the form should be filled out |  |
| 4) The user can complete the forms they require using the app      |  |
| 5) The user can preview and access the forms once completed        |  |
| 6) The user can customise the app with user settings               |  |
| 7) The user can save and retrieve previous documents               |  |

As the list shows, all the requirements but one were met. As discussed in section 5.7, the requirement to save and retrieve documents was not successfully implemented before the end of the project. This feature would be useful, as one end user pointed out, so that risk assessments and visit plans could be accessed easily whilst on a trip, however it was felt that implementing this feature poorly would detract from the quality of the rest of the app and therefore it was better to put the feature down as “Further Work” than try to implement it in a hurry. Moreover, this was one of the last requirements identified in the project, as a feature which would give the app a professional finish, rather than one of the essential requirements set out by the

client. Since the app is not currently in a position to be distributed, the absence of a Saved Documents function is not a significant failing of the project.

From a technical point of view, the main achievements of the project were succeeding in developing a functioning app for iOS and Android, successfully accessing several native features of the phone and achieving the form processing using client-side technologies to negate the need for a server. The author's main regret about the project is that it was not possible to get more user input before the app was complete. Feedback about the types of hazards and controls to offer in the app and how the forms are filled-in in practice would have helped to develop a more useful final product. Furthermore, an understanding of how Going Out There is implemented by local authorities and in particular the use of specific forms and the systems in place around them would have been beneficial at the outset, although this may not have changed the outcome of the project.

In general, the end user feedback was very positive, the app succeeded in being user friendly and simplifying the process of planning an off-site visit, users liked the features the app offered and there seems to be genuine interest from teachers in using a tool like this. In particular the functionality to generate the forms through the app was very well received as this would genuinely save teachers time. The client also seemed very happy with the final product, stating that their expectations had been exceeded and the representative from SAPOE was impressed enough to want to take the project further.

As the app is, the main barrier to its uptake is whether establishments would accept the forms generated by the app and whether the PDF format is compatible with the way that establishments handle these assessments. The user settings attempted to address this issue, by allowing users to customise their forms with an establishment logo, but unfortunately this is probably not adequate. This issue highlights that the problem we set out to solve was more complicated than the author appreciated at the outset.

Further features were identified in the user feedback which would make the app more useful, and these are discussed in section 7.4 on Further Work. The author feels that the technical challenges in this project were met and whilst the app is not a finished product, it does go a long way toward solving the problem that was set out at the beginning.

## **7.2 Evaluation of PhoneGap**

The choice of PhoneGap as a framework was not a limiting factor in the project. Initial concerns that user experience would suffer as a result were not borne out by the user feedback, with consistently positive comments about its user friendliness and visual design and critical comments focussing more on the content of the app. Furthermore, PhoneGap is best suited to

developing simple applications and imposing a level of simplicity may actually be an advantage to many applications.

The use of PhoneGap allowed the objective of developing for both Android and iOS to be achieved and a near complete app was produced within a short time frame, with only limited experience at the outset. The failure to implement a saved documents feature, was in part due to limitations associated with PhoneGap as the difficulties arose in designing a suitable user interface for retrieving the documents using HTML, CSS and jQuery mobile, however given more time an acceptable solution may have been found.

One major problem that was encountered during development was the availability of support for third party plugins and in particular finding up to date information, as many plugins and the resources used to trouble shoot the problems were at least 3 years old. This led the author to feel that the use of PhoneGap may be in decline and that some of the newer options such as Ionic and React Native should have been considered at the outset of the project. Overall, it has been a positive experience using PhoneGap. The principle of using a cross-platform framework offers plenty of advantages over native development, although they currently do not match the quality of native applications, the author looks forward to seeing how this field develops in the future.

### **7.3 Evaluation of the Agile Approach**

The main Agile principles adopted in the project were an iterative approach and stakeholder engagement. The iterative approach to development had advantages in that:

- It allowed the app to evolve as different requirements were proposed.
- It gave structure to the development process with the focus on one feature at a time and meetings acting as deadlines for milestones to be reached.
- It allowed work to start without a clear statement of what the end product should look like.

However, the lack of clear scope and objectives at the outset may also have been a limiting factor as much time was wasted looking into different possibilities and a clearer idea of the end product would have been beneficial at times during the app's development.

Stakeholder engagement was also very beneficial, but it was not implemented as successfully as it could have been in this project. Firstly, despite best efforts, access to teaching staff during the development stage was limited and secondly more engagement from SAPOE could have been sought at the outset rather than at the end of the project. In hindsight, a lack of domain knowledge on the part of the author was a problem and this could have been avoided



through better stakeholder engagement, although it should be remembered that this was a student project and getting access to the right people with the right knowledge was not an easy task.

