

**UNIVERSITY OF MEDICAL SCIENCES & TECHNOLOGY**

**FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY**

**Final Year Project**

**MOBI-COLLECT v.1.0: Textual Data Collection  
using Mobile Phones**

Submitted in partial fulfillment for the award of

The Degree of Bachelor of Science in

Computer Science and Information Technology

Submitted by  
**Shripal Parekh**  
**CS2007-014**

Supervisor: Dr.Mohammed Awad

**UMST - 2009/2010**

---

## **Abstract**

The usage of mobile phones is abundant in our daily lives in various aspects from making phone calls or sending text messages to checking e-mails or news updates to planning our activities or managing our budget.

This project aims at making use of this wide spread usage of mobiles to help in the data collection process. It designs and develops a web based system called "MobiCollect" that is used for creating forms or questionnaires to be later accessed by the data collectors using their mobile phone web browser in order fill in the form with the appropriate data.

Once the system design and implementation is completed it will be tested and evaluated to ensure the satisfaction of at least the minimum requirements of the proposed system.

## **Acknowledgements**

I would firstly like to thank my project supervisor, Dr. Mohammed Awad, for answering all of my questions, keeping track on my progress and supporting me throughout the project and having his trust in my ability to successfully deliver it.

I would like to further extend my thanks to the whole teaching unit of the faculty of Computer Science including my all time mentor Mr. Mohammed Izzeldin, for all the guidance and advises provided during the project progress.

I am deeply grateful to my family and friends for their much appreciated motivation and moral support.

Lastly, I want to acknowledge Dr. Joel Selainko – Co-founder of Episurveyor which was the primary guideline I followed for this project and Dr. Joel promptly answered my queries and gave valuable suggestions.

# Contents

1	INTRODUCTION.....	6
1.1	OVERVIEW.....	6
1.2	PROBLEM STATEMENT.....	6
1.3	PROJECT AIM.....	7
1.3.1	OBJECTIVES.....	7
1.3.2	SCOPE OF THE PROJECT.....	8
1.4	POSSIBLE ENHANCEMENTS.....	8
1.5	DELIVERABLES.....	8
1.6	METHODOLOGY.....	8
1.7	TOOLS AND TECHNIQUES.....	9
1.8	INITIAL PROJECT SCHEDULE.....	9
2	BACKGROUND RESEARCH.....	10
2.1	INTRODUCTION.....	10
2.2	MOBILE DEVICE TYPES.....	10
2.2.1	STANDARD MOBILE DEVICES.....	10
2.2.2	PDAs.....	10
2.2.3	SMART PHONE DEVICES.....	11
2.2.4	CONCLUSION.....	11
2.3	DEVELOPMENT ENVIRONMENT.....	12
2.4	WIRELESS WEB.....	13
2.5	DATABASE.....	13
2.6	HUMAN COMPUTER INTERACTION CONCERNS.....	14
2.6.1	LEARN-ABILITY.....	14
2.6.2	FLEXIBILITY.....	15
2.6.3	ROBUSTNESS.....	15
3	LITERATURE REVIEW.....	16
3.1	OVERVIEW.....	16

3.2	EPISURVEYOR.....	16
3.3	MOBILE RESEARCHER.....	17
3.4	FRONTLINE SMS.....	17
3.5	GENERAL COMPARISON MATRIX.....	18
3.6	COMPARED WITH MOBICOLLECT.....	19
<b>4</b>	<b>ANALYSIS AND DESIGN.....</b>	<b>20</b>
4.1	INTRODUCTION.....	20
4.2	FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS.....	20
4.2.1	FUNCTIONAL REQUIREMENTS.....	20
4.2.2	NON-FUNCTIONAL REQUIREMENTS.....	21
4.3	OOAD.....	22
4.3.1	UML.....	22
4.3.2	USE-CASE DIAGRAMS.....	23
4.3.3	CLASS DIAGRAM.....	26
4.3.4	ACTIVITY DIAGRAMS.....	27
4.3.5	SEQUENCE DIAGRAMS.....	29
4.3.6	COLLABORATION DIAGRAMS.....	31
<b>5</b>	<b>IMPLEMENTATION.....</b>	<b>33</b>
5.1	INTRODUCTION.....	33
5.2	PLAN AND METHODOLOGY.....	33
5.3	DESIGN ADDITIONS.....	34
5.4	WEBSITE INTERFACE.....	35
5.5	PROBLEMS ENCOUNTERED.....	36
5.5.1	AJAX CONTROLS DESIGN ISSUES.....	37
5.5.2	LACK OF VALIDATION SUPPORT.....	37
5.5.3	UNAVAILABILITY OF TOOLS FOR TESTING MOBILE WEB PAGES.....	38
<b>6</b>	<b>TESTING.....</b>	<b>39</b>
6.1	INTRODUCTION.....	39
6.2	OPTIMAL PERFORMANCE.....	39

6.3	FUNCTIONAL TESTING AND EVALUATION.....	40
6.3.1	LOGIN.....	40
6.3.2	REGISTRATION.....	41
6.3.3	FORGOT PASSWORD.....	42
6.3.4	HOMEPAGE.....	42
6.3.5	CREATE FORM.....	43
6.3.6	CREATE COLLECTOR.....	46
6.3.7	VIEW COLLECTORS.....	47
6.3.8	DATA COLLECTION.....	47
6.3.9	VIEW FORMS.....	48
6.3.10	EXPORT DATA.....	49
6.3.11	ACCOUNT PAGE.....	50
6.4	SUMMARY.....	50
7	CONCLUSION AND FUTURE WORK.....	51
7.1	PROJECT ACHIEVEMENTS.....	51
7.2	CONCLUSION.....	51
7.3	RECOMMENDATIONS FOR FUTURE WORK.....	52
	REFERENCES.....	53

# CHAPTER 1

## INTRODUCTION

---

### 1.1 Overview

Mobile data collection is simply the process of using a mobile to record data. Typically, the process occurs while the user is out "in the field", away from an office or central location, where access to a main system or database is limited.

While the concept of using Mobile phones for surveys and data collection has been around for some years now, its usage hasn't been very widespread until the recent drop of mobile phone prices.

Once limited to large corporations and businesses using PDA (Personal Digital Assistants) data collection technology, is now affordable for everybody including students, researchers and small business owners - anybody using today's mid-range mobile phones not just the comparatively expensive PDAs.

### 1.2 Problem statement

The usage of traditional paper-based forms to collect data has several drawbacks when it comes to quantitative data collection.

They are expensive, inefficient, and inflexible; moreover require costly and error-prone data entry.

### 1.3 Project Aim

The aim of this project is to overcome the paper-based data collection problems by the usage of mobile technology.

This is to be achieved by development of a web application that allows user to design and manage forms that are on the fly connected to the organization's database, and uploaded to mobile optimized web pages.

Main services where the project can show significance are: forms, questionnaires, surveys and polls to be used by any organization requiring remote data collection.

#### 1.3.1 Objectives

Develop a web application for the research team to work on for the design of forms/questionnaires and data management collected from the responses.

The designed forms should be made available on mobile web pages to the data collectors; the data collected is stored in the corresponding database table created together with the mobile web page (as shown in figure 1.1)

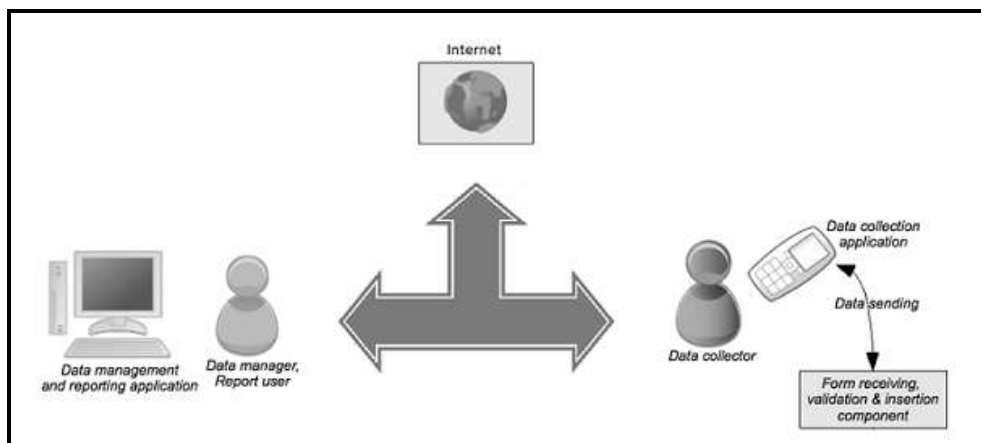


Fig 1.1 – System Architecture



### **1.3.2 Scope of the Project:**

The purpose of this project is to serve data collection that is primarily quantitative in nature and is collected by remote capturers / fieldworkers or a considerably big number of survey participants.

It does not aim at collecting qualitative data such as complex diagrams, long text responses – for which paper based collection would be more appropriate.

### **1.4 Future Enhancements**

The current system will allow the user to capture textual data – entered on the mobile web browser. In future releases of MobiCollect some noteworthy improvements to be expected are:

- Allow uploading Photos from the phone's memory or taken via camera.
- GPS Data can also be obtained using particular mobile's internal GPS feature.
- The form creator interface in the website can be enhanced to implement drag and drop features.
- Allow user to select fields in existing database tables of the organization to insert data.
- Allow editing existing forms and backing up data already collected by them.

### **1.5 Deliverables**

At the end of this project, the below mentioned materials will be delivered:

- Working Software Implementation
- Project Thesis

## 1.6 Methodology

Object-oriented technology will be used for developing MobiCollect. However, the development will not be based on any particular methodology.

## 1.7 Tools and Techniques

- Microsoft Visual Studio 2008 will be used to develop the front-end for MobiCollect using ASP.NET 3.5 with C# as the programming language.
- MS SQL Server 2008 Express will be used to develop the back-end for MobiCollect
- Sparx Systems Enterprise Architect 7.5 will be used for UML 2.0 modeling.

## 1.8 Initial Project Schedule

The below figure 1.2 illustrates the tentative initial project schedule to be followed.

