Exploration and Requirements

Table of Contents

[Iteration 1 (Exploration & Requirements) 2](#_Toc349848834)

[Project Overview 2](#_Toc349848835)

[Target Audient 2](#_Toc349848836)

[Technical Approach 3](#_Toc349848837)

[ASP.NET 3](#_Toc349848838)

[Java 3](#_Toc349848839)

[Azure 3](#_Toc349848840)

[HTML5/CSS 3](#_Toc349848841)

[JavaScript/JQueryMobile 3](#_Toc349848842)

[XML 4](#_Toc349848843)

[CSV 4](#_Toc349848844)

[Entity Framework 4](#_Toc349848845)

[SQL 4](#_Toc349848846)

[User Stories 4](#_Toc349848847)

[Log In 4](#_Toc349848848)

[Generate Registration Form 5](#_Toc349848849)

[Select Input type 5](#_Toc349848850)

[Table Exports 6](#_Toc349848851)

[Admin Access 6](#_Toc349848852)

[Account Creation 6](#_Toc349848853)

[Public URLS 7](#_Toc349848854)

[Mobile Forms 7](#_Toc349848855)

[Release Plan 8](#_Toc349848856)

[Iteration 2 10](#_Toc349848857)

[Application Architecture Definition 10](#_Toc349848858)

[MVC 10](#_Toc349848859)

[Revised User Stories 10](#_Toc349848860)

[Log In 10](#_Toc349848861)

[Generate Form 10](#_Toc349848862)

[Select Input type 11](#_Toc349848863)

[Table Exports 11](#_Toc349848864)

[Admin Access 12](#_Toc349848865)

[Account Creation 12](#_Toc349848866)

[Public URLS 13](#_Toc349848867)

[Mobile Forms 13](#_Toc349848868)

[Statistics 13](#_Toc349848869)

[Payment 14](#_Toc349848870)

[Implemented User Stories 14](#_Toc349848871)

[Story ID: 1 – Log In 14](#_Toc349848872)

[Story ID: 2 – Generate Forms 14](#_Toc349848873)

[Story ID: 3 14](#_Toc349848874)

[Story ID: 6 – Account Creation 14](#_Toc349848875)

[Class Diagram 15](#_Toc349848876)

[Database Schema 16](#_Toc349848877)

[Entity Framework 16](#_Toc349848878)

[SMO Database 17](#_Toc349848879)

[Revised Release Plan 17](#_Toc349848880)

[18](#_Toc349848881)

[Difficulties and Learning Outcomes 19](#_Toc349848882)

[Story ID: 1 – Log In 19](#_Toc349848883)

[Story ID: 2 – Generate Forms 19](#_Toc349848884)

[Story ID: 6 – Account Creation 19](#_Toc349848885)

[Bibliography 20](#_Toc349848886)

# Iteration 1 (Exploration & Requirements)

## Project Overview

The project attempts to simplify the sign up and registration process by allowing for an electronic solution to the traditional pen and paper approach still widely used today. Large-scale form processing applications may be impractical for smaller businesses or personal uses. Creating the data in a digital format from the offset eliminates unnecessary data input and simplifies the process for the user. Capturing this data digitally allows demographics and other statistics to be generated quickly and conveniently.

Application integration is an important aspect of this system, as users will invariably wish to process this data with any number of applications for use in mailing lists, etc. .CSV files are supported by a wide variety of applications and as such we feel this is the best format to focus on for export purposes.

The Application encompasses a Web Service built on the ASP.NET platform and Database backend implemented using Entity Framework and SQL Database on Microsoft’s Azure platform and a Client side application and Web App.

## Target Audient

The application can serve a wide range of purposes but is primarily aimed at small to medium businesses, Charity organisations and personal use.

The application would be suitable for use at events such as trade fairs employment fairs, conventions, seminars, and charity functions. Tablets running the application could be installed in stores for customers, allowing businesses to engage customer footfall with the e-commerce side of the business as well. Door-to-door salespersons, street teams or personal users organizing trips, meetings or college society sign ups may also find the application useful.

## Technical Approach

### ASP.NET

The ASP.NET MVC platform has been chosen to provide the HTML view of the full browser web page, user authentication and to provide backend integration with the server and database. This language is one of our required Microsoft technologies as part of our entry into Microsoft’s Imagine Cup competition.