## **7.4 Further Work**

Based on all the feedback, it is the author's opinion that the app has real potential to be developed into a useable product, either as an app for a local authority, or as a companion app to the Going Out There website. As it stands, the app is a good prototype or proof of concept model which could be used as a starting point for another round of development. If the project is taken further the author would make the following recommendations:

- Use a focussed plan for the features, covering specific details of what should be included in the app.
- Get engagement from teachers and local authority representatives from the outset to define what their requirements are, for example using the app to generate specific forms as issued by each authority.
- Expand the options to be covered in the Risk Assessment to include a wider range of scenarios, such as advice on Child Protection.
- Develop the Saved Documents feature to allow users access to the forms they need while on a visit.
- Consider adapting the app for use on a desktop or as a web page – both of these could be possible with adaptations to the existing code.
- Consider how signatures could be applied to the documents generated by the app (i.e., use of digital signatures if the document is not going to be printed off).
- Consider the visual design and layout, for example the banner and app logo could be replaced by the SAPOE logo or a local authority logo to give a more professional finish.

The technical design of the app has advantages in its simplicity, the fact that it is self-contained makes it cheaper to maintain as there is no back-end server to process the information, no user accounts or databases to store these. While a more complicated system might offer extra functionality, it would start to intersect with the functions offered by EVOLVE and there is no benefit in developing a system which does the same thing as a product already on the market.

## 7.5 Summary

This project succeeded in producing a tool which addresses the problems faced by teachers planning an off-site school visit, by putting the relevant information in to a straightforward, easy to access format and giving them a way of completing a risk assessment and a visit plan quickly on their personal phone. The resulting app exceeded the expectations of the client and impressed the representative from SAPOE and the technical achievements of the project should not be undervalued.

At the start of the project it was hoped that the end result would be an app store ready application which could be distributed to teachers all around Scotland, but unfortunately this was not achieved in the timescale of the project. The main failing was the over simplification of a complex problem and failure to research and understand the current situation at the outset. Given the author's understanding of that situation now, it seems unlikely that an app suitable for all local authorities would have been achievable within the constraints of this project, however by working more closely with SAPOE or a local authority, something usable may well be possible in the future.

It is very much hoped that the work carried out in this project will be taken further and that by demonstrating what is possible to SAPOE, local authorities and Education Scotland, the app has succeeded in opening a dialogue about how teachers could be helped to plan off-site visits using technology. By helping to remove the barriers, this may in turn help more children to access the many benefits of Outdoor Education in their local environment.

## 8 References

- [1] Going Out There. <http://www.goingoutthere.co.uk/> [Accessed September 2017]
- [2] jQuery Mobile. <http://jquerymobile.com/> [Accessed September 2017]
- [3] GitHub. <https://github.com/MrRio/jsPDF> [Accessed September 2017]
- [4] GitHub. <https://github.com/simonbengtsson/jsPDF-AutoTable> [Accessed September 2017]
- [5] Camera - Apache Cordova. <https://cordova.apache.org/docs/en/latest/reference/cordova-plugin-camera/> [Accessed September 2017]
- [6] Cordova-plugin-email. <https://www.npmjs.com/package/cordova-plugin-email> [Accessed September 2017]
- [7] Cordova-open. <https://github.com/disusered/cordova-open> [Accessed September 2017]
- [8] Plugin Search - Apache Cordova. <https://cordova.apache.org/plugins/> [Accessed September 2017]
- [9] Waite,S., Passy, R., Gilchrist, M., Hunt, A. & Blackwell, I. 2016. *Natural Connections Demonstration Project, 2012-2016: Final Report*. Natural England Commissioned Reports, Number215.
- [10] Ofsted, “Learning Outside the Classroom, how far should you go?”, October 2008 reference no. 070219
- [11] Nicol, R. et al, “Outdoor education in Scotland: A summary of recent research”, 2007, Scottish National Heritage
- [12] Learning and Teaching Scotland, “Curriculum for Excellence through Outdoor Learning”, 2010
- [13] Grant, D. “From Lyme Bay to licensing to de-regulation? The current state of safety within the outdoor activities sector”, March, 2013, Rural Policy Centre Policy Briefing
- [14] What We Do – SAPOE. <http://www.sapoe.org.uk/about-us/what-we-do#rt-header> [Accessed September 2017]
- [15] Risk management: Health and Safety checklist for teaching and support staff in classrooms. <http://www.hse.gov.uk/risk/classroom-checklist.htm>. [Accessed August 2017].
- [16] .Mid Lothian Out of School Visits. [https://evolve.edufocus.co.uk/evco/assets/midlothian/5\\_5\\_out\\_of\\_school\\_visits.pdf](https://evolve.edufocus.co.uk/evco/assets/midlothian/5_5_out_of_school_visits.pdf). [Accessed August 2017].
- [17] EVOLVE. <http://www.edufocus.co.uk/pages/evolve/visits.asp> [Accessed August 2017].
- [18] EVOLVE promotional material. [http://www.offsite-education.co.uk/doc/EVOLVE\\_outline\\_specification.pdf](http://www.offsite-education.co.uk/doc/EVOLVE_outline_specification.pdf) [Accessed August 2017].