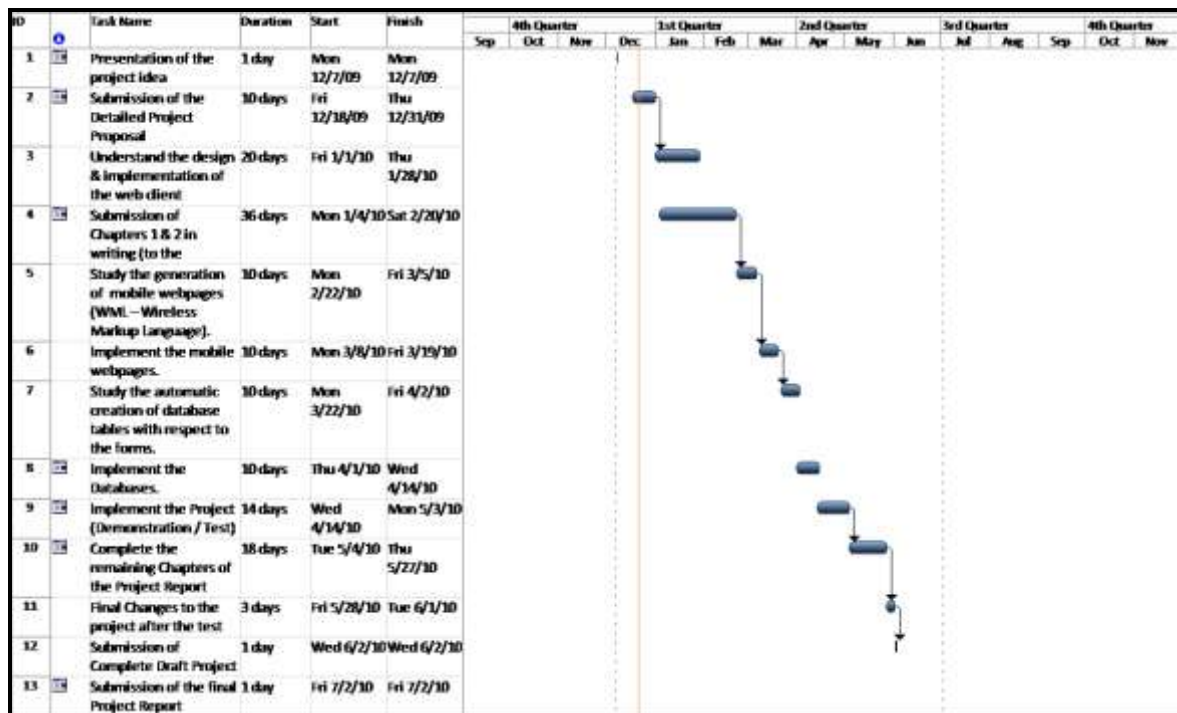


Fig. 1.2 – Initial Project Schedule

## **Chapter 2**

### **BACKGROUND RESEARCH**

---

#### **2.1 Introduction**

The following sections: Mobile Device Types, ASP.NET 3.5, MS SQL Database 2008, and Wireless Web are background topics with regard to the project.

#### **2.2 Mobile Device Types**

This section gives a very general overview on the differences amid the different classes of mobile devices.

##### **2.2.1 Standard Mobile Devices**

In simple terms it is a long range portable device for telecommunications, with capabilities that vary greatly between manufactures and models, some are sophisticated and provide various features while others are offered at low price range while providing the basic features. Some standards have been adopted by manufacturers, such as embedded Java support, and standards for sending/receiving various types of media, such as streaming video and graphical formats. The display resolutions as well as the operating systems for mobile phones vary greatly.

### **2.2.2 PDA**

PDA (personal digital assistant) is a term for any small mobile hand-held device that provides computing and information storage and retrieval capabilities for business or personal use, often for keeping schedule calendars and address book information handy. Some PDA devices offer a variation of the MS Windows operating system called Windows CE, and offer similar functionality to the standard desktop version. [1]

### **2.2.3 Smart Phone Devices**

A smart phone can be defined as a mobile device that integrates the functionality of a mobile phone with that of a PDA or other type of computer device. [2] The device usually carries an operating system such as Symbian or Microsoft's Windows Mobile. Devices of this nature also tend to have additional abilities such as WLAN connectivity, data entry via touch screen/stylus, additional third party applications can be easily installed and have a better screen resolution. Smart phones also tend to carry greater processing power and storage capability which give third party software developers great freedom and flexibility when designing applications.

### **2.2.4 Conclusion**

The key challenging task for generating the mobile web-pages is due to the limitations and numerous variations inherent to standard mobile devices. This concern is less relevant to PDA and smart phone devices that allow for great compatibility due to the standardisation of operating systems conceded on a large percentage of devices. There are various different screen resolutions present;

MobiCollect will be targeted at supporting the widespread resolutions: 128x160, 176x220 and 240x320 pixels (as shown in figure 2.1 from Chapter 4 of *DotMobi Mobile Web Developer Guide*). [3]

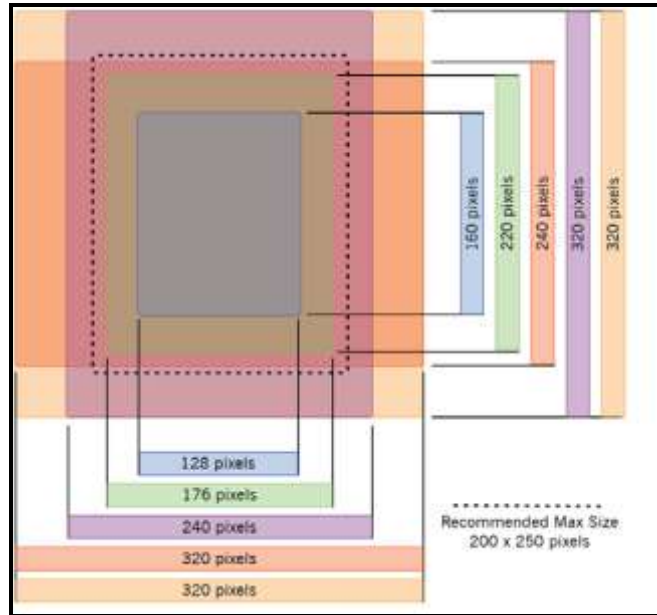


Fig. 2.1 Screen Resolutions

## 2.3 Development Environment

This section gives a brief look into the development environment.

ASP.NET is a web application framework developed and marketed by Microsoft to allow programmers to build dynamic web sites, web applications and web services. [4]

.NET pages, known officially as "web forms", are the main building block for application development. Web forms are contained in files with an ".aspx" extension; these files typically contain static (X)HTML markup, as well as markup defining server-side Web Controls and User Controls where the developers place all the required static and dynamic content for the web page. It allows and recommends using the dynamic program code i.e. the code-behind model, which places this code in a separate file or in a specially designated script

tag. Code-behind files typically have names similar to the content file (but with the final extension denoting the programming language used - .cs or .vb).

## 2.4 Wireless Web

Wireless Application Protocol – A WAP gateway produces a more efficient representation of the Web content that can be more easily transmitted over wireless networks (as shown in figure 2.2). It has a very low airtime cost to transmit the data using GPRS, EDGE or 3G connection.

Recently most of the smart mobile phones have wifi support, which enables the mobile to make use of any available wireless networks in the range. Would work out cheap connection as well but the mobile is comparatively more costly than the standard mobiles. [5]

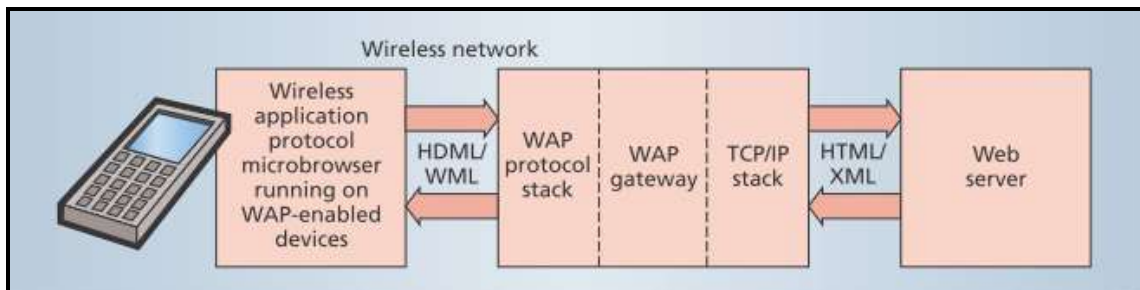


Fig 2.2 – Wireless Web

## 2.5 Database

A database can be understood as a collection of related files. How those files are related depends on the model used. The relational database model was a huge step forward, as it allowed files to be related by means of a common field. In order to relate any two files, they simply need to have a common field, which makes the model extremely flexible. [6]

SQL Server Express is free for development and production for all users, including individual developers and organizations. It allows the development of high-performance applications that take advantage of the security, reliability, and scalability of the SQL Server Database Engine.

## **2.6 Human Computer Interaction concerns**

This section studies the HCI measures in order to ensure optimum system usability. Full details on interface design and testing approaches will be explored in the later sections in this report. In terms of background research into HCI this section briefly outlines of three principles of usability – Learn-ability, Flexibility and Robustness, although only aspects that are significant to the interface will be listed. The HCI principles stated above will be taken into account for part of both the design and evaluation stages of the web application. [7]

### **2.6.1 Learn-ability**

How easily can a new user learn how to use the system? This involves many factors as follows:

- Predictability – Is the result of any user action predictable?
- Synthesisability – Is there any feedback from the system to confirm if a task is complete or not?
- Familiarity – Is the system familiar to a user from their previous experiences?

## **2.6.2 Flexibility**

Does the interface support a variety of interaction styles?

- User initiative – Is the user able to initiate any valid action whenever they desire? An issue of who is in control, the user or the machine. This is not incredibly relevant to the interface but worth mentioning for completeness.

## **2.6.3 Robustness**

Does the system give back adequate feedback to the user to confirm what is happening?

- Observable – Does the system display enough information to allow the user to know what is going on?
- Responsiveness – Does the system respond to user actions in a reasonable amount of time?



## **CHAPTER 3**

### **LITERATURE REVIEW**

---

#### **3.1 Overview**

Today, mobile phones are such a familiar part of our lives that we don't only use them for simply talking or texting others but in many other aspects like entertainment, work or surfing internet.

We take it for granted that we can talk to other people at any time, from wherever we may be; we are beginning to see it as normal that we can access information, take photographs, record our thoughts with one device, and that we can share these with our friends, colleagues or the wider world.