Some of the benefits of ASP.NET over other possible solutions such as PHP include a higher level of abstraction, allowing easier development in a short time period. (Microsoft, n.d.) Operating in a more familiar OOP development environment will lead to increased productivity and a smaller learning curve. ASP.NET can also be fully integrated with Microsoft’s Azure service, whereas with other programming languages the Azure SDK does not include a full set of Azure APIs at present leading to more complex code when it comes to integrating. (Parker, 2011)

### Java

The app is to be developed using the Java programming language with the Android SDK as a native android app for use with both tablets and smartphones. (Android, n.d.) Android development is supported cross platform between Mac OS X, Windows and Linux Operating systems through the Eclipse IDE which accommodates both members of the programming teams home operating systems. (Android, n.d.)

### Azure

Windows Azure is a cloud computing solution developed by Microsoft. This platform provides us with many benefits including the following:

Websites can be deployed using ASP.NET using Git or FTP, SQL Database, an API built on REST, HTML and XML, Azure integrates well with Git and Eclipse. (Microsoft, n.d.)

### HTML5/CSS

HTML 5 technologies will be used in both the full browser implementation of the site and in combination with other technologies to develop the front end of the mobile application. HTML5 was developed to provide flexibility and allow more to be accomplished with stand alone HTML. Some of the major benefits associated with HTML is the addition of the canvas and video tag as well as support for geolocation, client side local storage/database, offline cache and support for basic threading. (W3Schools, n.d.)

### JavaScript/JQueryMobile

jQuery Mobile integrates HTML5, CSS3, jQuery and jQuery UI into a touch-optimized, cross platform mobile framework. It is designed to enhance development and to support a wide range of mobile platforms. Ajax support will prove invaluable to the project. For these reasons we have decided to use jQuery Mobile in combination with an Android SDK for developing a hybrid version of the application. (JQuery, n.d.)

### XML

XML and JSON will be the main data formats used for communication between mobile device’s and the server. Due to limitations of mobile devices, a vast amount of apps have to transmit large amount of data over the Internet in order to deliver full functionality to the user. XML and JSON are the most common data exchange formats today and they both offer flexible and capable approaches to this problem. (IBM, n.d.)

### CSV

The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. Surprisingly, while this format is very common, it has never been formally documented. so the format is operationally defined by the many applications which read and write it. (Python.org, n.d.)

### Entity Framework

Entity Framework (EF) is an object-relational mapper that enables .NET developers to work with relational data using domain-specific objects. It provides a layer of abstraction that allows us to bypass the use of SQL statements to access and modify relational databases.

### SQL

To dynamically create user tables we will use a combination of Entity Framework and SQL. Template tables will be created using Entity Framework to allow users to specify customer columns which will then be committed and stored as an SQL table in the database.

## User Stories

### Log In

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 1** | **Story Title: Log In** | **Priority: High** | **Estimate: 2 hours** |
| As a user I want to log in so that I can access my account | | | |

**Confirmation:**

1. **Success** – valid user logged in and referred to personal home page
   1. User Presented with a list of their associated sheets table entries.

Which can be selected to view the individual signatories.

1. **Failure** – display message
   1. Username and hash of password does not match that stored in the database.
   2. Display failed login message, allow retry up to 3 attempts.

### Generate Registration Form

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 2** | **Story Title: Generate Registration Form** | **Priority: High** | **Estimate: 5 hours** |
| As a User I want to generate custom registration forms so that I can capture data from clients | | | |

**Confirmation:**

1. **Success** – Valid table names provided, table created, stored and associated with user profile
   1. User returned to list of tables including newly created one.
2. **Failure** – Invalid Input
   1. User enters values disallowed by form validation
      1. Indicated by highlighting invalid field
   2. Failed table creation
      1. Notify user and allow retry

### Select Input type

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 3** | **Story Title: Select Input Type** | **Priority: High** | **Estimate: 5 hours** |
| As a User I want to select predetermined input types (radio buttons, checkboxes) for certain custom fields, and be able to specify custom answers. | | | |

**Confirmation:**

1. **Success** – Table created with defined input types and default question layout.
   1. User returned to list of tables including newly created one.
2. **Failure** – Invalid Input
   1. User enters values disallowed by form validation
      1. Indicated by highlighting invalid field
   2. Failed table creation
      1. Notify user and allow retry

### Table Exports

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 4** | **Story Title: Table Exports** | **Priority: High** | **Estimate: 5 Hours** |
| As a User I want to export Tables so that I can easily import them to other applications. | | | |

