Abstract

What is MyFestival? Here you’ll find a list of festivals/events at the touch of your fingertips. All you have to do is to search on the website or download our app.

Project Report



Patrick Melia

PRJ400

Vivion Kinsella

B.Sc. in Computing (Software Development) Level 8 Year 4

# Acknowledgements

I would like to thank my supervisor Vivion Kinsella, who accepted to be my supervisor from day one and who invested quite a lot of time overseeing my progress for the project over the last 9 months. I would also like to thank all the lectures I talked to about this project over the duration of the last year and who gave me feedback on a few areas of the project.

Table of Contents

[Acknowledgements 1](#_Toc401138433)

[Detailed Description of the problem to be solved: 5](#_Toc401138434)

[Project Description: 5](#_Toc401138435)

[Festival Web App: 5](#_Toc401138436)

[Festival Mobile App: 5](#_Toc401138437)

[Product Vision 5](#_Toc401138438)

[Vision for a Festival Mobile App: 5](#_Toc401138439)

[Who will this help? 5](#_Toc401138440)

[Geoffrey Moore: 5](#_Toc401138441)

[Background Information and Research: 6](#_Toc401138442)

[Survey of any existing systems that address the similar problem: 6](#_Toc401138443)

[What is Culture Fox? 6](#_Toc401138444)

[Analysis of Culture Fox 6](#_Toc401138445)

[Registering with Culture Fox: 6](#_Toc401138446)

[After login: 6](#_Toc401138447)

[Account 7](#_Toc401138448)

[Manage Your Account: 7](#_Toc401138449)

[Venues 7](#_Toc401138450)

[View All 7](#_Toc401138451)

[Add Venue 8](#_Toc401138452)

[Findings 10](#_Toc401138453)

[Result of technical research into suitable platforms and Technologies: 10](#_Toc401138454)

[Php: 10](#_Toc401138455)

[ASP: 10](#_Toc401138456)

[MVC: 10](#_Toc401138457)

[Web Pages: 11](#_Toc401138458)

[Creating the mobile app: 11](#_Toc401138459)

[HTML, CSS and JavaScript: 11](#_Toc401138460)

[C#: 11](#_Toc401138461)

[Java: 11](#_Toc401138462)

[Requirements Specifications for the system: 11](#_Toc401138463)

[WCF Service Application: 12](#_Toc401138464)

[Features of WCF 12](#_Toc401138465)

[GitHub 14](#_Toc401138466)

[Xamarin 14](#_Toc401138467)

[How it Works 14](#_Toc401138468)

[Share code between platforms 14](#_Toc401138469)

[Leverage the full .NET runtime 15](#_Toc401138470)

[Native compilation 15](#_Toc401138471)

[SQL Azure 15](#_Toc401138472)

[Scrum 15](#_Toc401138473)

[What is scrum? 15](#_Toc401138474)

[User Stories: 16](#_Toc401138475)

[What are user stories? 16](#_Toc401138476)

[Design 18](#_Toc401138477)

[Twitter Bootstrap 18](#_Toc401138478)

[Wireframing 18](#_Toc401138479)

[Sketch’s for the mobile app 20](#_Toc401138480)

[ERD 25](#_Toc401138481)

[Architecture diagram 26](file:///C:\Users\Patrick\Google%20Drive\Final%20Project\Images%20for%20write%20up\Patrick%20Melia%20-%20Project.docx#_Toc401138482)

[My walkthrough on working with WCF Service Application 27](#_Toc401138483)

[Objective: 27](#_Toc401138484)

[Overview: 27](#_Toc401138485)

[Requirements: 27](#_Toc401138486)

[Creating a WCF Service: 28](#_Toc401138487)

[Configuring Remote Access to IIS Express 30](#_Toc401138488)

[Creating a Xamarin.Android Application 31](#_Toc401138489)

[Record of key implementation decisions 33](#_Toc401138490)

[Logo 33](#_Toc401138491)

[Two Views combined into one: 33](#_Toc401138492)

[User to add a new Festival Type or Event Type into the system: 34](#_Toc401138493)

[Populating festival id in Events table: 35](#_Toc401138494)

[Adding a festival/event logo: 36](#_Toc401138495)

[ViewModels: 36](#_Toc401138496)

[Why use ViewModels? 36](#_Toc401138497)

[Glyphicons: 37](#_Toc401138498)

[Populating the mobile app with data: 37](#_Toc401138499)

[Confirming this is your account: 38](#_Toc401138500)

[Partial Views 38](#_Toc401138501)

[Alerts 38](#_Toc401138502)

[Icons 39](#_Toc401138503)

[Testing 40](#_Toc401138504)

[Method: 40](#_Toc401138505)

[Participants: 40](#_Toc401138506)

[Tasks: 40](#_Toc401138507)

[Testing Procedure: 41](#_Toc401138508)

[Testing Results: 41](#_Toc401138509)

[Analysis and Discussion 42](#_Toc401138510)

[Potential Changes: 42](#_Toc401138511)

[Summary of what was achieved 42](#_Toc401138512)

[Bibliography 43](#_Toc401138513)

# Detailed Description of the problem to be solved:

For my 4th year project my idea was to build a Festival System for users to add details of a Festival along with adding events to that festival. And the next part of the process would be to display the information on a mobile app, for any user to see what is going on around them.

# Project Description:

This is a service where I'll add festivals to website. The user creates an account with us and form there you’ll be able to add as much detail as you possibly can about your festival and events.

Every festival will have events (Main, Children's and Pubs), and on every page it will show where each event is along with the information for each event. Corporate Vision

Festival Web App:This will be like a web portal to add a festival along with events to website and from there, the content you input into the system, will be pulled down into an android mobile app.

Festival Mobile App:This mobile app will call a service to import all the wanted data from the database and display it in the app.

## Product Vision

### Vision for a Festival Mobile App:

* Simple to use: Easy for anyone to use.
* The person using this app will have a list of events along with a GPS location.

### Who will this help?

Holiday makers:People who travel to a country, i.e. Ireland and would like to see what will be happening over the duration of their stay.

Geoffrey Moore:

**For** a festival that’ll extend their marketing in mobile **who** needs a basic mobile app with functionality. The app will help a person to find an event **that** will be on during that day/night. **Unlike** visiting a website to get information/location for an event this app will help them and **our product** will hopefully help them.

# Background Information and Research:

## Survey of any existing systems that address the similar problem:

Before I started rushing into building the website, I went online to see if there was any existing system available to date. I typed ‘free event listing Ireland’ into google and seen that Culture Fox toped the search list.

## What is Culture Fox?

Culture Fox is like your one stop place for search for culture events all around Ireland and abroad. Whatever event you are looking, the user friendly Culture Fox app or site, has a broad range of events to suit your needs. You can search for events or festivals by location, date, nearby by using your location via google maps. Even for the people who only speak “Ás Gaeilge”, can get the site translated for them to read up on a particular festival/event. I have looked at a few areas of this website that I have briefly outlined what these different stages are. They are:

* Registering
* After login
* Venues:
  + Viewing Venues you added.
  + Adding a venue.

# Analysis of Culture Fox

### Registering with Culture Fox:

To register with Culture Fox, all you need is to input for your login details is an email, and password along with a confirm password field. You also have the choice to add “Registrant details”. You have to input your first and last name, company, contact number and address. You are also given the chance to tell a bit about your background and the event that you wish to promote.

### After login:

When you login your account on Culture Fox, you are brought to a welcome screen, Figure 1.

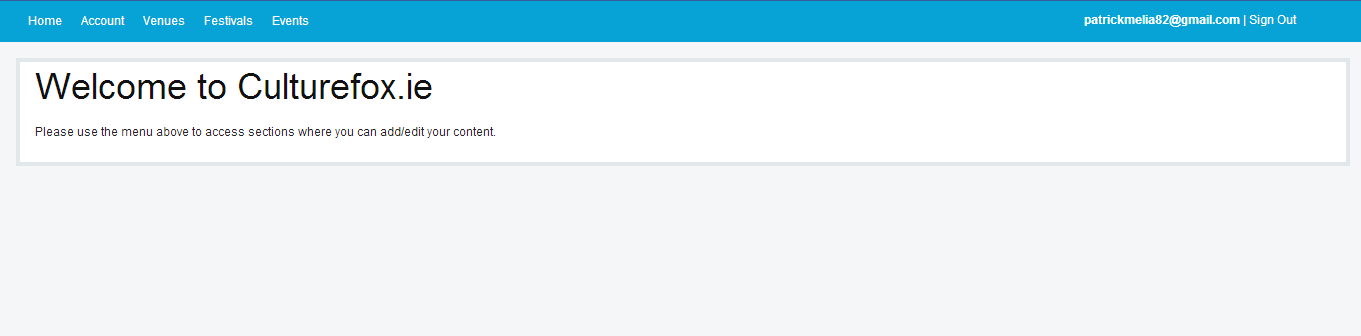
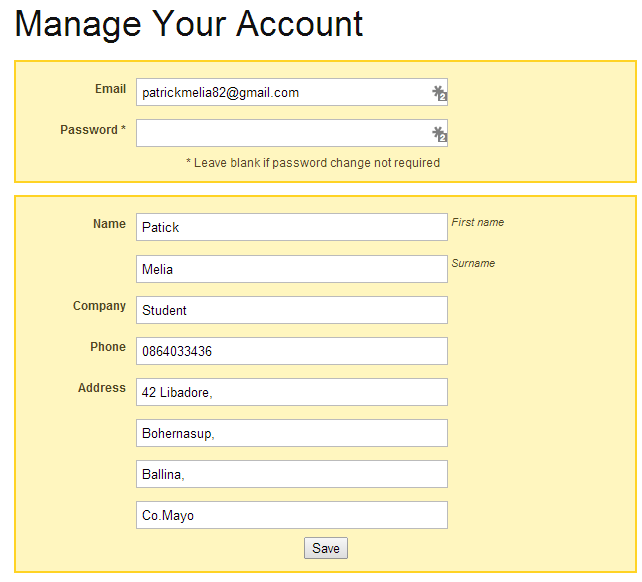


Figure 1

On the navigation bar, you’ll see you can add a festival to the system, add events, and add venues. You also have the ability to manage your account.