Moreover, as the technology advances the mobile content keeps getting richer and more easily shared over the internet this leads to the new concepts of using mobile phones for various applications. The rest of this chapter discusses the stand of the art project which uses mobile technology as a data collection tool.

#### **3.2 EpiSurveyor**

DataDyne's EpiSurveyor is an open-source mobile technology software program that enables public health workers to easily create handheld data entry forms, collect data on mobile devices and transfer the information back to a desktop or laptop for analysis. For this innovation and its impact on the developing world, the Lemelson-MIT Program has awarded EpiSurveyor the 2009 \$100,000 Lemelson-MIT Award for Sustainability, an award that recognizes and supports inventors or innovators whose work enhances economic opportunities and community well-being. Officially established as an electronic data collection standard by the World Health Organization, EpiSurveyor is now the most widely adopted open source mobile health software in the world. [8]

### **3.3 Mobile Researcher**

A recent entry into the mobile data collection market, Mobile Researcher is an end-to-end data collection service rather than a user-managed application. Mobile Researcher handles all the system configuration and data management - all you need to do is choose your options, train your data collectors and then sit back and wait until you have enough data to export for analysis.[9]

This software-as-service model means that you pay no setup costs, but instead are charged per completed form submitted to the system (using 'credits' bought from the company). You are also responsible for data transmission costs from the sender's phone, using either SMS or the much cheaper GPRS options. [9]

### **3.4 Frontline SMS**

At Stanford University a group of students founded FronlineSMS:Medic, an organization aimed “to advance healthcare networks in underserved communities using innovative, appropriate mobile technologies.” For this purpose the organization developed a free, open-source software platform to enable “large-scale, two-way text messaging using only a laptop, a GSM modem, and inexpensive cell phones.”[10]

The FrontlineSMS software installed on a laptop, it allows sending and receiving of SMS messages to any mobile phone. It essentially transforms any laptop into a mobile phone itself, acting as a hub for messages from multiple users with cell phones. [10]

### 3.5 General Comparison Matrix

Table 3.1 below compares the currently existing famous mobile data collection services [10]:

**Table 3.1 – Mobile Data Collection Services Comparison Matrix**

	<b>Developed By</b>	<b>Runs on</b>	<b>Service Model</b>	<b>System Components</b>
<b>Java Rosa</b>	Java Rosa open source community, Open Rosa Consortium	All bit very low-end Java phones	Free/Open source	Client and data transmission, forms designer and data analysis planned for late 2009
<b>RapidSMS</b>	UNICEF and open source community	All phones, Java not required	Free/Open source	End-to-end
<b>FrontlineSMS</b>	Kiwanja.net	Java phones including low end	Free to NGOs, open source (except for forms client)	End-to-end
<b>Mobile Researcher</b>	Clyral	All Java phones	Hosted solution / pay	End-to-end
<b>EpiSurveyor</b>	Datadyne	Java phones / Smartphones	Free, open source (desktop app) / free and paid, proprietary (web based)	Data collection, transfer, forms designer, GPS
<b>Nokia Data Gathering</b>	Nokia	Nokia Smartphone	Free for NGOs	End-to-end, GPS
<b>MobiCollect</b>	UMST - Shripal	All GPRS (wap) enabled phones	Academic Project	End-to-end, web-based forms designer, data collector accounts, view and export data

### **3.6 Summary**

Comparing the existing services with MobiCollect (as shown in Section 3.5), the main difference is the tools that are used for the development.

The interface of the Web Application at the server resembles the Episurveyor. However, both are totally different in terms of implementation where Episurveyor is built based on the Javarosa while MobiCollect will be implemented as ASP.Net Web Application with back-end support of the MS SQL 2008 Express. At the data collector end – Episurveyor requires the installation of Java Application on the mobile phone which can then be used to fill the forms after searching over the internet and retrieving them whereas MobiCollect uploads the forms to the web server as .NET mobile web pages that can simply be accessed using the mobiles' in built WAP-browser/Web-browser.

## **CHAPTER 4**

### **ANALYSIS AND DESIGN**

---

#### **4.1 Introduction**

This section outlines the functional and non-functional requirements and describes the system's scenarios using UML diagrams (use cases, activity, class, collaboration and sequence diagrams where appropriate).

This chapter also describes and justifies the system architecture solution and the design of each part of the system developed.

#### **4.2 Functional and Non-Functional Requirements**

Functional Requirements define what function a system or its component should perform. Functional Requirements are supported by Non-Functional requirements that define how the function should be performed i.e. they impose quality requirement/constraints on the implementation or design.

##### **4.2.1 Functional Requirements**

1. User can create a new form.
2. User can select the type for each question in the form.
3. User can view the data captured for a particular form.
4. User can browse existing forms.
5. User can search existing forms by keywords.
6. User can delete records from the data.
7. User can delete forms and their data.
8. User can retrieve form on mobile phone web-browser via internet.
9. User can fill the form from mobile phone browser.

10. User is authenticated using username and password before the web application client can be used.
11. Data collector is authenticated using username and password before being able to retrieve forms on the mobile phone.
12. User can change account details and password once logged in.
13. User can view date of each captured data.
14. User can view data collector's name for each record.
15. User can create new data collector accounts.

#### **4.2.2 Non-Functional Requirements**

1. Security:

The system must be secure from unauthorized access, meaning that parts of the system should only be available to users with a valid username and password. The data stored about the systems users must be protected so that only system administrators can view the stored data and the passwords must be stored as a hashed value so that if the database becomes exposed users' passwords are not revealed.

2. Usability:

The web-client should be as easy as possible to be used by the user and due to the diversity of mobile phones the mobile pages interface should be kept as simple and neat as possible. To avoid extra data transfer cost as well as difficulty in viewing the forms. Overall simplicity and user-friendly interface is required.

3. Scalability:

It is important that the system is scalable to expansion as more users use the system and the demand reaches a critical point where an expansion is needed in the future, the system should be scalable to this required expansion without the need of re-writing large portions of the system.

4. Data Integrity:

Tied in with the security element, the system has a number of data entry points, each of these points must be validated to ensure only correct valid data is every written to the database.

### **4.3 Object Oriented Analysis and Design (OOAD)**

OOAD is a software engineering approach that models a system as a group of interacting objects. Each object represents some entity of interest in the system being modeled, and is characterized by its class, its state (data elements), and its behavior. [11]

Object-oriented analysis (OOA) applies object-modeling techniques to analyze the functional requirements for a system. Object-oriented design (OOD) elaborates the analysis models to produce implementation specifications. OOA focuses on *what* the system does, OOD on *how* the system does it. [11]

#### **4.3.1 Unified Modeling Language (UML)**

UML is a standardized general-purpose modeling language in the field of software engineering. The standard is managed, and was created by, the Object Management Group. UML includes a set of graphical notation techniques to create visual models of software-intensive systems. [12]

UML has many different models/diagrams, with different viewpoints for the stakeholders (analysts, coders, testers, users etc...) of the software system.

UML aims to provide a language so expressive that all stakeholders can benefit from at least one UML diagram.

Some of the most important diagrams are:

1. The Use Case Diagram: How will the system interact with the outside world/user?
2. The Class Diagram: What objects do we need and how will they be related?
3. Activity Diagram: How will the activities be carried out?
4. Collaboration Diagram: How will the objects interact?
5. Sequence Diagram: How will the objects interact?

### 4.3.2 Use Case Diagrams

1. Administrator: able to login and create new forms, view existing forms' data, search through the forms and create new data collector accounts.

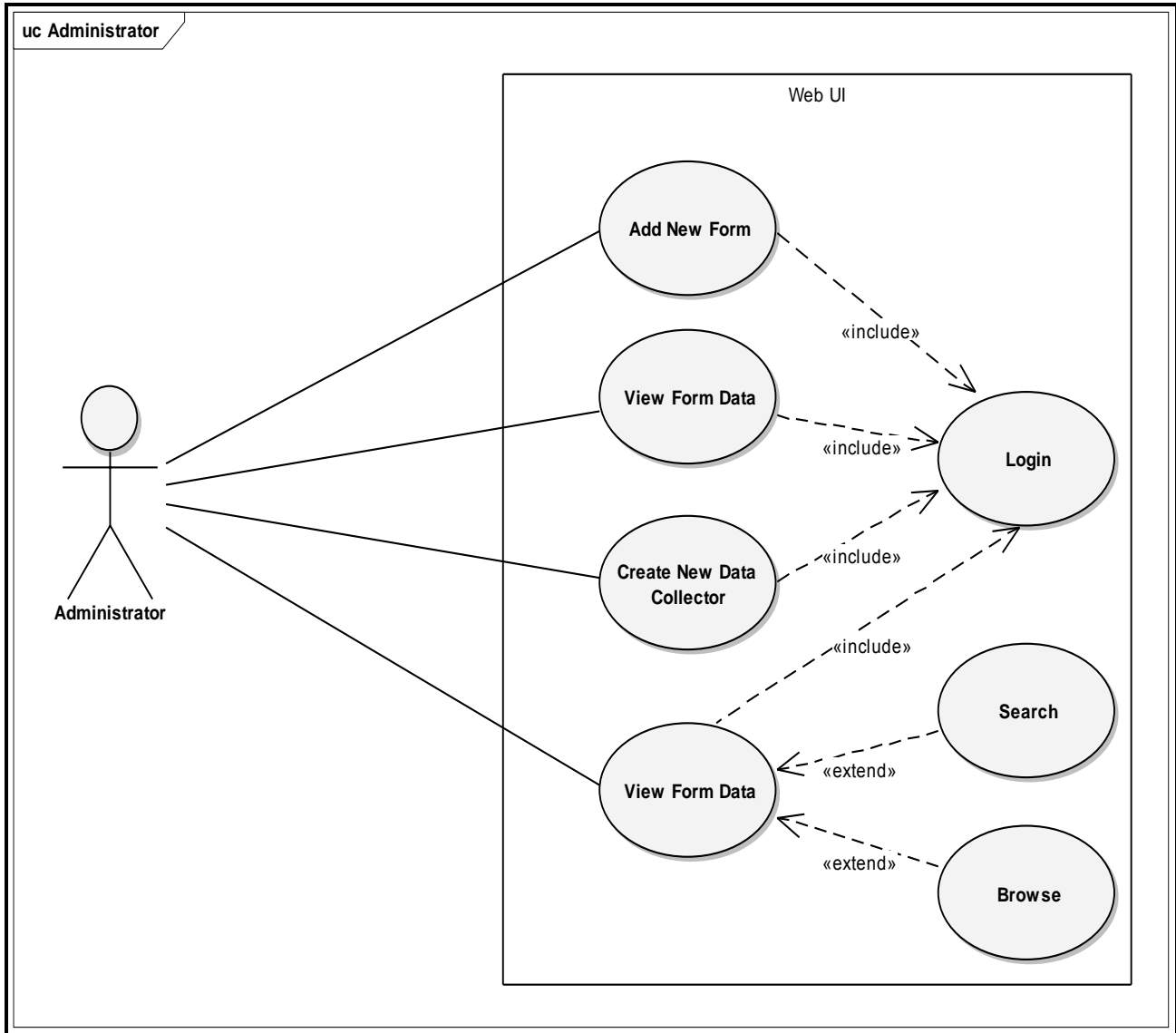


Fig 4.1 –Administrator Use Case

Use case above is showing the tasks performed by the Administrator which are: login, add new form, view form data, search form data and add new data-collector.