**Confirmation:**

1. **Success** – File downloaded to client machine.
   1. File downloaded to client machine at user defined location
2. **Failure** – Cannot export table.
   1. Table may be being updated by another device (AJAX call)
   2. Table not found

### Admin Access

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 5** | **Story Title: Admin access** | **Priority: High** | **Estimate:4 Hours** |
| As a sit administrator I want access users information to build profiles and send emails. | | | |

**Confirmation:**

1. **Success** – View list of usernames, full names, location and email addresses.
   1. View displayed to admin
2. **Failure** –
   1. Detail failures & messages

### Account Creation

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 6** | **Story Title: Account Creation** | **Priority: High** | **Estimate: 5 Hours** |
| As a User I want to be able to create new account so that I can log in. | | | |

**Confirmation:**

1. **Success** – User account created
   1. user redirected to log in page
2. **Failure** – Cannot create account, display message
   1. Account already exists
   2. Invalid inputs/password does not match

### Public URLS

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 7** | **Story Title: Public URLs** | **Priority: Medium** | **Estimate:** |
| As a User I want to share public links to social networking sites so that clients can complete form remotely. | | | |

**Confirmation:**

1. **Success** – Public URL provided to user
   1. Option to share directly to social networking sites
2. **Failure** – Unable to provide public URL

### Mobile Forms

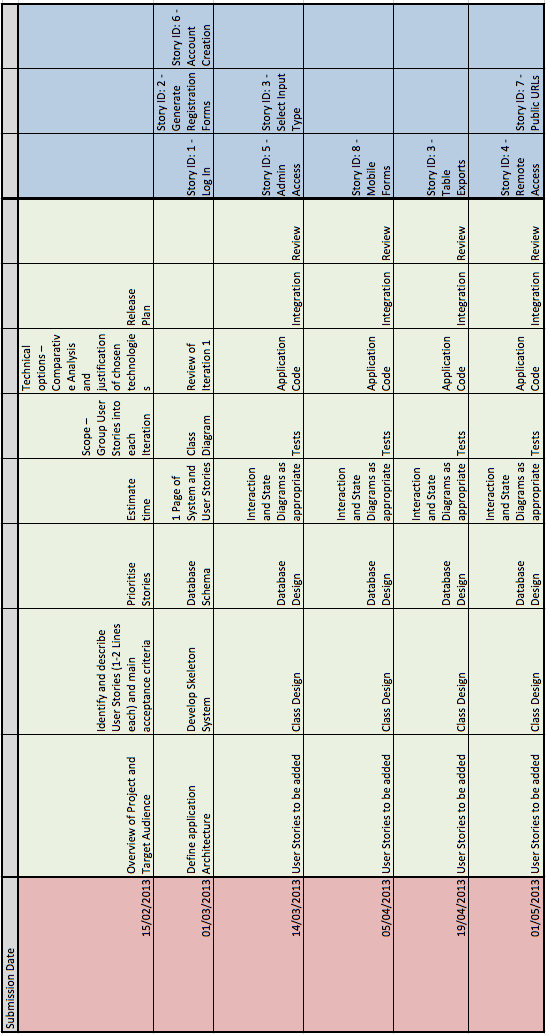
|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 8** | **Story Title: Mobile forms** | **Priority: High** | **Estimate:** |
| As a User I want to populate tables from the mobile app so that I can present the form to customers in person | | | |

**Confirmation:**

1. **Success** – User verified and tables associated with user displayed
   1. User selects table to populate
   2. Form presented to screen
2. **Failure** – No Tables to display
   1. User provided with option of creating new table.

## Release Plan

Also Attached

.

# Iteration 2

## Application Architecture Definition

### MVC

The Model View Controller architecture is being implemented in this project.

MVC is an Object Orientated development methodology to enable code reuse by separating the application into 3 parts: the Model, View and Controller.

The model is charged with managing fundamental behaviours and the data for the application, typically seen implemented as a database. The view is the layer at which the user may interact with the application. It is the user interface and is responsible for rendering data passed to it from the model. The controller handles user input via the view layer and makes calls to model objects.

## Revised User Stories

### Log In

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 1** | **Story Title: Log In** | **Priority: High** | **Estimate: 2 hours** |
| As a user I want to log in so that I can access my account | | | |

**Confirmation:**

1. **Success** – valid user logged in and referred to personal home page
   1. User Presented with a list of their associated sheets table entries.

Which can be selected to view the individual signatories.