## Account



Figure

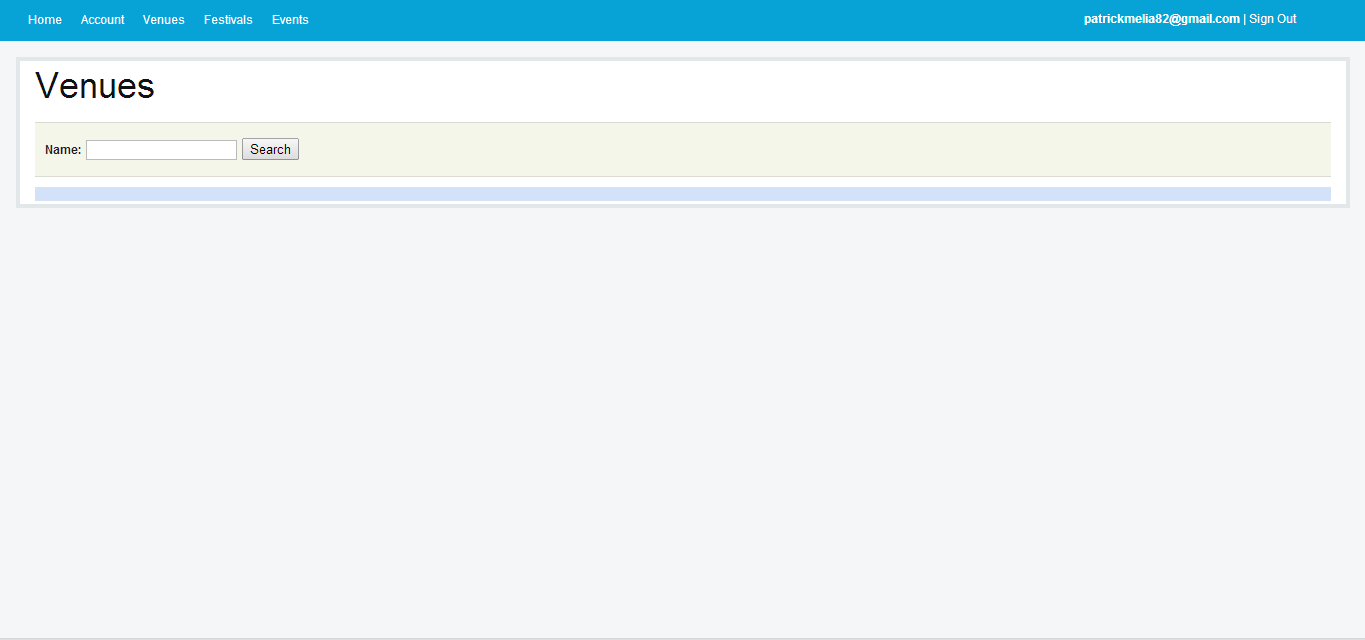
### Manage Your Account:

From Figure 2, you will see a page for managing your account page. Here you have the ability to alter your First name, Surname, Company Name, Phone Number, and address. You ca also change your password. Once you have made these changes you can click the <Save> button.

## Venues

### View All

Seen in figure 3 (seen on the next page), you if you had a list of venues that you added to your account on Culture Fox. So if you had a list of venues there, you are given the opportunity to search for a venue. This would be a handy feature, encase you had to alter any information.

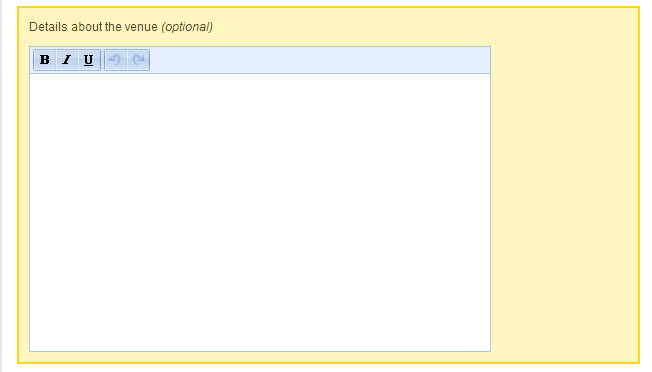
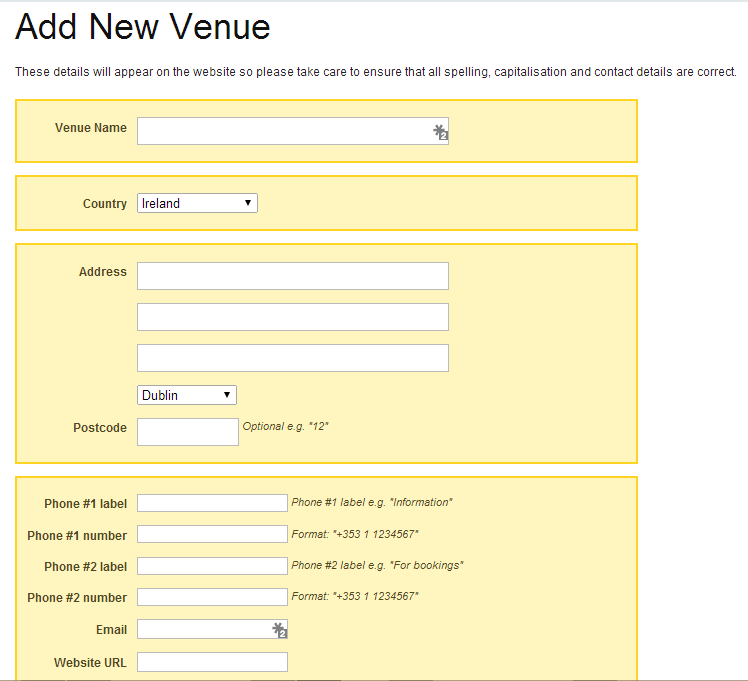


Figure

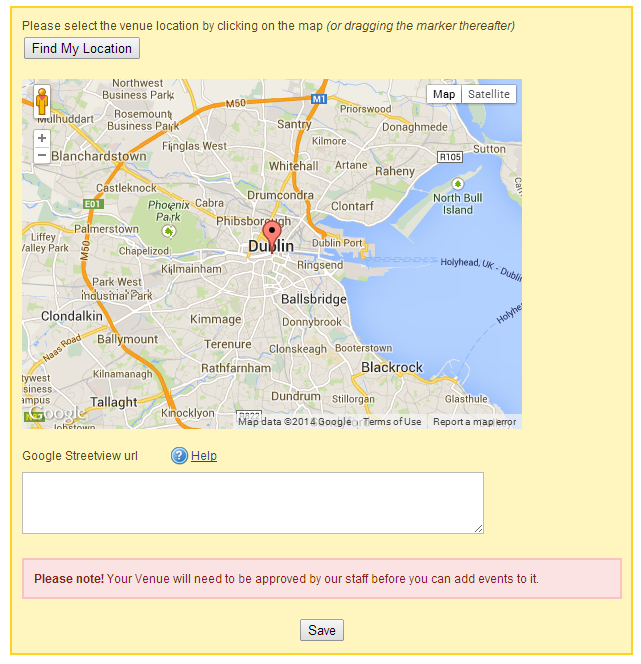
## Add Venue

As you can see in Figure 4 - 6, there is a lot of information to add about a venue on this particular page. What would be better, would be a way to space the inputted information out, over a duration of a few pages.

Figure 4



Figure



Figure

## Findings

After looking at Culture Fox’s website and what they have to over when you are creating an account, adding a festival, event, venue, etc., they would like to get the best possibly information off you to try and broadcast what you need to broadcast. I can take a few off feature, i.e., locating an event by google amps and try to implement this into my own system.

# Result of technical research into suitable platforms and Technologies:

So to build this system I would need to build/design a website along with a mobile app. For starters I am going to do a mobile app for the android platform since I already owe an android smartphone.   
  
I first started to look into the technologies with regards to the website. Here is the different platforms I looked into:

### Php:

PHP (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

|  |  |
| --- | --- |
| ADVANTAGES | DISADVANTAGES |
| * PHP is accessible. * It’s available for free * It’s Available with documentation in many languages. * There are many support groups, forums, and teams to support PHP. * There is a wealth of online information regarding PHP. | * Misperception on lack of Support. * Lack of IDE (Integrated Development Environment) * PHP has no formal error handling mechanisms. |

### ASP.Net:

There are two different platforms under the hood of ASP that I can use, and they are:

### MVC:

MVC is the separation of Model, **V**iew and Controller — nothing more, nothing less. It's simply a paradigm; an ideal that you should have in the back of your mind when designing classes. Avoid mixing code from the three categories into one class.

For example, while a table grid view should obviously present data once shown, it should not have code on where to retrieve the data from, or what its native structure (the model) is like. Likewise, while it may have a function to sum up a column, the actual summing is supposed to happen in the controller.

A 'save file' dialog (view) ultimately passes the path, once picked by the user, on to the controller, which then asks the model for the data, and does the actual saving.

This separation of responsibilities allows flexibility down the road. For example, because the view doesn't care about the underlying model, supporting multiple file formats is easier: just add a model subclass for each.

### Web Pages:

ASP.NET Web Pages and the new Razor syntax provide a fast, approachable, and lightweight way to combine server code with HTML to create dynamic web content. Connect to databases, add video, link to social networking sites, and include many more features that let you create beautiful sites using the latest web standards.

# Creating the mobile app:

So I began to look into the different ways on creating mobile app. I am going to build an android app since I have a Samsung S3 min. You can create a mobile app by using:

## HTML, CSS and JavaScript:

The rise in the use of mobile devices has led many a developer to throw up their hands in frustration. Should a business require both a website and a mobile application? This report discusses why the future may be lie in HTML 5, CSS and JavaScript applications that work equally well on all mobile platforms. PhoneGap is an open source development framework for building cross-platform mobile apps using HTML, CSS and JavaScript. This would be a good approach to use since I already know HTML, CSS and JavaScript.

## C#:

The one reason why I’d like to use C# is because, and I already know C#. So I googled “how to create a mobile application by using C#” and a company called Xamarin, more info can be found on [page 8](#_Xamarin?), has built and ide to help you constructed your apps using the .Net framework/ mono for android.

## Java:

This is another way on creating android apps, only thing is I would have to learn a new language.

# Requirements Specifications for the system:

I am going to use MVC 4 because I’m familiar with the layout because I used it for my 3rd Year Project last year. To build the android app, I am going to use Xamarin/mono for android, due to the fact I am very familiar with C#. I would love to have picked Java, because it is good to learn new languages but due to the time frame of the project I decided to use Xamarin. I am also going to have to create a WCF Service application, this will help the android app to communicate with the database, and populate the app with information.

# WCF Service Application:[[1]](#footnote-1)

Windows Communication Foundation Service (WCF Service) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application. An endpoint can be a client of a service that requests data from a service endpoint. The messages can be as simple as a single character or word sent as XML, or as complex as a stream of binary data. A few sample scenarios include:

* A secure service to process business transactions.
* A service that supplies current data to others, such as a traffic report or other monitoring service.
* A chat service that allows two people to communicate or exchange data in real time.
* A dashboard application that polls one or more services for data and presents it in a logical presentation.
* Exposing a workflow implemented using Windows Workflow Foundation as a WCF service.
* A Silverlight application to poll a service for the latest data feeds.

While creating such applications was possible prior to the existence of WCF, WCF makes the development of endpoints easier than ever. In summary, WCF is designed to offer a manageable approach to creating Web services and Web service clients.

## [Features of WCF](javascript:void(0))

WCF includes the following set of features. For more information, see WCF Feature Details.

* **Service Orientation**

One consequence of using WS standards is that WCF enables you to create *service oriented* applications. Service-oriented architecture (SOA) is the reliance on Web services to send and receive data. The services have the general advantage of being loosely-coupled instead of hard-coded from one application to another. A loosely-coupled relationship implies that any client created on any platform can connect to any service as long as the essential contracts are met.

* **Interoperability**

WCF implements modern industry standards for Web service interoperability. For more information about the supported standards, see Interoperability and Integration.