2. Web-Application System: interface to the administrator for the design and creation of the database table and mobile web-page for user defined form/survey. Hence it will be able to create new Database Tables and generate appropriate mobile web-page for the form.

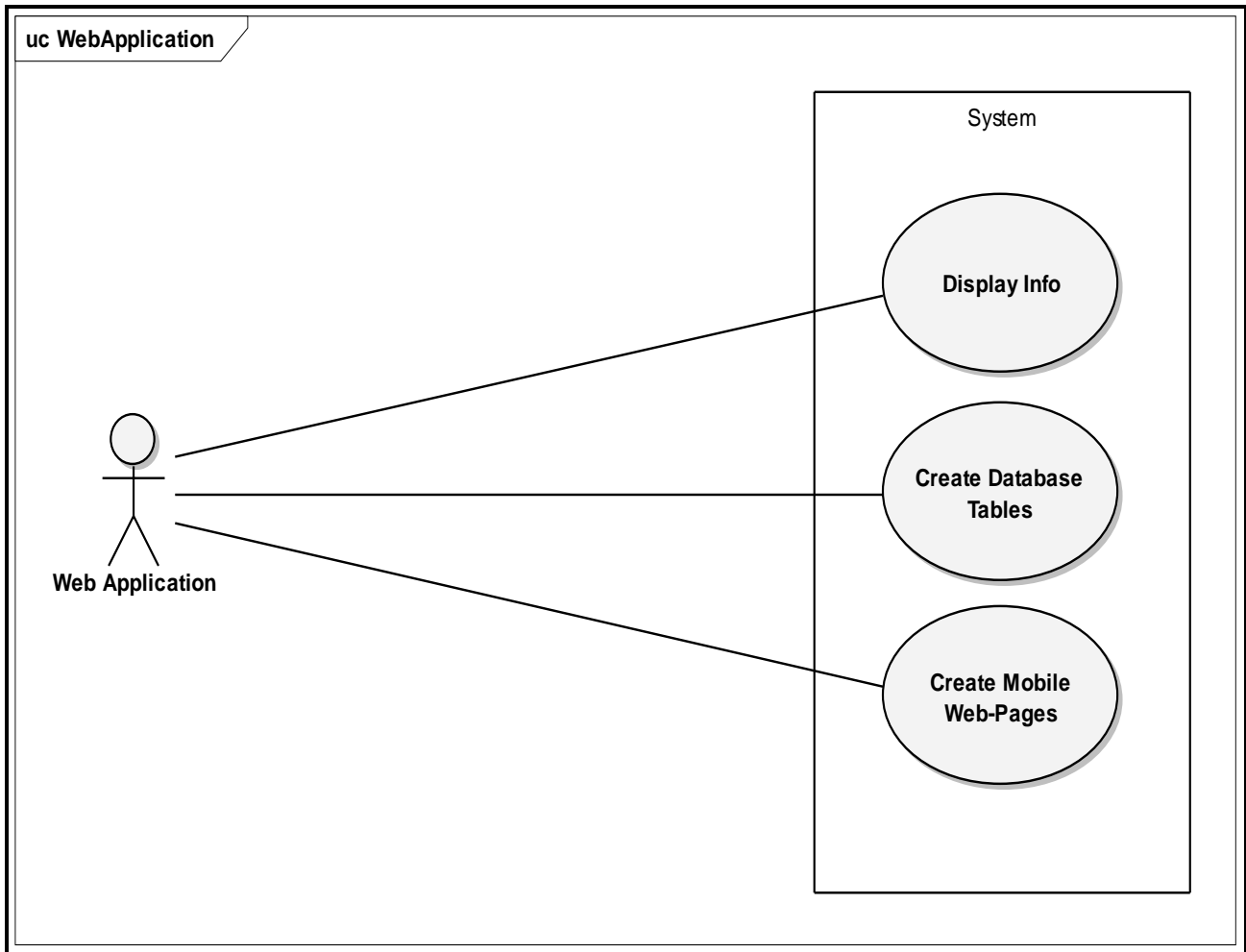


Fig 4.2 –WebApplication Use Case

Use case above is showing the tasks performed by the System which are: interact with Administrator (UI), Create Database Tables, and Create mobile web-pages.

3. Data Collector: login and browse forms, search forms by keywords, save data to the database.

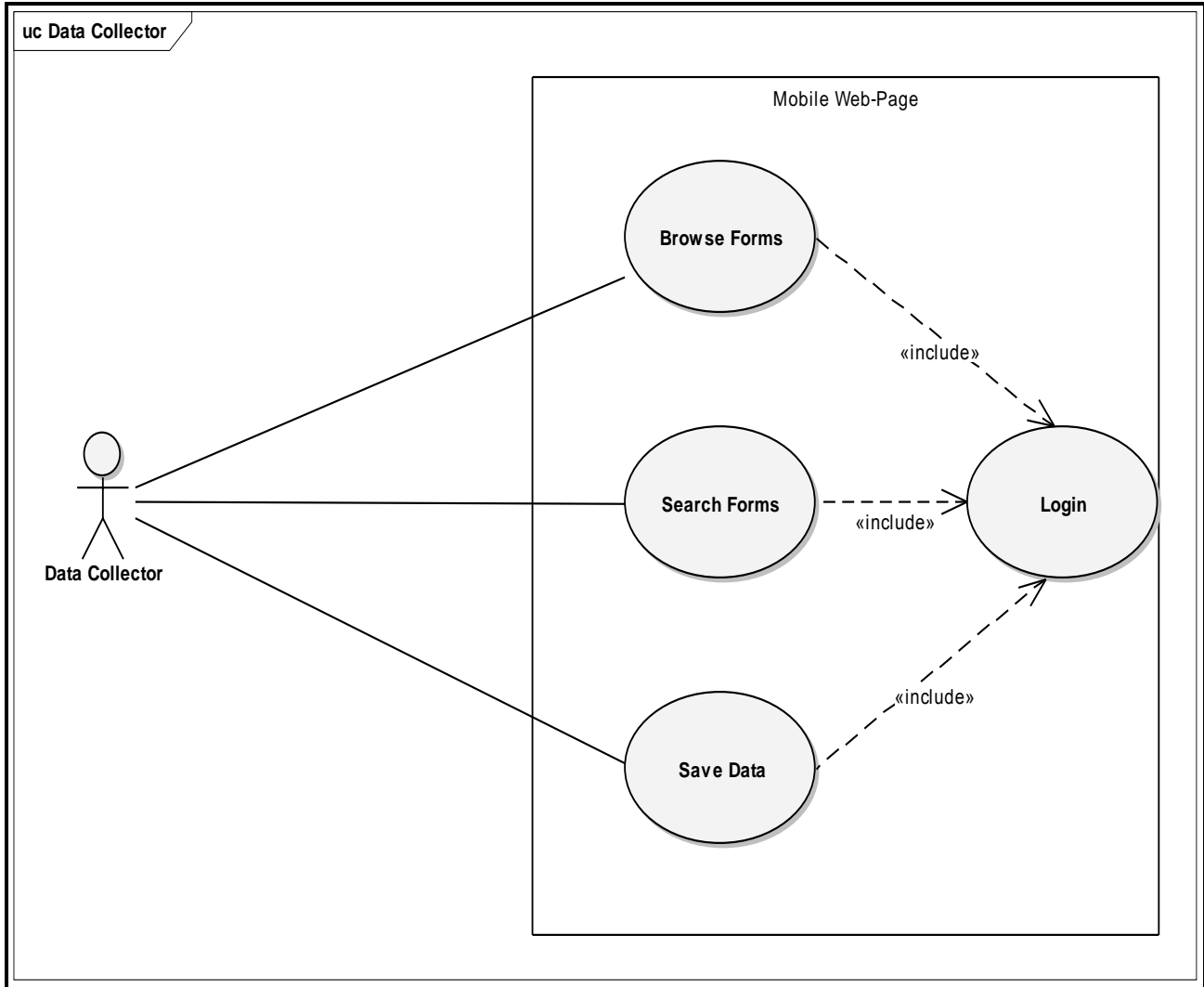


Fig. 4.3 – Data Collector Use Case

Use case above is showing the tasks performed by the Data-Collector (Using Mobile) which are: login to their account, browse/search forms, fill and save data.

### 4.3.3 Class Diagram:

The structure of the classes is as per the asp.net pages of the project (as shown in figure 4.4).