1. **Failure** – display message
   1. Username and hash of password does not match that stored in the database.
   2. Display failed login message, allow retry up to 3 attempts.

### Generate Form

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 2** | **Story Title: Generate Form** | **Priority: High** | **Estimate: 5 hours** |
| As a User I want to generate custom registration forms so that I can capture data from clients | | | |

**Confirmation:**

1. **Success** – Valid table names provided, table created, stored and associated with user profile
   1. User returned to list of tables including newly created one.
2. **Failure** – Invalid Input
   1. User enters values disallowed by form validation
      1. Indicated by highlighting invalid field
   2. Failed table creation
      1. Notify user and allow retry

### Select Input type

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 3** | **Story Title: Select Input Type** | **Priority: High** | **Estimate: 5 hours** |
| As a User I want to select predetermined input types (radio buttons, checkboxes) for certain custom fields, and be able to specify custom answers. | | | |

**Confirmation:**

1. **Success** – Table created with defined input types and default question layout.
   1. User returned to list of tables including newly created one.
2. **Failure** – Invalid Input
   1. User enters values disallowed by form validation
      1. Indicated by highlighting invalid field
   2. Failed table creation
      1. Notify user and allow retry

### Table Exports

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 4** | **Story Title: Table Exports** | **Priority: High** | **Estimate: 5 Hours** |
| As a User I want to export Tables so that I can easily import them to other applications. | | | |

**Confirmation:**

1. **Success** – File downloaded to client machine.
   1. File downloaded to client machine at user defined location
2. **Failure** – Cannot export table.
   1. Table may be being updated by another device (AJAX call)
   2. Table not found

### Admin Access

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 5** | **Story Title: Admin access** | **Priority: High** | **Estimate:4 Hours** |
| As a sit administrator I want access users information to build profiles and send emails. | | | |

**Confirmation:**

1. **Success** – View list of usernames, full names, location and email addresses.
   1. View displayed to admin
2. **Failure** –
   1. Detail failures & messages

### Account Creation

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 6** | **Story Title: Account Creation** | **Priority: High** | **Estimate: 5 Hours** |
| As a User I want to be able to create new account so that I can log in. | | | |

**Confirmation:**

1. **Success** – User account created
   1. user redirected personal homepage
2. **Failure** – Cannot create account, display message
   1. Account already exists
   2. Invalid inputs/password does not match

### Public URLS

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 7** | **Story Title: Public URLs** | **Priority: Medium** | **Estimate:** |
| As a User I want to share public links to social networking sites so that clients can complete form remotely. | | | |

**Confirmation:**

1. **Success** – Public URL provided to user
   1. Option to share directly to social networking sites
2. **Failure** – Unable to provide public URL

### Mobile Forms

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 8** | **Story Title: Mobile forms** | **Priority: High** | **Estimate:** |
| As a User I want to populate tables from the mobile app so that I can present the form to customers in person | | | |

**Confirmation:**

1. **Success** – User verified and tables associated with user displayed
   1. User selects table to populate
   2. Form presented to screen
2. **Failure** – No Tables to display
   1. User provided with option of creating new table.

### Statistics

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 9** | **Story Title: Statistics** | **Priority: Medium** | **Estimate: 2 Days** |
| As a User I want to analyse table data so that I can better target and understand my clients and determine target markets | | | |

**Confirmation:**

1. **Success** – Table analytics displayed
   1. User can filter results into categories
2. **Failure** – No Analytics available
   1. Table empty
   2. No data fields in table are capable of having analytics associated with them

### Payment

|  |  |  |  |
| --- | --- | --- | --- |
| **ID: 10** | **Story Title: Payment** | **Priority: Low** | **Estimate:**  **2 Weeks** |
| As a User I want Clients to pay subscription fees in-line with the registration form. | | | |

**Confirmation:**

1. **Success** – User logs in with 3rd party payment details (e.g. PayPal, PayClick, WePay, etc.)
   1. Details verified, payment received.
2. **Failure** – Details Not verified
   1. User notified of payment failure and returned to form

## Implemented User Stories

### Story ID: 1 – Log In

User can log in to the application. Once acceptable log in criteria are entered, the user is redirected to their personal home screen.

This view allows the user to see any of their existing forms or create new forms. Forms may be edited, deleted or submitted to the SQL. Once submitted, data can be entered into the table through the user view. Any data which has been entered into the form can be viewed by clicking the form name from the user home page.