* **Multiple Message Patterns**

Messages are exchanged in one of several patterns. The most common pattern is the request/reply pattern, where one endpoint requests data from a second endpoint. The second endpoint replies. There are other patterns such as a one-way message in which a single endpoint sends a message without any expectation of a reply. A more complex pattern is the duplex exchange pattern where two endpoints establish a connection and send data back and forth, similar to an instant messaging program. For more information about how to implement different message exchange patterns using WCF see Contracts.

* **Service Metadata**

WCF supports publishing service metadata using formats specified in industry standards such as WSDL, XML Schema and WS-Policy. This metadata can be used to automatically generate and configure clients for accessing WCF services. Metadata can be published over HTTP and HTTPS or using the Web Service Metadata Exchange standard. For more information, see Metadata.

* **Data Contracts**

Because WCF is built using the .NET Framework, it also includes code-friendly methods of supplying the contracts you want to enforce. One of the universal types of contracts is the data contract. In essence, as you code your service using Visual C# or Visual Basic, the easiest way to handle data is by creating classes that represent a data entity with properties that belong to the data entity. WCF includes a comprehensive system for working with data in this easy manner. Once you have created the classes that represent data, your service automatically generates the metadata that allows clients to comply with the data types you have designed. For more information, see Using Data Contracts

* **Security**

Messages can be encrypted to protect privacy and you can require users to authenticate themselves before being allowed to receive messages. Security can be implemented using well-known standards such as SSL or WS-SecureConversation. For more information, see Windows Communication Foundation Security.

* **Multiple Transports and Encodings**

Messages can be sent on any of several built-in transport protocols and encodings. The most common protocol and encoding is to send text encoded SOAP messages using is the HyperText Transfer Protocol (HTTP) for use on the World Wide Web. Alternatively, WCF allows you to send messages over TCP, named pipes, or MSMQ. These messages can be encoded as text or using an optimized binary format. Binary data can be sent efficiently using the MTOM standard. If none of the provided transports or encodings suit your needs you can create your own custom transport or encoding. For more information about transports and encodings supported by WCF see Transports in Windows Communication Foundation.

* **Reliable and Queued Messages**

WCF supports reliable message exchange using reliable sessions implemented over WS-Reliable Messaging and using MSMQ. For more information about reliable and queued messaging support in WCF see Queues and Reliable Sessions.

* **Durable Messages**

A durable message is one that is never lost due to a disruption in the communication. The messages in a durable message pattern are always saved to a database. If a disruption occurs, the database allows you to resume the message exchange when the connection is restored. You can also create a durable message using the Windows Workflow Foundation (WF). For more information, see Workflow Services.

* **Transactions**

WCF also supports transactions using one of three transaction models: WS-AtomicTtransactions, the APIs in the System.Transactions namespace, and Microsoft Distributed Transaction Coordinator. For more information about transaction support in WCF see Transactions [from BPUEDev11].

* **AJAX and REST Support**

REST is an example of an evolving Web 2.0 technology. WCF can be configured to process "plain" XML data that is not wrapped in a SOAP envelope. WCF can also be extended to support specific XML formats, such as ATOM (a popular RSS standard), and even non-XML formats, such as JavaScript Object Notation (JSON).

* **Extensibility**

The WCF architecture has a number of extensibility points. If extra capability is required, there are a number of entry points that allow you to customize the behaviour of a service. For more information about available extensibility points see Extending WCF.

# GitHub

I have been learning how to use GitHub for the last few months. Only thing is I have been using the GUI, which save me a bit of time, learning how to use git/git bash along with commit your project, by using the command prompt. This is handy to use, so I can create a backup of my applications encase my laptop failed. This also is great to use because employers place high value on students that have experience with version control software so for this reason I wanted to get some practical experience using it. I can commit changes and sync it to the server within seconds by using the client GitHub application.

# Xamarin



Xamarin is a library that exposes a single set of APIs for accessing common mobile device functionality across iOS, Android, and Windows platforms. This increases the amount of code developers can share across mobile platforms, making mobile app development easier and faster. Xamarin currently abstracts the contacts, camera, and geo-location APIs across iOS, Android and Windows platforms. Future plans include notifications and accelerometer services. When installing it you have the ability to use their own ide or you can add Xamarin features into Visual Studios.

## How it Works

Write your app in C# and call any native platform APIs directly from C#. The Xamarin compiler bundles the .NET runtime and outputs a native ARM executable, packaged as an iOS or Android app.

## Share code between platforms

Architect your app so that the UI is cleanly separated from the rest of the code. Deliver a device-specific, native user experience by writing code that calls the native APIs, while sharing business logic, data access, and network communications code.

## Leverage the full .NET runtime

Xamarin contains a fully functional implementation of the .NET runtime, called Mono, which is bundled with your app so that your code executes with all of the power of C# and .NET, including memory management, reflection, and the .NET base class libraries.



## Native compilation

Xamarin's compiler is smart - so smart that it can produce exactly the right output for each platform. Xamarin.iOS does full Ahead-of-Time (AOT) compilation to produce an ARM binary suitable for Apple's App Store, while Xamarin.Android takes advantage of Just In Time compilation right on the Android device. Your shared code never needs to know the difference!

Figure 7 -

# SQL Azure



Microsoft Azure SQL Database is a relational database-as-a-service that delivers predictable performance, scalability, business continuity, data protection, and near-zero administration to cloud developers and solution architects. This is the technical library for Azure SQL Database.

# Scrum

For my project, I used scrum. This would help keep track of my weekly outcomes and objectives of this project. Here is what is my finds of scrum, and its meaning.

## What is scrum?

Scrum is an iterative and incremental agile software development framework for managing software projects and product or application development. It defines "a flexible, holistic product development strategy where a development team works as a unit to reach a common goal". It challenges assumptions of the "traditional, sequential approach" to product development. Scrum enables teams to self-organize by encouraging physical co-location or close online collaboration of all team members and daily face to face communication among all team members and disciplines in the project.

# User Stories:

I used user stories through the development of my project. I defined to main roles for this system.

## What are user stories?

User stories is one or more sentences in the everyday or business language of the end user or user of a system that captures what a user does or needs to do as part of his or her job function. User stories are used with agile software development methodologies as the basis for defining the functions a business system must provide, and to facilitate requirements management. It captures the 'who', 'what' and 'why' of a requirement in a simple, concise way, often limited in detail by what can be hand-written on a small paper notecard. Here is a picture of my user stories with regards to a festival attendee (Figure ), and festival organiser (Figure 6).

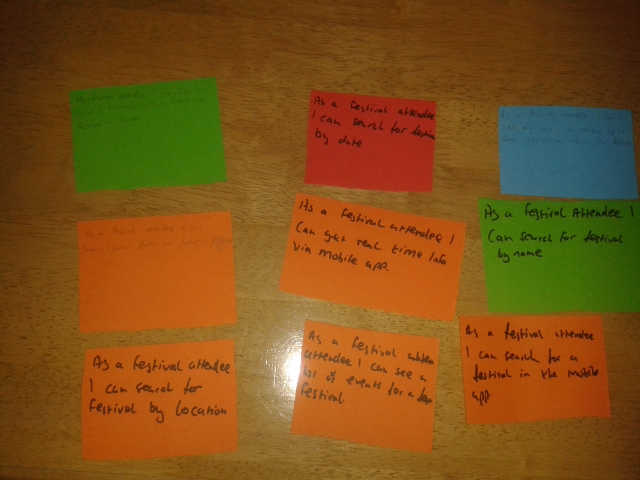
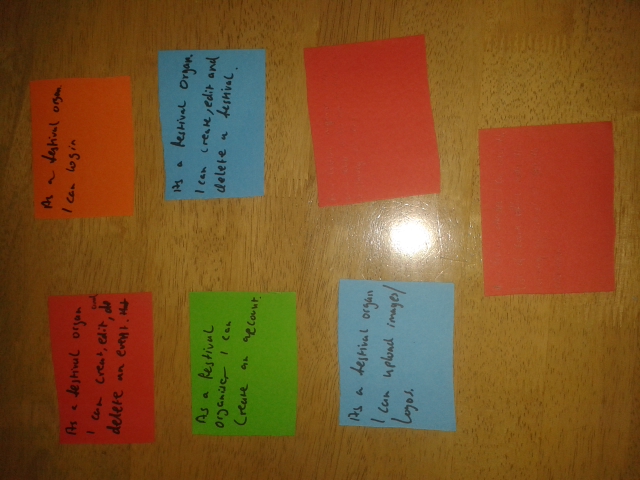


Figure 5



Figure

# Design

For the design process, I started to design both the website and the mobile app.

## Twitter Bootstrap

Since I used Twitter Bootstrap last year alongside with Asp.Net MVC 4 last year, I decided to use this CSS framework again since I liked to look and feel. This would save me to design my own CSS from scratch even though that would be nice to experience but at the time that was a keep part of the system to get done. I had time in the end so I did alter small areas of the CSS.

Wireframing  
So to design the layout of my website I was looking online to see if I can find any UI tool I could use to design my website. I found a program called Balsamiq Mock-ups. Balsamiq is a wireframing and mock up tool with a high focus on usability. Quickly come up with mock ups and easily share them with your clients.

Here is my mock up screens:

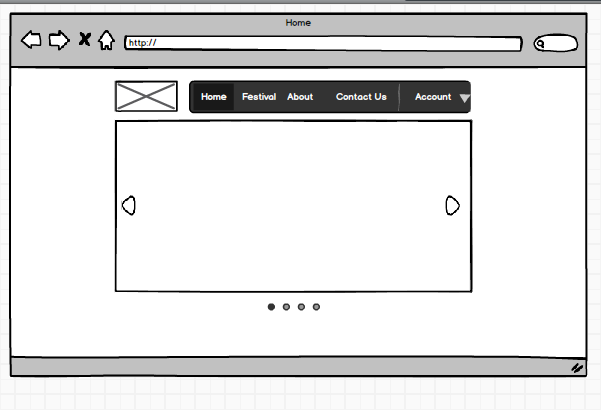


Figure - Homepage

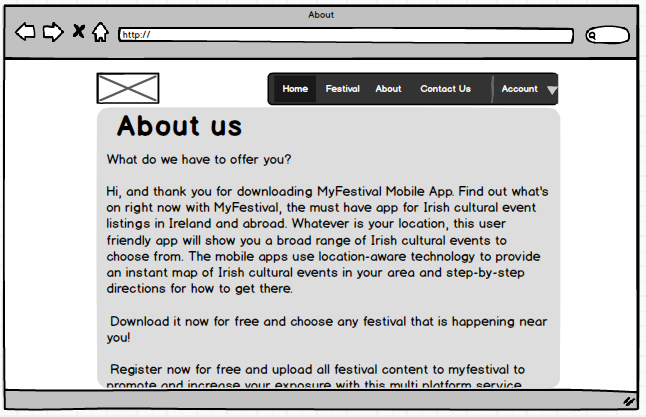


Figure - About Us

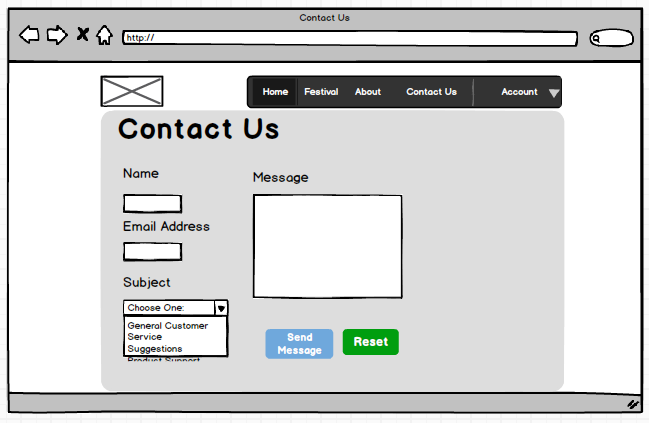


Figure - Contact Us

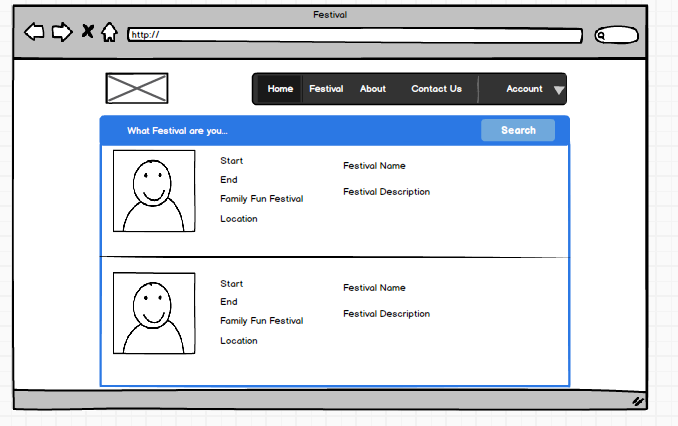
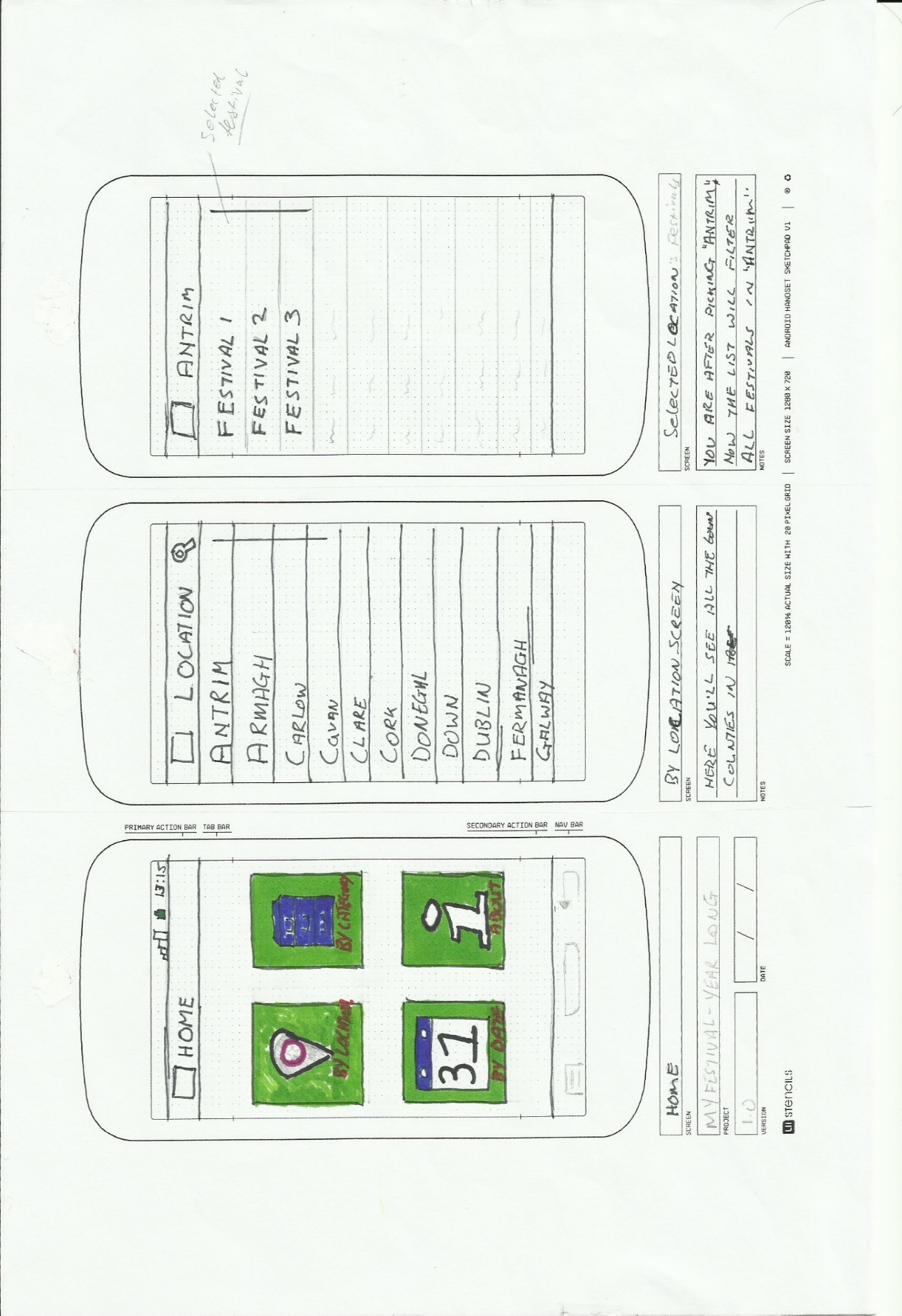