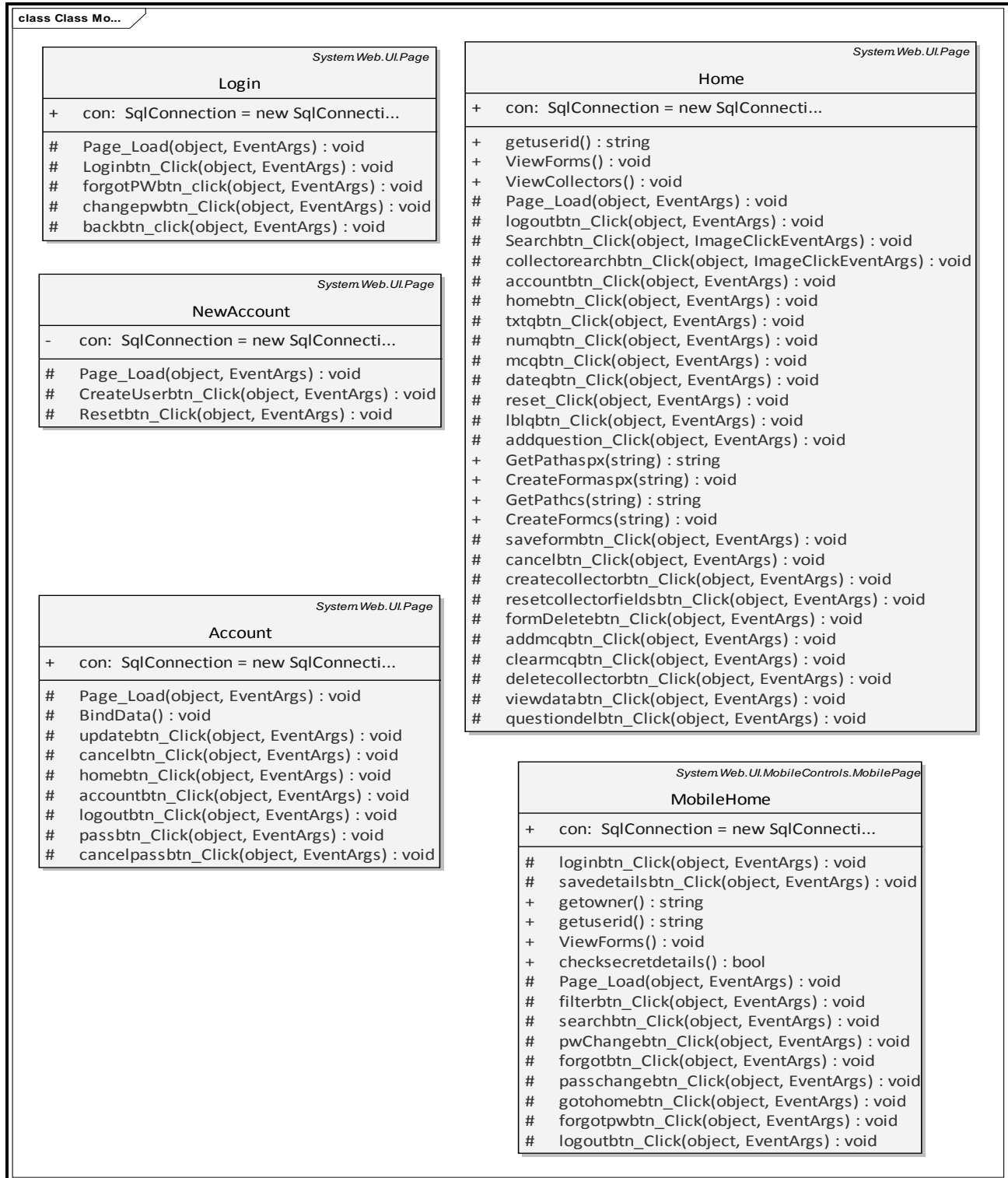


Fig. 4.4 – Class Diagram

### 4.3.4 Activity Diagrams

#### 1. Administrator Activity:

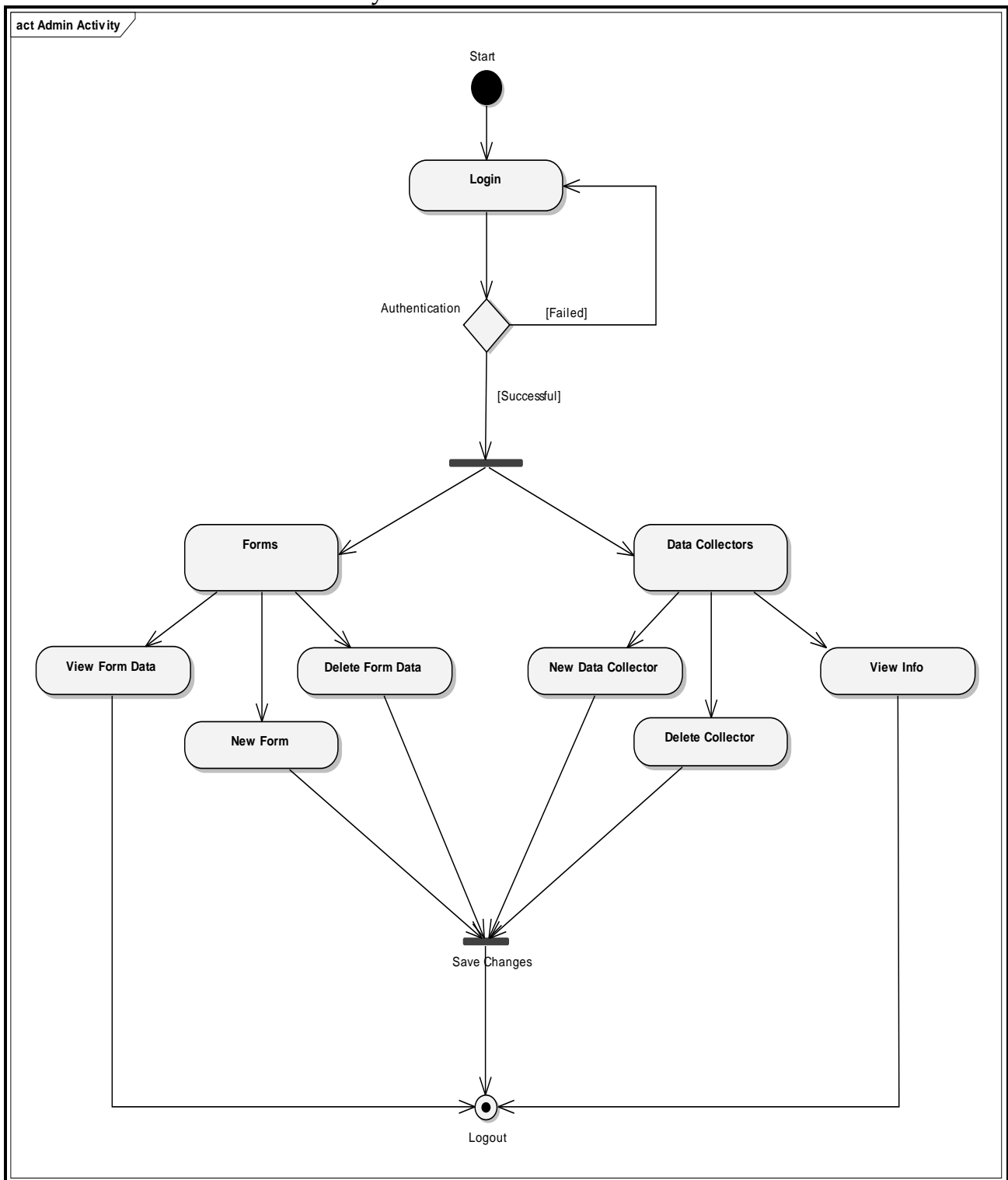


Fig. 4.5 – Administrator Activity

## 2. Data Collector Activity:

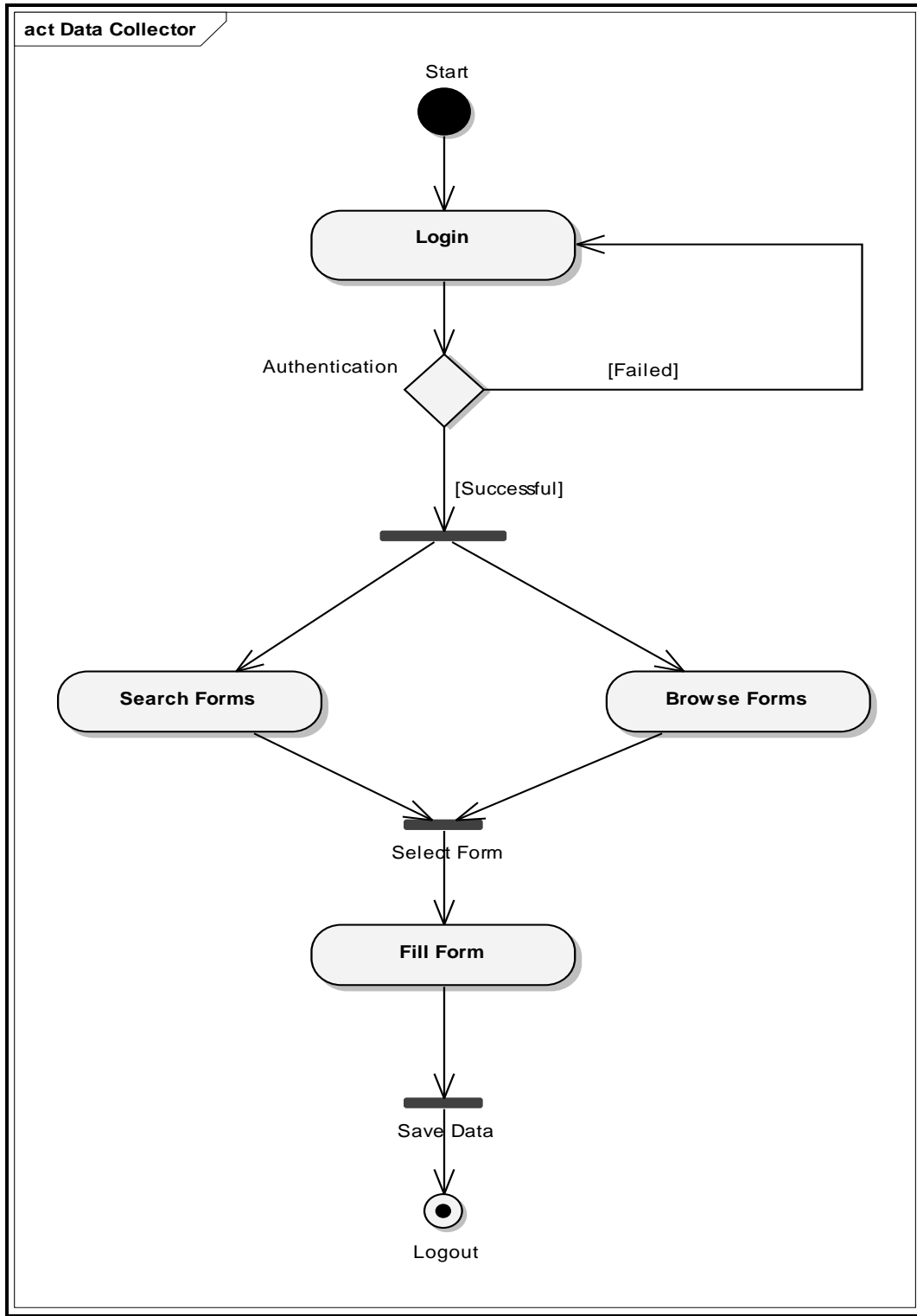


Fig. 4.6 – Data Collector Activity

### 4.3.5 Sequence Diagrams

#### 1. User Login:

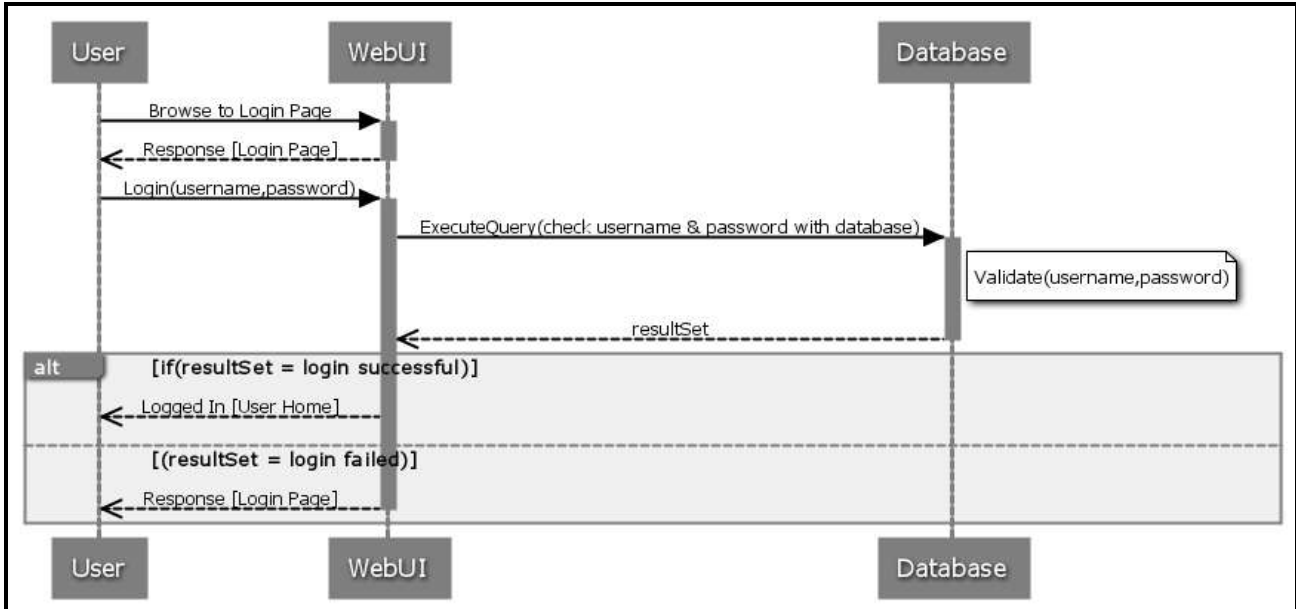


Fig. 4.7 – User Login Sequence

#### 2. Search Form Data:

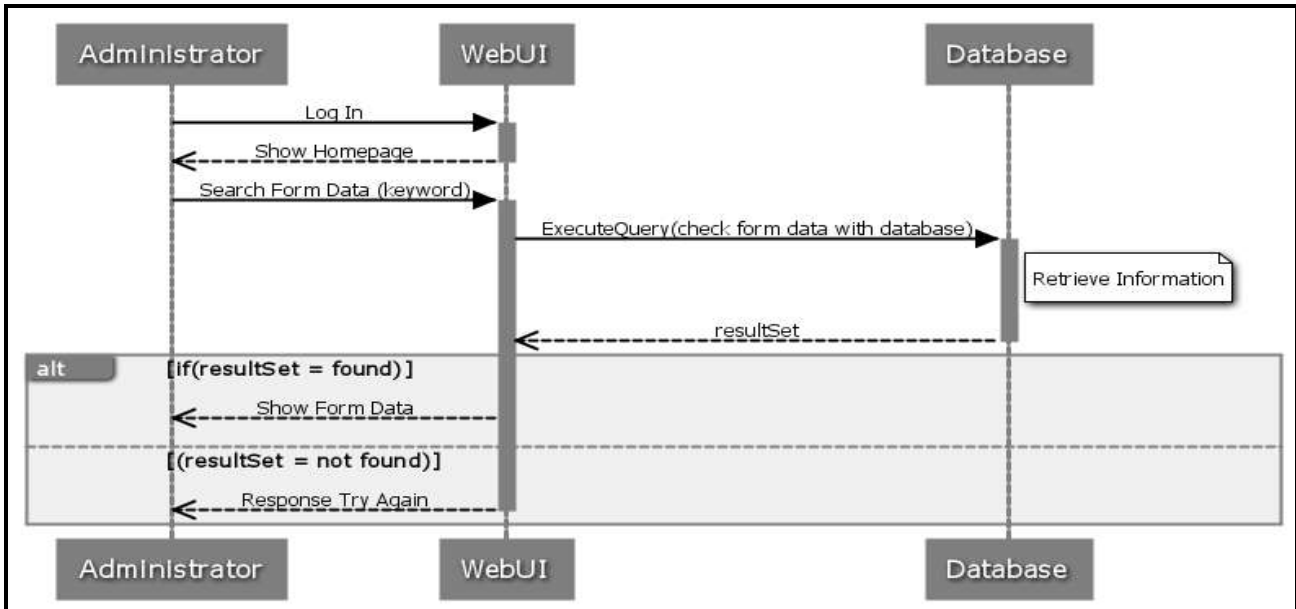


Fig. 4.8 – Search Form Data Sequence

### 3. Create Form:

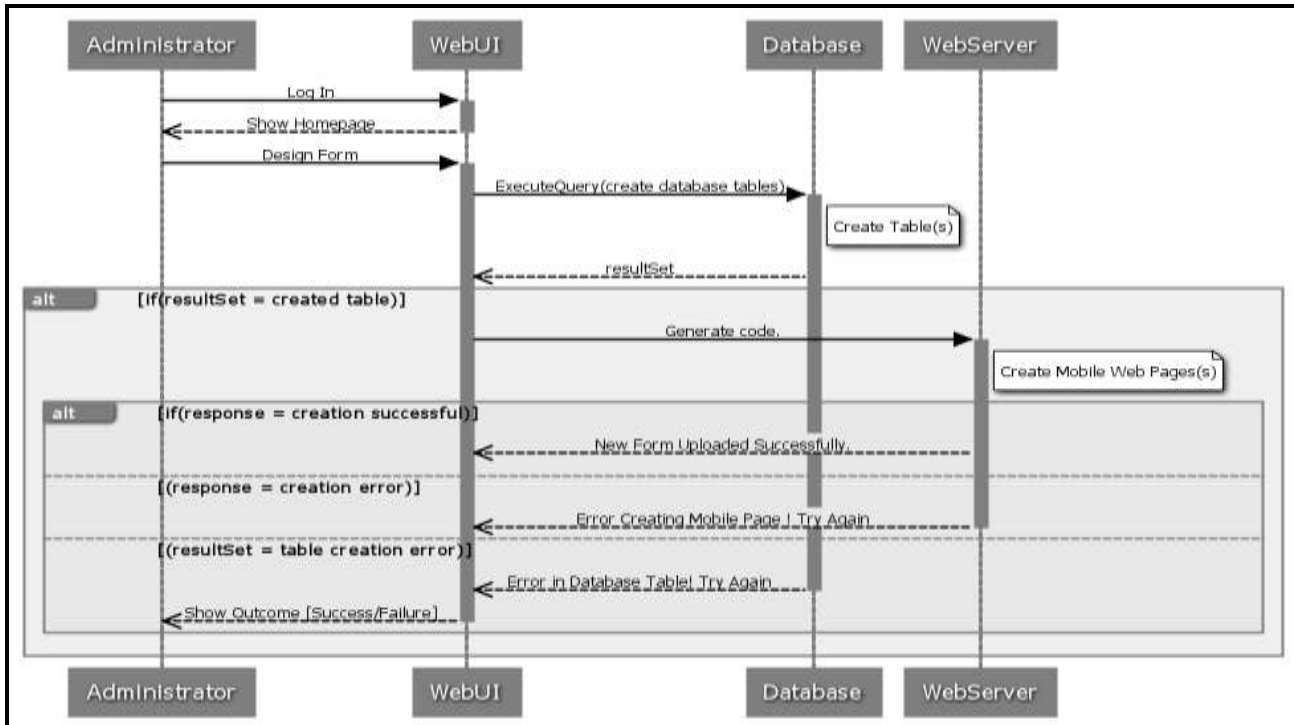


Fig. 4.9 – Create Form Sequence

### 4. Collect Data:

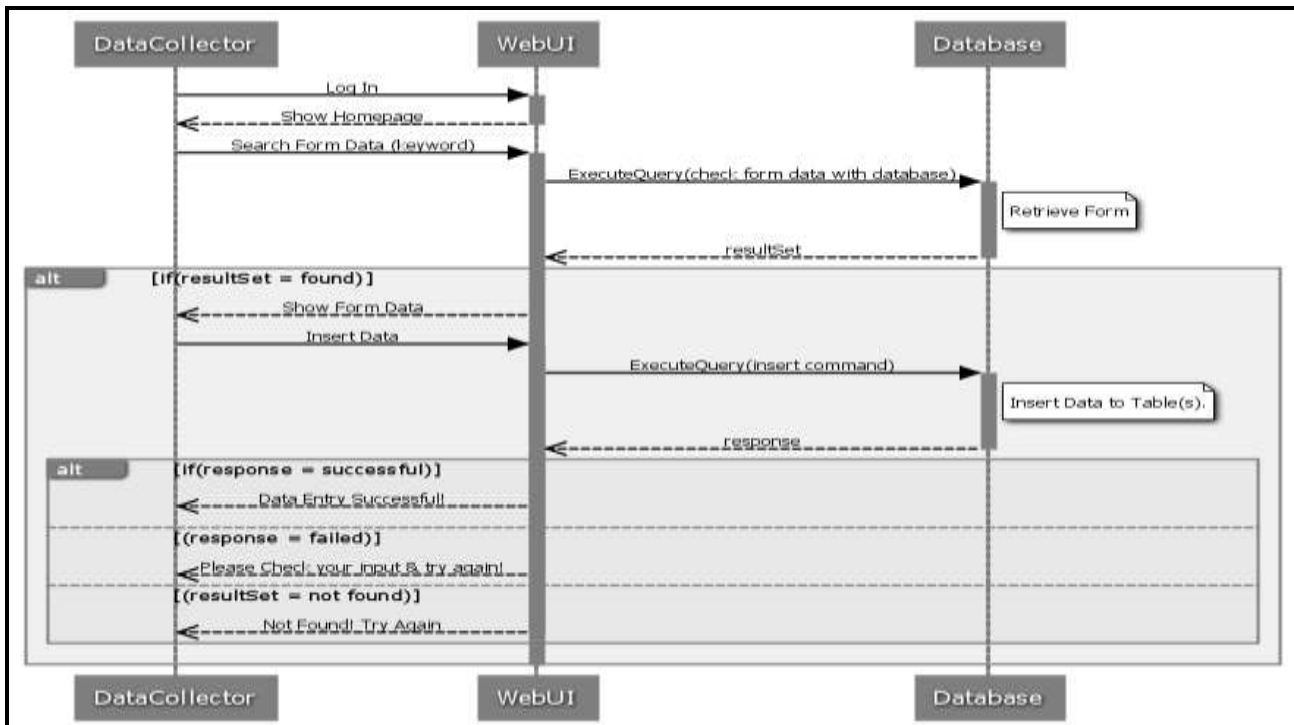


Fig. 4.10 – Collect Data Sequence

## 4.3.6 Collaboration Diagrams

### 1. User Login:

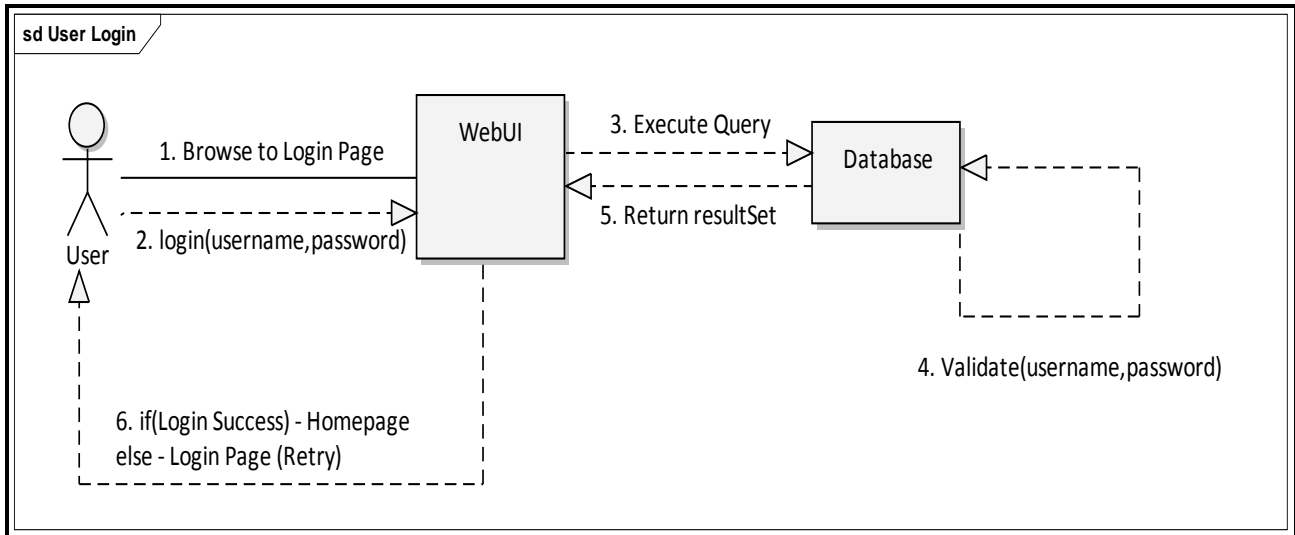


Fig. 4.11 – User Login Collaboration

### 2. Search Form Data:

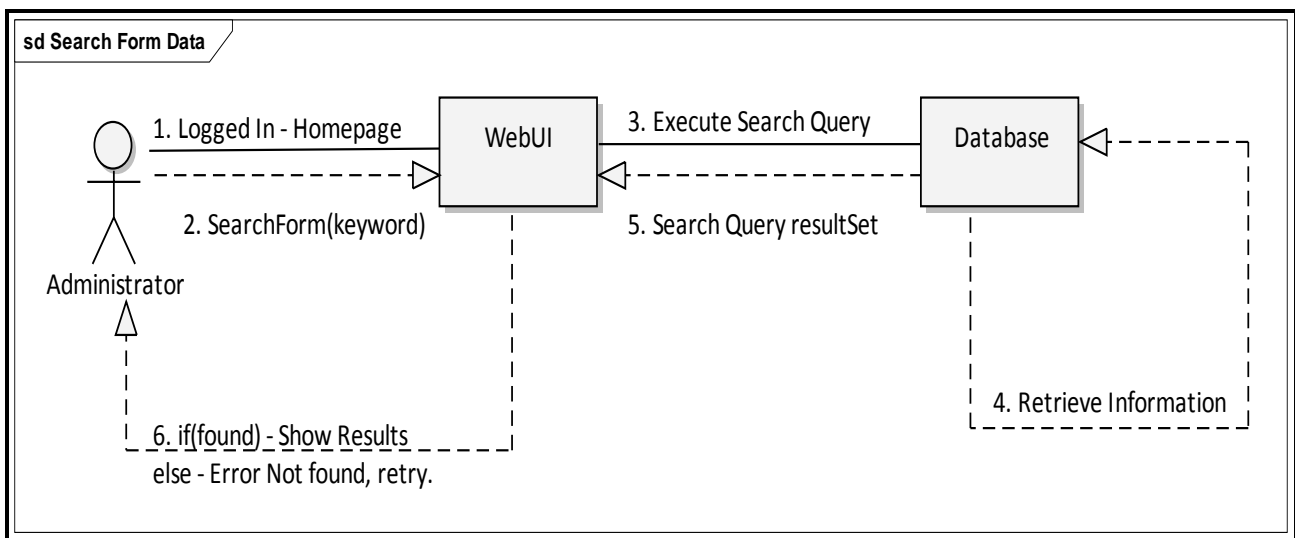


Fig. 4.12 – Search Form Data Collaboration



### 3. Create Form:

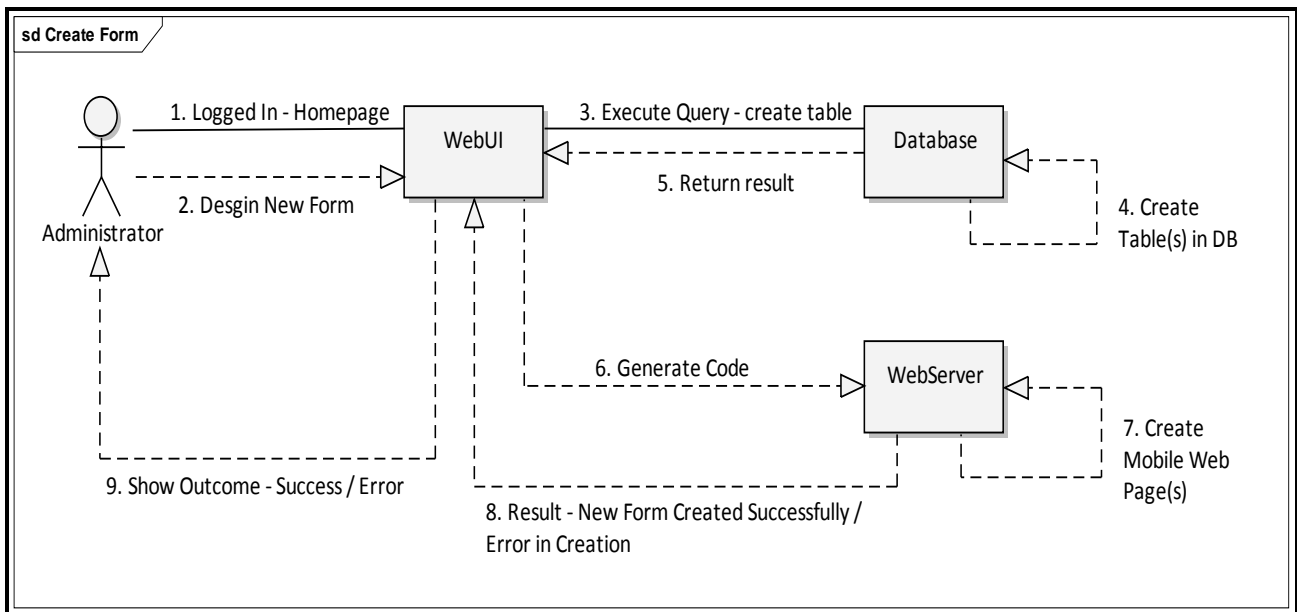


Fig. 4.12 – Create Form Collaboration

### 4. Collect Data:

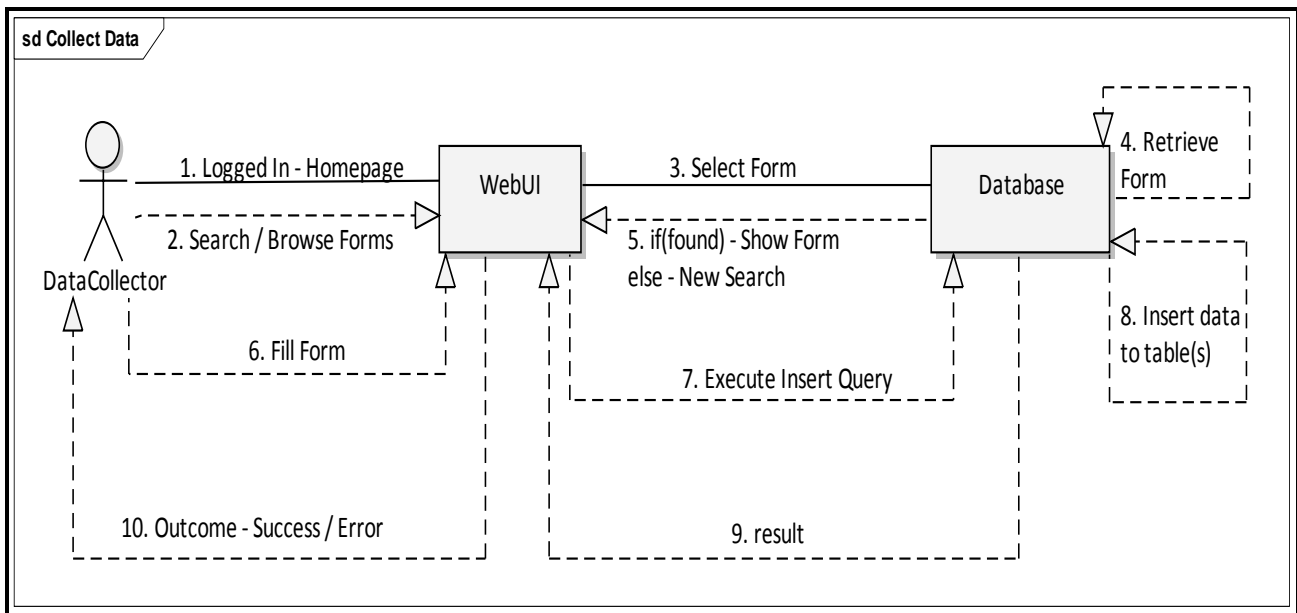


Fig. 4.13 – Collect Data Collaboration

## CHAPTER 5

### IMPLEMENTATION

---

#### 5.1 Introduction

This section discusses the implementation of the system, it is based on the system design in the previous chapter, but due to emerging requirements throughout the implementation iterations, the completed system contains a number of additions and differences in design.

#### 5.2 Plan and Methodology

In alignment with the mixture of Object Oriented and Incremental Development methodology that was used, the build was split into time boxes to gradually accomplish the project objectives, due to the incremental iterative nature of the development each time box was split into a number of iterations with most time allocated to the important (core) requirements of the system.

Once these requirements were fulfilled, the features were reviewed and constantly tested to debug any errors discovered and add additional features needed to be incorporated into the system hence keeping the development process synchronized with the newly emerging changes required.