### Story ID: 2 – Generate Forms

The user can create new forms by providing the Form name and any required form fields. When submitting the form to the SQL database, a unique identifier for the form is created by concatenating the users’ ID and the form ID. Once submitted, the action is confirmed and the form is ready to be used.

### Story ID: 3

When generating forms and their fields, the user must specify of what type (Text, Number, Date, etc.) input the field should expect. This information is used when creating the SQL database tables for the forms and for formatting the user view displays.

### Story ID: 6 – Account Creation

Users must supply a username, email address and password (which must be entered twice) to register an account. Once a valid input is received, the account is created and the user is logged in and presented with their personal home page.

## Class Diagram



## Database Schema

### Entity Framework



### SMO Database



## Revised Release Plan

## C:\Users\shanemurphy\Dropbox\3rdYearImagineCup\Submitions\Upload 2 - 1st March\ReleasePlan.PNG

## Difficulties and Learning Outcomes

As we began coding our solution, we quickly discovered our experience of .NET and C# to be very basic. This proved to be the main obstacle we needed to overcome in this iteration, as we built on existing knowledge to understand the .NET framework and C# at a much deeper level.

We increased time spent on the system by a number of hours to compensate for time spent searching for coding solutions and logical solutions to our problems. Because of this, we managed to stay on course and complete all tasks and user stories required for Iteration 1.

At the end of this iteration, it was decided to implement error logging to better document, keep track of and understand our problems.

### Story ID: 1 – Log In

#### Profile Linking

Linking the users’ profile, managed using SimpleMembership with their associated tables, which were generated and stored on a separate SQL Server database. To do this, we needed to change the standard SimpleMembership profile, connect this profile to our own Model Classes and get the User ID using WebSecurity.

### Story ID: 2 – Generate Forms

#### Unique Table IDs

To store individual user tables, we needed to ensure that all table names were unique. To do this, we appended the User ID and Form ID to the table name. E.g. UserTable\_1\_2 for User ID = 1, Form ID = 2.

#### Cascade Delete

Our application requires the use of Entity Framework and a Separate SQL Server. To keep our databases in sync, we needed to ensure deleting the Entity Framework table would also drop the SQL equivalent table.

#### Views using Column Names

Column names, which were taking as input from the user, needed to be passed to a view to display this data to the user again. Because these columns were actually stored in an Entity Framework table as data in the table, we needed a way to pass this data to the view, rather than the actual column names (which display as Field 1, Field 2, Field 3, etc.)

#### Reading data back from SQL Database

We found it challenging to read any data back from our SQL Server tables into our system. Past experience with Java and SQL through JDBC proved to be extremely useful to us in this task, however returning information to the application via a result set, as in JDBC, was not an option.

### Story ID: 6 – Account Creation

#### Small issues

We encountered many issues working with 2 databases simultaneously. Using SMO with the SQL Database and managing User Profiles using Entity Framework brought on many small issues regarding combining data, authenticating data and returning data to user views.

# Bibliography

Android. (n.d.). *Android*. Retrieved December 07, 2012, from http://www.android.com/about/

Android. (n.d.). *Exploring the SDK*. Retrieved December 07, 2012, from http://developer.android.com/sdk/exploring.html

IBM. (n.d.). *Articles, online tutorials, and other technical resources on XML standards and technologies*. Retrieved December 03, 2012, from DeveloperWorks: http://www.ibm.com/developerworks/xml

JQuery. (n.d.). *JQueryMobile*. Retrieved November 30, 2012, from http://jquerymobile.com/

Microsoft. (n.d.). *ASP.NET Overview*. Retrieved December 03, 2012, from http://msdn.microsoft.com/en-us/library/4w3ex9c2(v=vs.100).aspx

Microsoft. (n.d.). *Getting Started with Azure*. Retrieved December 07, 2012, from http://www.windowsazure.com/en-us/home/features/overview/

Parker, R. (2011, June 1). *How to migrate your ASP.NET site to the Azure cloud*. Retrieved December 03, 2012, from Developer Fusion: http://www.developerfusion.com/article/119960/upgrade-your-aspnet-site-to-the-cloud/

Python.org. (n.d.). *CSV File Reading and Writing*. Retrieved January 31, 2013, from http://docs.python.org/2/library/csv.html

W3Schools. (n.d.). *HTML 5 Intro*. Retrieved December 05, 2012, from http://www.w3schools.com/html/html5\_intro.asp