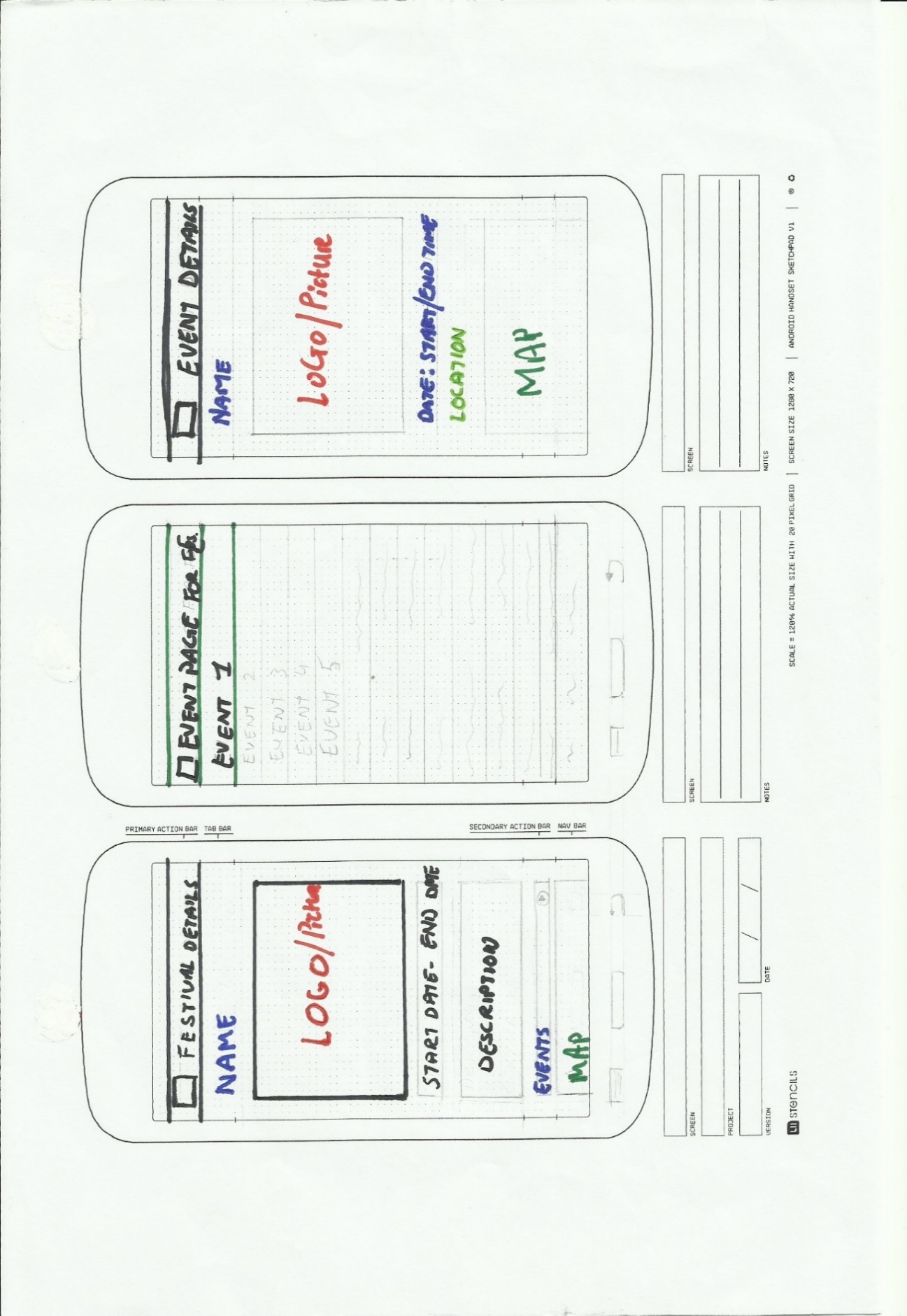
Figure - Find a Festival

## Sketch’s for the mobile app

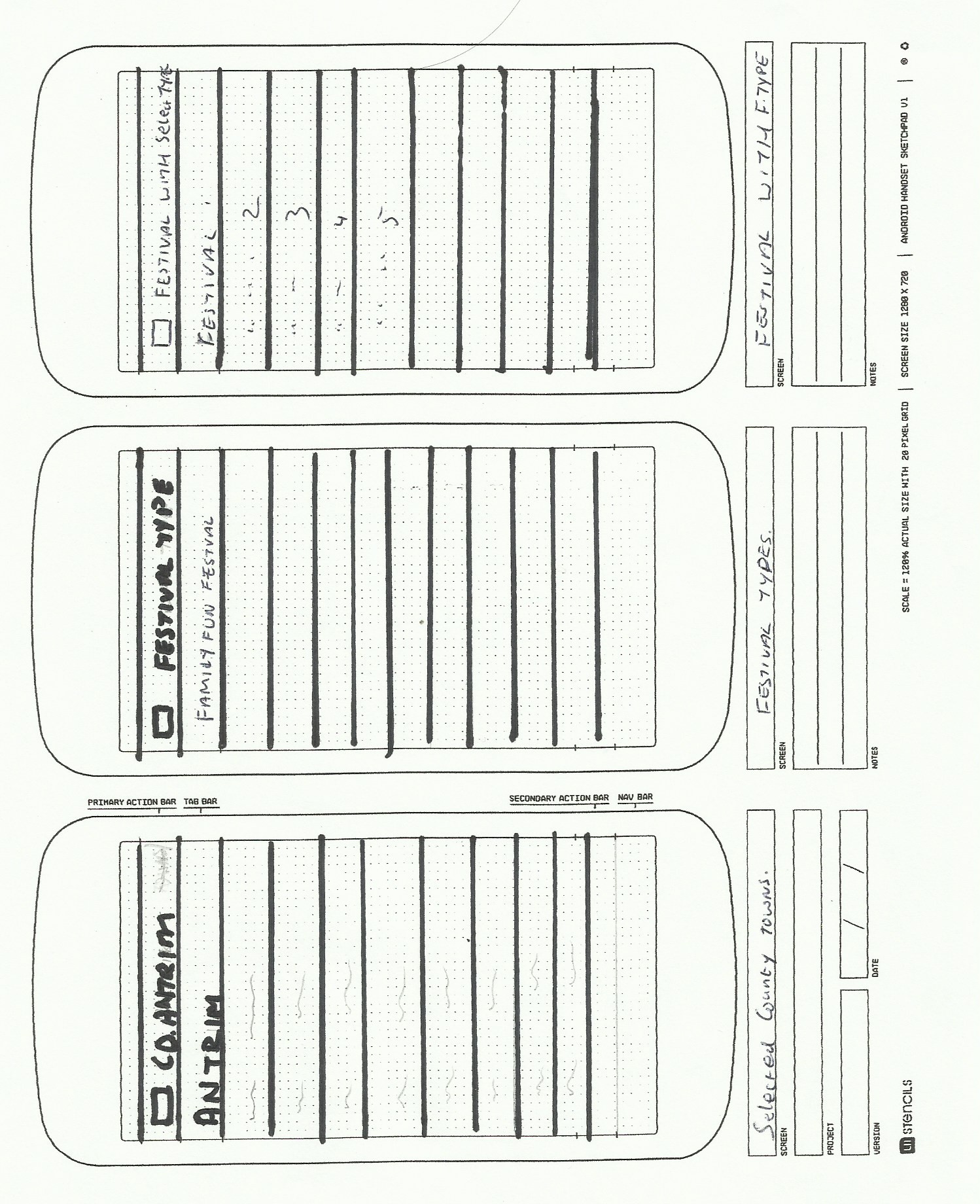
For the mobile app, rather than using Balsamiq, I found UI sketch’s for the Android smartphone UI Stencils [[[2]](#footnote-2)]. (You see them over the next four pages.)



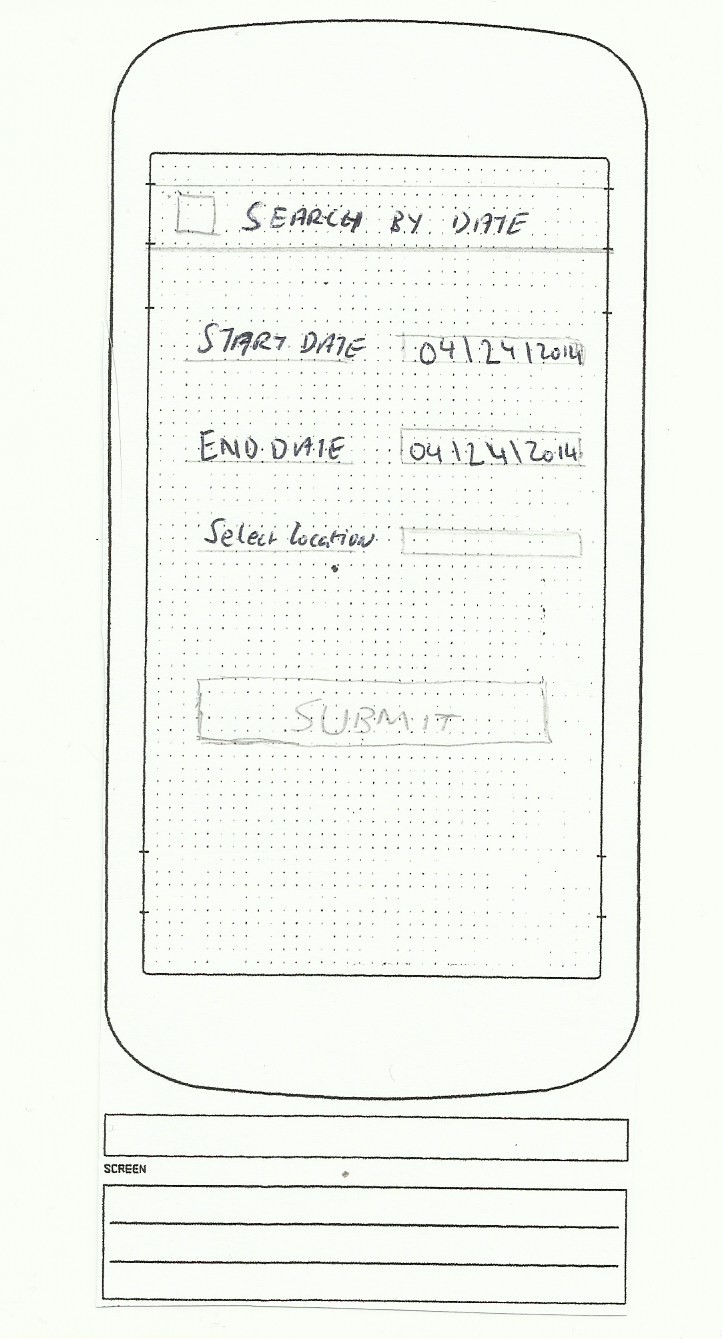
Figure



Figure



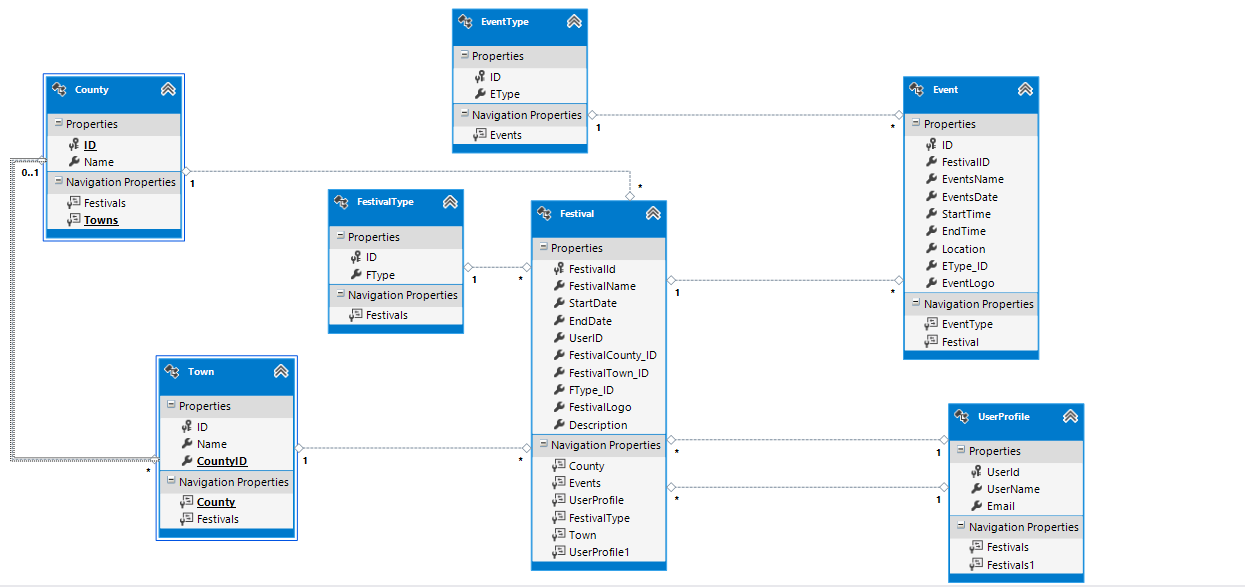
Figure



Figure

# ERD

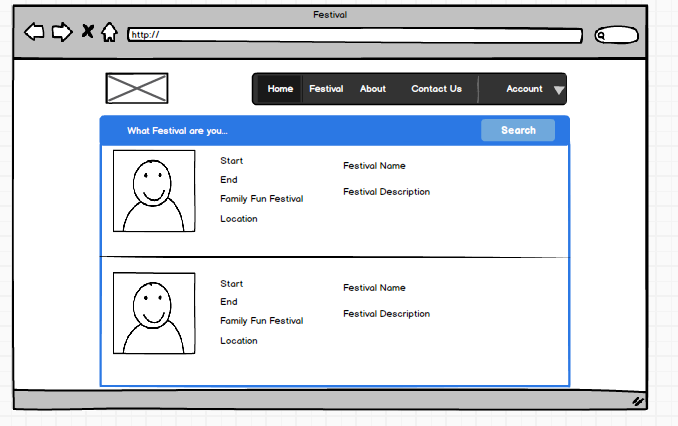
An **entity–relationship model** (**ER model**) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them.



Figure

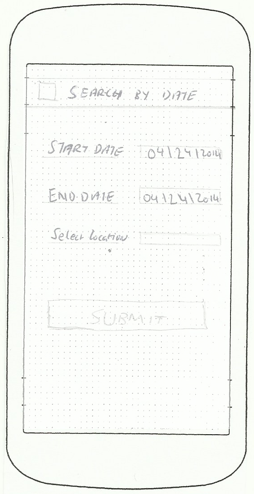
# Architecture diagram

An **architectural model** (in software) is a rich and rigorous diagram, created using available standards, in which the primary concern is to illustrate a specific set of trade-offs inherent in the structure and design of a system or ecosystem. Software architects use architectural models to communicate with others and seek peer feedback. An architectural model is an expression of a viewpoint in software architecture.



SQL AZURE

|



In

Out

View

Controller

Model

WCF SERVICE

Application

# My walkthrough on working with WCF Service Application

## Objective:

To call a WCF Service from a mobile app to populate ListViews with data from the database.

## Overview:

It is a common requirement for mobile applications to be able to communicate with backend systems. There are many choices and options for backend frameworks, one of which is [Windows Communication Foundation](#_WCF_Service_Application) (WCF). This walkthrough will provide an example on how my Xamarin mobile application can consume a WCF service using the BasicHttpBinding, as outlined below:

**Create a WCF Service** - In this section we will create a very basic WCF service having two methods. The first method will take a string parameter, while the other method will take a C# object. This section will also discuss how to configure a developer's workstation to allow remote access to the WCF service.

**Create a Xamarin.Android Application** - Once the WCF service has been created, we will create a simple Xamarin.Android application that will use the WCF service. This section will cover how to create a WCF service proxy class to facilitate communication with the WCF service.

## Requirements:

This walkthrough assumes that you have some familiarity with creating and using WCF services.