**Table: 5.1 Time-boxes**

Time-box	Task	Time (approx.)
1	Website Design and Coding	20 days
2	Database Design	10 days
3	.NET Mobile Website	10 days
4	Website(.NET Mobile Code Generation)	30 days
5	Website (Database Tables Generation)	30 days

As illustrated in (Table 5.1), the first time box was used to develop the interface design of the website.

This includes master pages and CSS style sheets used for giving a common look and feel to all the pages of the website, and using AJAX-Control toolkit to make the use of the website as interactive as possible. This was followed by implementing the basic functionalities of the website (i.e. login, create account, view data, edit account information, log out).

The second time box was for the database design, the simplest part to implement due to the nature of the system, only required a couple of Database Tables: User Accounts and another for the Forms.

The third time box was for the implementation of web page for the mobile forms, this took lesser time compared to the website implementation due to simplicity of the design and limited features of mobile web pages to be used solely for filling up data into the forms. The basic features are: login, search, view forms, logout.

The fourth time box was used to implement the interface for Creating Forms and implementing the code generation for .NET Mobile (aspx pages) as well as the code behind (.cs files) for the forms designed, this was the most important feature of the project and was highly complex due to various options provided to the users for designing a form with different question types with the appropriate validation options.

The fifth time box was used to generate the appropriate DDL statements for the generation of corresponding database table to store the data collected by the form. Also the generation of insert command for updating the record into that