- [19] EVOLVE - Choose Service.  
[https://evolve.edufocus.co.uk/evco10/evchome\\_public.asp?domain=](https://evolve.edufocus.co.uk/evco10/evchome_public.asp?domain=) [Accessed August 2017].
- [20] Scottish local authorities. <http://www.gov.scot/Topics/Government/local-government/localg/usefullinks> [Accessed August 2017].
- [21] Parentmail. <https://www.parentmail.co.uk/> [Accessed August 2017].
- [22] Class Dojo. <https://www.classdojo.com/en-gb/?redirect=true> [Accessed August 2017].
- [23] CURRICULUM APP - Toisc Limited. <http://www.toisc.com/toisc-apps/curriculumforscotland> [Accessed September 2017]
- [24] Gartner. <http://www.gartner.com/newsroom/id/3725117> [Accessed August 2017]
- [25] Ohrt, J. & Turau, V. "Cross-platform development tools for smartphone applications." *Computer*, 45 pp.72-79 2012
- [26] Native, Web or Hybrid Apps? What's the Difference?  
<https://www.mobiloud.com/blog/web-hybrid-native-apps/> [Accessed August 2017]
- [27] Xamarin. <https://www.xamarin.com/> [Accessed August 2017]
- [28] G2 Crowd Xamarin reviews. <https://www.g2crowd.com/products/xamarin/reviews> [Accessed September 2017]
- [29] G2 Crowd PhoneGap reviews. <https://www.g2crowd.com/products/phonegap/reviews> [Accessed September 2017]
- [30] G2 Crowd Appcelerator reviews.  
<https://www.g2crowd.com/products/appcelerator/reviews> [Accessed September]
- [31] Xamarin for all. <https://blog.xamarin.com/xamarin-for-all/> [August 2017]
- [32] PhoneGap Apps. <http://www.adobe.com/devnet/phonegap/articles/phonegap-apps-powered-by-developercom.html> [Accessed September 2017]
- [33] Appcelerator. <http://www.appcelerator.com/mobile-app-development-products/> [Accessed September 2017]
- [34] Appcelerator Blog. <http://www.appcelerator.com/blog/2013/07/but-i-thought-titanium-was-cross-platform/> [Accessed August 2017]
- [35] Agile Software Development Basics and fundamentals – CodeProject.  
<https://www.codeproject.com/Articles/1064114/Agile-Software-Development-Basics?msg=5177236> [Accessed August 2017]
- [36] PhoneGap Going Further. [PhoneGap.com/gettingstarted/5-going-further](http://PhoneGap.com/gettingstarted/5-going-further) [Accessed August 2017]
- [37] PhoneGap Docs. <http://docs.phonegap.com/references/desktop-app/pair-with-dev-app/>[Accessed August 2017]
- [38] PhoneGap Getting Started. <http://docs.phonegap.com/getting-started/1-install-phonegap/cli/>[Accessed September 2017]

- [39] PhoneGap Products. <https://phonegap.com/products/>[Accessed September 2017]
- [40] PhoneGap Build. <https://build.phonegap.com/> [Accessed September 2017]
- [41] Trice, A. “PhoneGap Explained Visually” 02 May 2012  
<https://phonegap.com/blog/2012/05/02/phonegap-explained-visually> [Accessed September 2017]
- [42] The Last Word on Cordova and PhoneGap. <http://blog.ionic.io/what-is-cordova-phonegap/> [Accessed September 2017]
- [43] Parallax. <https://parall.ax/> [Accessed September 2017]
- [44] Apple Developer Guide.  
[https://developer.apple.com/library/content/documentation/IDEs/Conceptual/AppDistributionGuide/LaunchingYourApponDevices/LaunchingYourApponDevices.html#//apple\\_ref/doc/uid/TP40012582-CH27-SW1](https://developer.apple.com/library/content/documentation/IDEs/Conceptual/AppDistributionGuide/LaunchingYourApponDevices/LaunchingYourApponDevices.html#//apple_ref/doc/uid/TP40012582-CH27-SW1)[Accessed September 2017]
- [45] iOS Human Interface Guidelines. <https://developer.apple.com/ios/human-interface-guidelines/graphics/launch-screen> [Accessed July 2017]