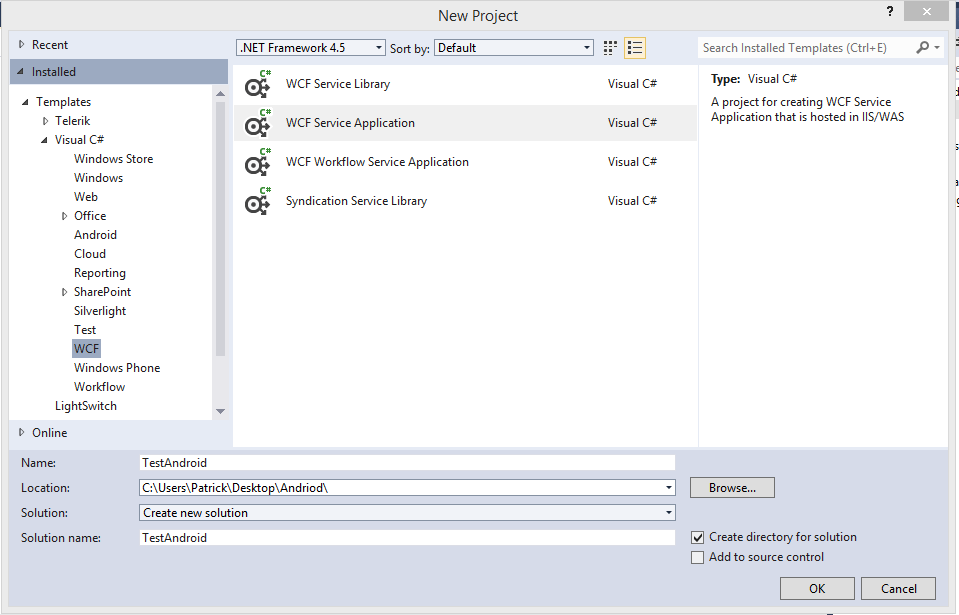
In order to create the WCF service proxies, you will need the [Silverlight 5 SDK](http://www.microsoft.com/en-us/download/details.aspx?id=28359) installed. Download and run the installer from Microsoft before proceeding with this walkthrough.

IIS Express will be used to host the WCF Service used in this walkthrough. This is the default web server for Visual Studio 2012/2013. Installing IIS Express in older versions of Visual Studio is beyond the scope of this walkthrough.

# Creating a WCF Service:

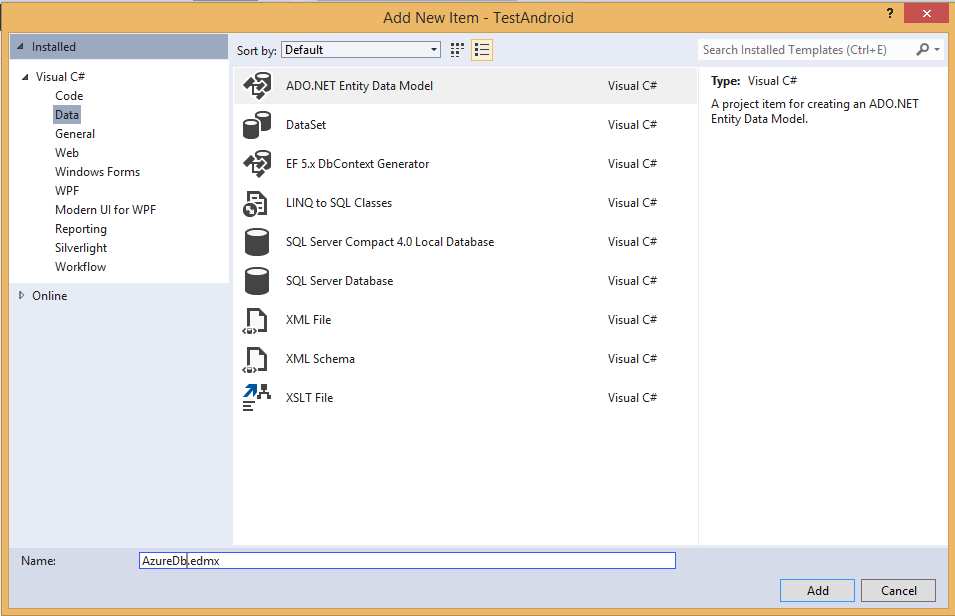
The first task before us is to create a WCF service for our mobile applications to communicate with.

1. Start-up Visual Studio 2012, and press Alt-F-N-P to open the New Project dialog. Select the WCF Service Application template from the new project dialog as shown in the following screenshot and name the project TestAndroid, and click the OK button (Figure 19).



Figure

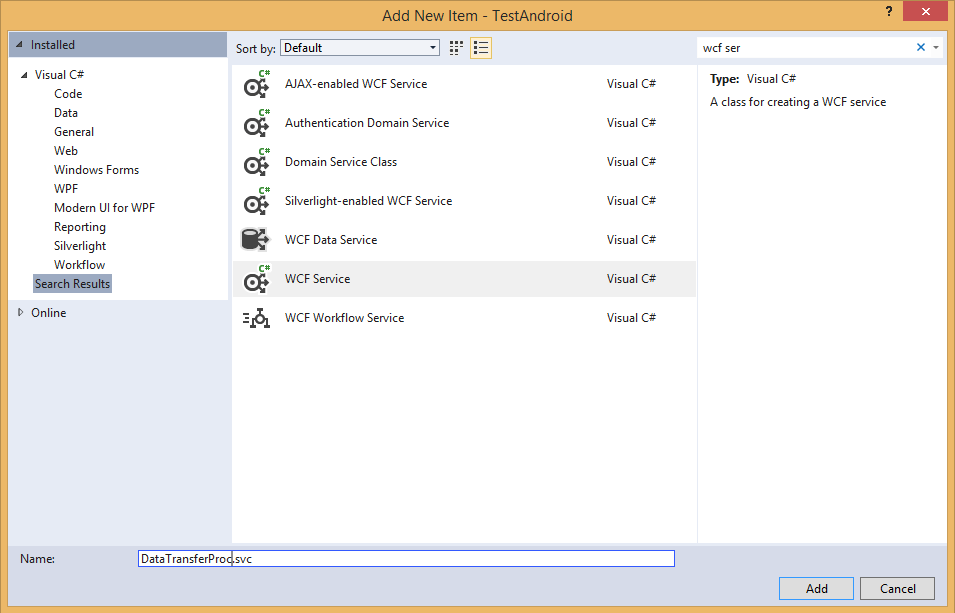
1. Next we need to create the service contract for the web service. Add an interface, name IDataTransferProc.cs and paste in the following code, which can be found on my GitHub repo by clicking here.
2. Next I add an ADO.NET Entity Data Model and called it AzureDb.edmx, this will create the connection to my database to get relevant information from the database to the mobile app.



Figure

After adding this I followed the relevant steps to add the connection to the database.

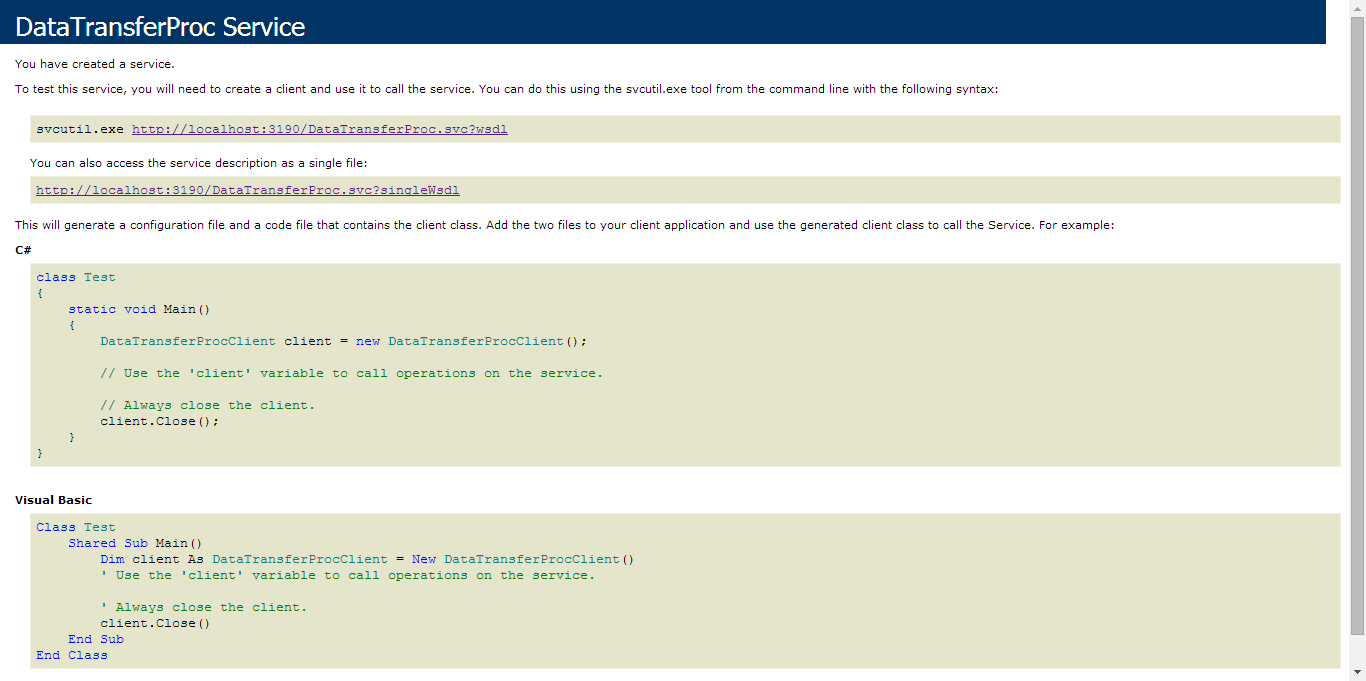
1. The final thing we need to do is to create the WCF Service class. Press Ctrl-Shift-A to bring up the Add New Item dialog:



Figure

Add a new WCF Service class to the project named DataTransferProc. Edit the class so that it implements IDataTransferProc and contains the code from the following this [link](https://github.com/PMelia07/MyFestivalWCF/blob/master/TestAndroid/DataTransferProc.svc.cs).

The Final step was for me to run the service (F5), if I got a page that looked like this screen shot, its works:

****

Figure

## Configuring Remote Access to IIS Express

By default, Windows 8 and IIS Express will not accept remote connections. Before any remote devices, such as an Android device or an iPhone can communicate with our WCF service we must make the following changes:

1. **Configure IIS Express to Accept Remote connections** - This step involves editing the config file for IIS Express to accept remote connections on a specific port and then setting up a rule for IIS Express to accept the incoming traffic.
2. **Add an Exception to Windows Firewall** - We must open up a port through Windows Firewall that remote applications can use to communicate with the WCF service.

You will need to know the IP address of your workstation. For the purposes of this example we'll assume that our workstation has the IP address 192.168.1.143.

Let's begin by configuring IIS Express to listen for external requests. We do this by editing the configuration file for IIS Express at %userprofile%\documents\iisexpress\config\applicationhost.config.

Locate the site element with the name TestAndroid. It should look something like the following XML snippet:

<site name="TestAndroid" id="20">

<application path="/" applicationPool="Clr4IntegratedAppPool">

<virtualDirectory path="/" path="/" physicalPath="C:\Users\Patrick\Desktop\TestAndroid\TestAndroid"/>

</application>

<bindings>

<binding protocol="http" bindingInformation="\*:3190:localhost" />

<binding protocol="http" bindingInformation="\*:3190:192.168.1.1" />

</bindings>

</site>

1. Next, we need to configure IIS Express accept incoming connections on port 9608. Start-up up an administrative command prompt, and run this command:

> netsh http add urlacl url=http:// 192.168.1.1:3190/ user=everyone

1. The final step is to configure Windows Firewall to permit external traffic on port 9608. From an administrative command prompt, run the following command:

> netsh advfirewall firewall add rule name="IISExpressXamarin" dir=in protocol=tcp localport=3190 profile=private remoteip=localsubnet action=allow

This command will allow incoming traffic on port 3190 from all devices on the same subnet as the Windows 8 workstation.