table once the collector clicks “save” on the form web page. This was equally important and complex as the web page code generation hence given the big effort time.

### 5.3 Design Additions

Some of the overlooked features of the website in the design stage were later implemented as general improvements on the system, these features include: forgot password/change password solution, editing personal account details, viewing form data on the website, option to delete form pages as well as database the table. The last addition, none the less a very important one was the option to export data of a form to an excel file. This feature is particularly useful for the analysis of the respective data. It has been implemented with the aid of GemBoxSpreadSheetFree .NET Component

### 5.4 Website Interface

This section describes the implementation of the website interface; it allows the user to use the data collection system by giving functionalities such as: view data, create forms and data collectors.



**Fig 5.1 Color Scheme**

Figure 5.1 above, shows the color scheme used for the entire website to maintain a consistent look.

The interface is based on AJAX Accordion Panes, figure 5.2 shows the interface screenshot for the create form feature in the home page of the website.

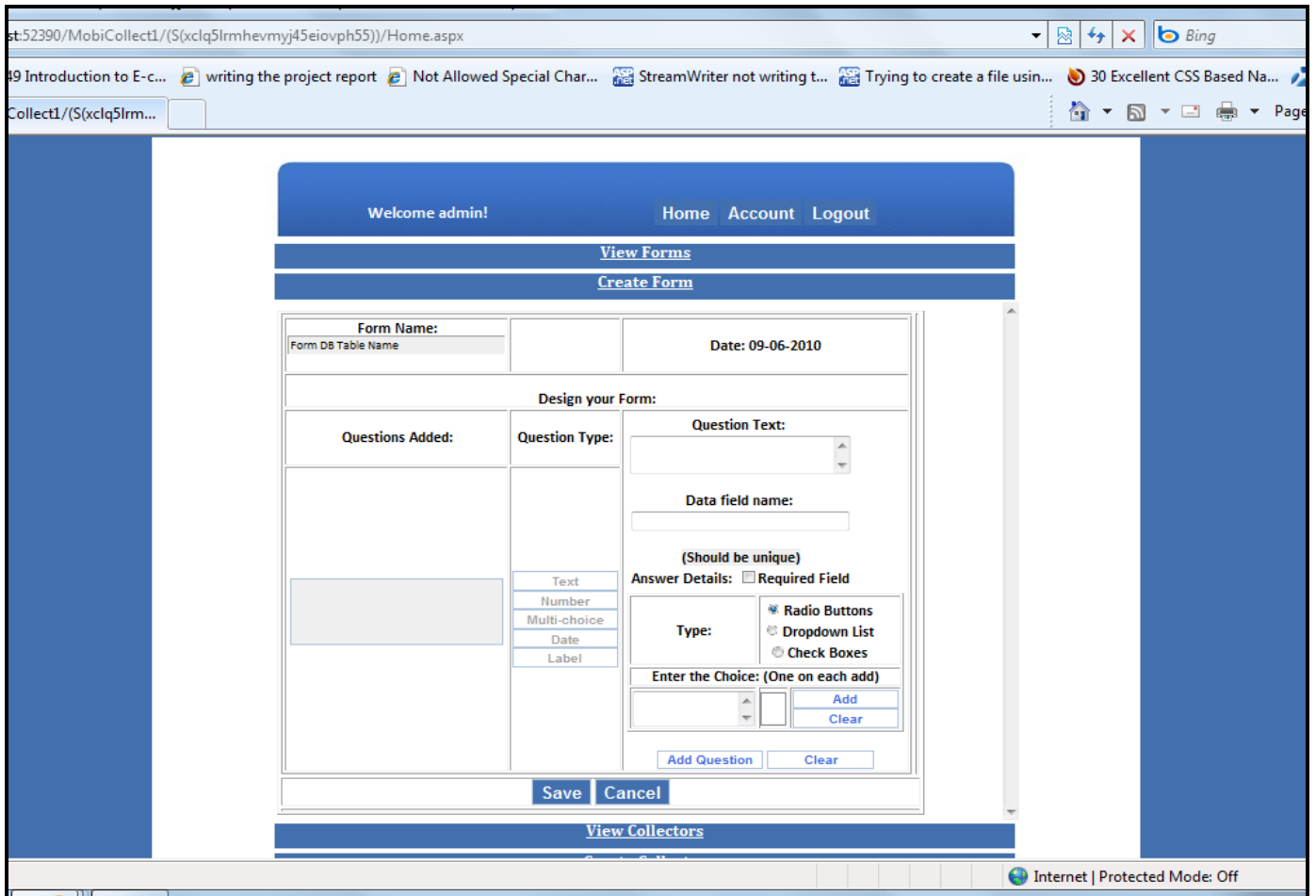


Fig. 5.2 Interface

## 5.5 Problems Encountered

During the implementation phase some problems were encountered, the following section outlines these issues. They differ in degree of difficulty and significance.

1. AJAX Controls design issues.
2. Lack of control validation options.
3. Lack of Visual Studio support for mobile web pages.

### **5.5.1 AJAX Controls design issues**

The first problem was due to the use of AJAX Control

“Accordion Panes” for implementing the website interface caused the drag and drop of controls from the VS toolbox to be non-functional.

Due to the complex nature of the website this feature could have been a great help in order to arrange and place the controls in the desired layout moreover even the customization of the controls from the IDE properties tab was not working, the only way to implement the content inside the accordion control was the use of code by hand.