At this point we have created a very basic WCF service hosted in IIS Express that will accept incoming connections from other devices or computers on our subnet. You can test this out by visiting http://localhost:9607/HelloWorldService.svc on your workstation and http://192.168.1.1:3190/DataTransferProc.svc from another computer on your subnet.

# Creating a Xamarin.Android Application

Now that we have the WCF service working, let's move on to creating a Xamarin.Android application.

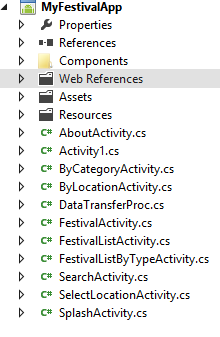
1. Add a new Android project to the solution and name it MyFestival.
2. Next we need to create a proxy for the web service. To create the proxy we'll use the *Silverlight Service Model Proxy Generation Tool* (SLsvcUtil.exe). You can find this command line utility at the following location:

C:\Program Files (x86)\Microsoft SDKs\Silverlight\v5.0\Tools\SLsvcUtil.exe

Ensure that the WCF service we created in the previous section is running, and then run SLsvcUtil.exe with the following command line:

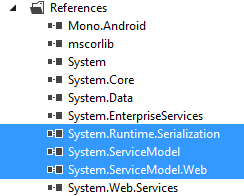
SLsvcUtil.exe /noConfig <http://localhost:31690/DataTransferProc.svc>

This will create a service proxy called DataTransferProc in the file DataTransferProc.cs. Add this file to your MyFestival project as shown in the following screenshot:



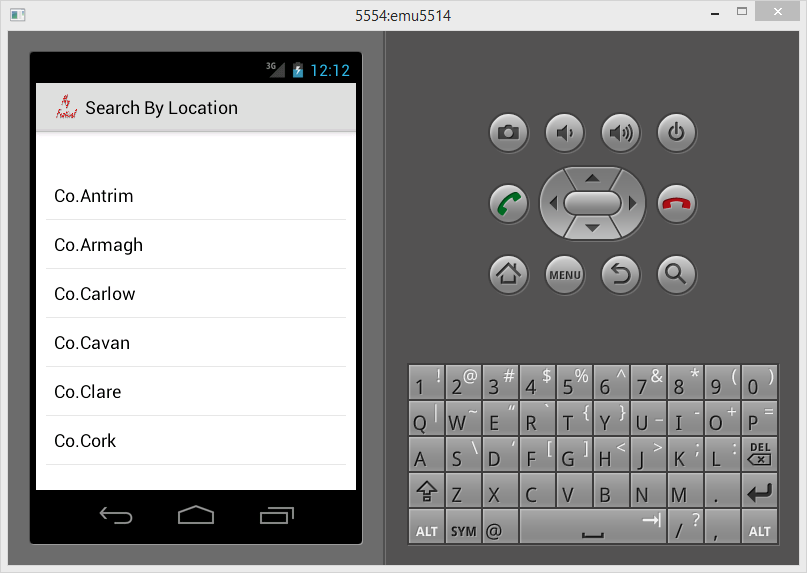
Figure

1. Before this generated proxy class will compile, we need to add to some references to our Xamarin.Android project, as shown in the following screenshot:



Figure

1. Next, update Main.axml with the following XML, which can be found [here](https://github.com/PMelia07/MyFestivalAndroidWCF/blob/master/ByCategoryActivity.cs).
2. With the above infrastructure in place, we can finish up the Android application. Edit the file Activity1.csand add the following instance variables that can be found [here](https://github.com/PMelia07/MyFestivalAndroidWCF/blob/master/ByLocationActivity.cs). You’ll find all the code to create the back end code to complete this task. From initializes the instance variables for our class and wires up some event handlers.
   * The code above instantiates and initializes a DataTransferProcClient object. The WCF proxy class can only call the WCF service asynchronously. Responses from the WCF service will handled by the xxxCompletedevents that were generated by SLsvcUtil.exe.
   * Finally, we need to add event handlers for the xxxCompleted events of the DataTransferProcClient proxy client.
3. Than finally all you have to do is load the app and away you go, you should see a view like this:



Figure

# Record of key implementation decisions

## Logo

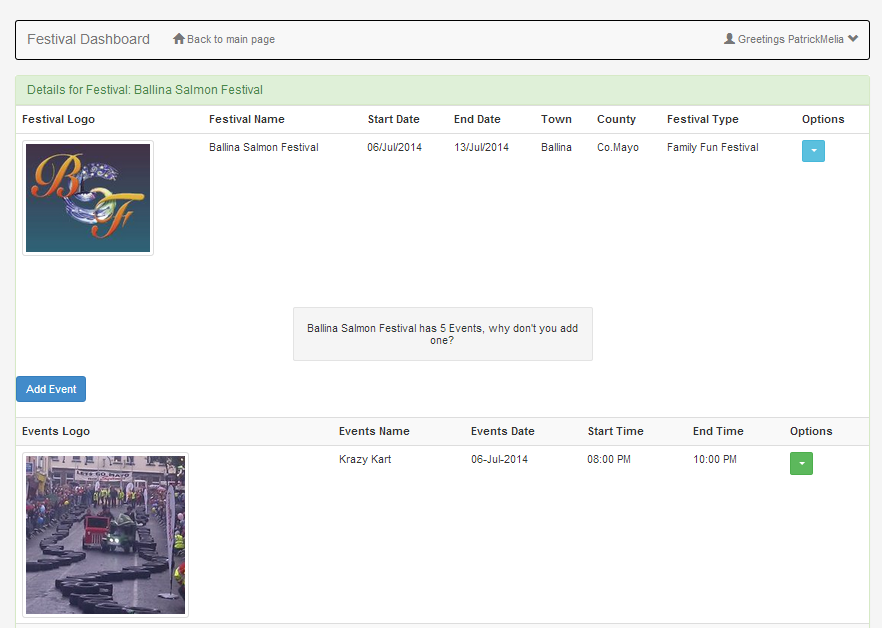
This was only done within the last month. But still, at least this suites the website, and make the website stand out a bit more. Here is the logo:



Figure

## Two Views combined into one:

For the Details view a selected festival, I combined the festival view along with the events of that festival. I have also place a counter on this page also to tell the user that festival x has x events. Here is a picture of what this view looks like, along with a piece of code below.



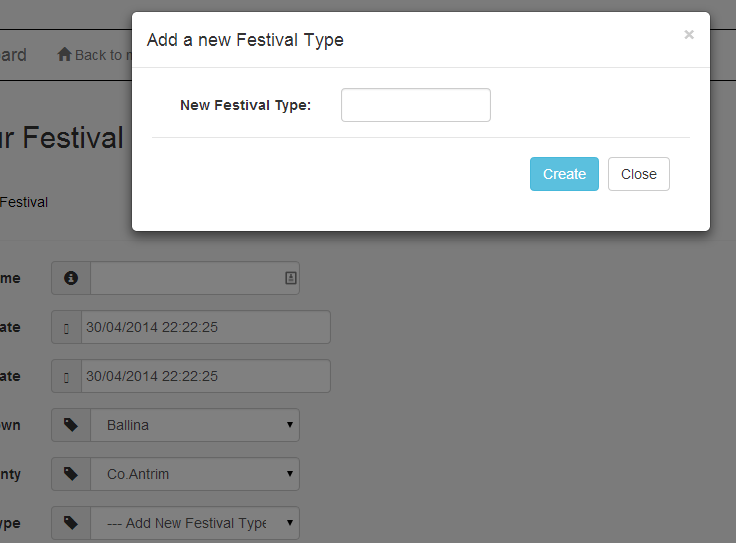
Figure

|  |
| --- |
| public ActionResult Details(int id)//, DataTable tbl)  {  int CurrentID = WebSecurity.CurrentUserId;  Festival festival = db.Festivals.Find(id);  if (festival == null)  {  return HttpNotFound();  }  else  {  festival.Events = (from e in db.Events  where e.FestivalID.Equals(id) && festival.UserID == CurrentID  select e).ToList();  }  return View(festival);  } |

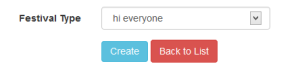
## User to add a new Festival Type or Event Type into the system:

I have added this feature so a user can create or add a new festival or event type. At first I had implanting was when “---Add a new Festival Type---” was selected, a modal (partial view) will become visible (Figure 28). This will all the user to add a new festival/event type. After you add a new type, the dropdown list will select the inputted type (Figure 29). Here is the core piece of code that will help you to add a new Festival/Event Type:

|  |
| --- |
| public JsonResult CreateFestivalType(FestivalTypeVM model)  {  if (ModelState.IsValid)  {  if (db.FestivalTypes.Where(ft => ft.FType.ToLower() == model.Name.ToLower()).ToList().Count == 0)  {  db.FestivalTypes.Add(new FestivalType { FType = model.Name });  db.SaveChanges();  model.FestivalTypeID = db.FestivalTypes.Where(ft => ft.FType.ToLower() == model.Name.ToLower()).Single().ID;  return Json(model);  }  else  {  ModelState.AddModelError(string.Empty, "Festival type already exsist");  return Json(model);  }  }  return Json(model);  } |



Figure



Figure

## Populating festival id in Events table:

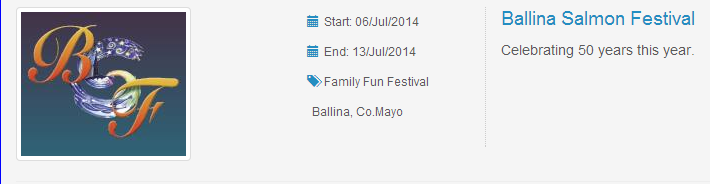
I have the web application to pass a festival from the festival to the event. This will be a lot easier rather than letting the user add an event to a particular festival.

|  |
| --- |
| HTTPGET :  public ActionResult Create2(int festID)  EventsVM events = new EventsVM { festivalID = festID };  Create View  @Html.HiddenFor(m => m.festivalID);  HTTPPOST  public ActionResult Create2(EventsVM model, int festID, HttpPostedFileBase imageFile)  Newevent.FestivalID = model.festivalID = festID; |

## Adding a festival/event logo:

When adding a festival or event to your account you have the chance to add a festival logo or image/event image or logo. This is not a requirement for the system. I also have created a folder for each of these uploads. Here is a core piece of code that helps you add them to the application.

|  |
| --- |
| string fileName = model.FestivalName;  string serverPath = Server.MapPath("~\\Content\\FestivalLogo");  Bitmap newImage = new Bitmap(Request.Files["imageFile"].InputStream);  newImage.Save(serverPath + "\\" + fileName + ".jpg", System.Drawing.Imaging.ImageFormat.Jpeg);  Newfestival.FestivalLogo = "../../Content/FestivalLogo/" + fileName + ".jpg"; |



Figure

## ViewModels:

## Why use ViewModels?

#### Reason 1: Remove logic from your Views

When you start working with ASP.NET MVC, you’ll most likely ask yourself why you should use a View Model. Using your domain model or entity model works perfectly fine. And it does. For a while. But as you continue to use your Models, you’ll discover that you have to add some adaptations in your Views. Here is a typical usage adaptation (using the Razor View engine):

Hello @model.UserName

Your age: @(model.Age != 0 ? model.Age.ToString() : "n/a")

Not so bad, is it? The problem is that you have introduced logic into your View, which is bad for two reasons:

1. You can’t unit test that code, and the only way to make sure that it works is user testing.
2. You need to repeat that code for every View that intends to use your Model (code duplication = code smell).

All those small adaptations will lead to a big mess eventually.

#### Reason two: Security

One of the biggest advantages with View Models is removing security risks. Your database objects or domain objects will most likely contain properties that the user should not be able to change. It can be a property called IsAdmin orReputation.

All those properties will automatically be changed by the model binder if they exist in the model (and are posted in the FORM, which is quite easy to do if you know HTML). Simply remove them from the View Model and they’ll never be changed.

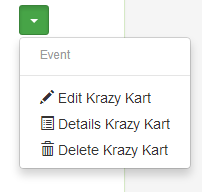
#### Reason three: Loose coupling

By using domain models or entity models, you are adding coupling between your lower layers and the presentation layer, and that is seldom good. Google “loose coupling” to find out more.

Basically, it means that if you change your domain/entity model, you have to change all your Views that use that Model. If you use a ViewModel, you only have to change the code that maps between an entity/domain model and the View Model.

## Glyphicons:

I have used a few glyphicons here and there throughout the whole application to try and make the application a bit more user friendly and let them know what their next step is when the click on the a particular link.



Figure

## Populating the mobile app with data:

At the time, I was planning on building the app along with a SQLite database, with the same information I have in the websites database. I didn’t think this was could at the time because you’d have to find a way on updating that database every time changes was made. So I decided I’d google different ways on populating an android app, and luckily there was a tutorial on Xamarin’s website [[[3]](#footnote-3)]. From there I followed the tutorial to build a WCF Service app that would communicate to the database. More step by step tutorial that I wrote up can be found [here](#_My_walkthrough_on).

## Confirming this is your account:

I am using Postal to send a confirmation email to the user when he or she creates an account with my application. From there you’ll click on the link within the email and this will send you straight back into the system and check to see if you have the right confirmation code. If you do, happy days, you’ll be brought to the login page. If not you’ll be ask to check your email again until you get it right.

|  |
| --- |
| Try  {  string confirmationToken =  WebSecurity.CreateUserAndAccount(model.UserName, model.Password,  new { model.Email }, true);  dynamic email = new Email("RegEmail");  email.To = model.Email;  email.UserName = model.UserName;  email.ConfirmationToken = confirmationToken;  email.HostLocation = Request.Url.Host + ':' + Request.Url.Port;  email.Send();  return RedirectToAction("RegisterStepTwo", "Account");  //return RedirectToAction("Index", "Festival");  }  catch (MembershipCreateUserException e)  {  ModelState.AddModelError("", ErrorCodeToString(e.StatusCode));  }  } |

## 

## Partial Views

A partial view enables you to define a view that will be rendered inside a parent view. Partial views are implemented as ASP.NET user controls (.ascx).  When a partial view is instantiated, it gets its own copy of the ViewDataDictionary object that is available to the parent view. The partial view therefore has access to the data of the parent view. However, if the partial view updates the data, those updates affect only the partial view's ViewData object. The parent view's data is not changed.

## Alerts

When creating adding updating and deleting information about your Festival/Event from the system, you’ll see that you will appear. Here is a picture of the alerts. (Figure 32)

2014-05-01_1458

2014-05-01_1457

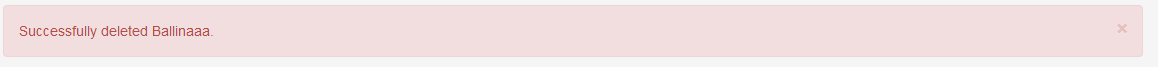


Figure 2

## Icons

For the mobile app I created my own icons by using Photoshop. I also created a splash screen, which you can see in the bottom right hand corner. (Figure 33)



Figure 3

# Testing

## 

## Method:

I now have a fully operating website for anyone to use, combined with user friendly and mobile first design, along with each element into one system that worked and flowed naturally much like an online process would. As a single member project, when I added a new feature, I always tested it out to make sure it works, rather than leaving it to the last minute to test all the features in my system.

One night at a local festival committee meeting, I brought my laptop and asked the a few members of the committee would they have a spare few minutes after the meet to look over my web application and give me feedback on what could be added or taken out of the system and provide the simplest and most authentic experience possible.

## Participants:

So after that meeting I got 3 people to test my system. These people have been active members of the festival for more than 5 years. Each member has a different background with the system.

* Participant 1: Public relations Officer (PRO) is about managing reputation. The PRO use all forms of media and communication to build, maintain and manage the reputation of the festival
* Participant 2: Event Leader, this person task is to run an event during the week of the festival.
* Participant 3: The Chairperson of a festival ensures that all positions are filled with local people. He oversees the coordinators and sub-committees and whom is responsible on the day of the Festival to ensure that all things run smoothly

## Tasks:

The tasks which were selected to be tested are outlined below.

* Navigate from the home page to the register page.
* Create an account with the system.
* Confirm his/her account by using the link in the confirmation email.
* Login to his /her account.
* Create a festival
* Add an event to that festival
* Edit a festival
* Edit an event to that festival
* Delete a festival or an event.

## Testing Procedure:

The testing procedure was done in the steps to get the best possible observations and feedback from the test.

* Equipment was previously set out.
* Each applicant had the overall process explained to them briefly before signing the consent forms. [[[4]](#footnote-4)]
* Each applicant was tasked with stepping through and completing each task one by one with the opportunity to give feedback at the end of each task so as to best maximise results, at the same time they were observed and notes were taken.
* Before each section to be tested, they were given a brief overview of what each section was supposed to accomplish as well as the features which would not be working so as to avoid the error any expectancy may have; it was a visual representation of what we wished to achieve but was not included in the scope of the assignment and testing being completed.
* Each applicant gave overall feedback at the end where data was compiled and stored for later use.

## Testing Results:

Here a small bit of feedback on what I got from the three participants:

* Participant 1: Event leader: Nicely designed and easy to follow. Only thing is a proper mapping system where you can drop a map marker to say where a relevant event took place.
* Participant 2: PRO: Again referring the nicely designed layout and the flow of adding a festival along with its events. Only thing is missing would be the contact details of the festival for example, email, Facebook, twitter and even a contact number of the festival office.
* Participant 3: Chairperson: Firstly this person has never used a computer in his life, so best thing he pointed out would be a tutorial video or a step by step guide on how to set an account, add your festival details along with the event details. Otherwise he liked the look and feel of the website.

During the duration of this test I then should them a paper mock-up of the mobile app and explain it is a working progress and see what they’d think of it. And if I had the time I’d show it to them and get their feedback. The videos can be found here:

* Participant 1: Has to look for a Festival and Event from a selected County and Town. Here is the link for this video: <https://www.youtube.com/watch?v=R8htLAlgGn0>
* Participant 2: Has to look for a Festival and Event by selecting a particular festival type. Here is the link for this video: <http://youtu.be/05Kw6rRw9pk>.
* Participant 3: Has to look for a Festival and Event by searching for the information by start date, end date, and selected location. Here is the link for this video: <http://youtu.be/AS-u6GePfcM>.

Or go start to my playlist and see all three videos: <https://www.youtube.com/playlist?list=PLvhqarEi_2eIlqLCWPCjTYjirDEh9cVtW>.

# Analysis and Discussion

Over the duration of this project I have achieve to create the Web app to administrate and serve the MyFestival App. I have meet the spec that my supervisor and I first set out back in October with regards to the web site.

## Potential Changes:

* If I was back in September right now, redoing this project I would first do the mobile app. The reason for this is so I could get a database built with inputted data. Another reason to add on to this, is to also have a mapping system within the app.
* I would have add a feature to the app to call the WCF Service application, and create a SQLite database, when the WCF Service has been called, so the data will be available offline. If the data within the database was alter it would dump the database and rebuild the database again from scratch and insert new data into the database.
* After getting the web application test by patently users, I got a small bit of feedback on what would be the next step on adding feature to the system. One good feature would have the ability to have a drag and drop marker on google maps so a user could use this as a reference to find a particular event. It was also suggested to have an introduction video to the site, this would also help first time users.

# Summary of what was achieved

Now that was a quick year! I have truly enjoyed doing this project, there was a few mountains I had to climb to achieve a few things but got most of the objectives that I have set out to do. So what have I achieved? Here is a list of what you can do

* Got an MVC 4 application created to my full potential:
  + A registered user
    - Create an account.
    - Confirm their account.
    - Add, edit and delete a festival.
    - Add, edit and delete an event belonging to a particular festival.
    - Add a new festival/event type to the website.
    - Add an image to both/either a festival or event.
  + A non-logged in user can
    - Look up all festivals/events on the website
    - Can search for festival by name.
* Mobile App
  + On the Xamarin. Android app you can:
    - View all the counties in the app.
    - View all towns from that selected county within the app.
    - View all festivals from that selected town within the app.
    - View all Festival Types in the app.

# Bibliography

Culture Fox [Online]. Culturefox.ie. Available from: <http://www.culturefox.ie/>  
[Viewed 10th November 2013 – 1st April 2014]

(Galloway Jon), (Haack Phil), (Wilson, Brad), (Scott Allen, K.), (Hanselman, Scott), October 2012. *Professional ASP.NET MVC 4*

(McClure, Wallace B.), (Blevins, Nathan), (Croft IV, John J), (Dick, Jonathan), (Hardly,Chris), March 2012. *Hardy Professional Android Programming with Mono for Android and .NET/C#*

PATRA Diptimaya, *Xamarin Android Application Development* [Online] - <https://leanpub.com/xamarin-android-app-dev> [viewed 20th January 2013]

1. Microsoft. (Unknown). *What Is Windows Communication Foundation.*Available: http://msdn.microsoft.com/en-us/library/ms731082(v=vs.110).aspx. Last accessed 20th of April 2014. [↑](#footnote-ref-1)
2. http://cdn.shopify.com/s/files/1/0042/9602/files/Android-Handset-Sketchpad-v1-a4-BW.pdf?338 [↑](#footnote-ref-2)
3. Xamarin. (). *Walkthrough - Working with WCF.* Available: http://docs.xamarin.com/guides/cross-platform/application\_fundamentals/web\_services/walkthrough\_working\_with\_WCF/. Last accessed 12th April 2014. [↑](#footnote-ref-3)
4. Melia, P. (2014). *Consent for Participation in Interview Research.docx.*Available: https://goo.gl/bE26lM. Last accessed 2014. [↑](#footnote-ref-4)