This issue was a challenge rather than a problem, it was overcome and moreover it had its benefits:

- The layout was designed manually using tables and CSS.
- Gave me a better control over the controls and the various properties by manually writing everything.
- This issue greatly helped in the knowledge code generation of the .aspx pages for the mobile web pages by giving me a much better understanding of the controls than I did have already.

### **5.5.2 Lack of validation support**

Another problem encountered controls like calendar, selectionlist in the asp.NET Mobile framework do not support the built-in validation controls (required field validator, custom validator, regular expression and others).

The validation of the data collection forms was an integral requirement of the project, and hence the issue had to be resolved manually. There were a number of possible workarounds, solving the problem programmatically, create a user

control supporting the existing validation controls and is similar to the built-in control, create user defined validation controls.

The first workaround was used to overcome the issue as it required relatively less programming effort than the other two approaches.

The solution acted similarly to the required field validator, the form would not be submitted unless the corresponding required field had a value (using if condition).

### 5.5.3 Unavailability of tools for testing mobile web pages

The third problem faced was unlike Visual Studio 2005, the newer Visual Studio 2008 no longer supports design view for mobile web pages, i.e. no drag and drop from the controls toolbox and no preview available in the design view, the only way is to manually write the controls with the help of IntelliSense provided.

Moreover, Visual Studio did not provide any built-in simulator for viewing the web pages; the Microsoft support mentioned links to download the program but the all links mentioned were dead – [13]

The solution, after a lot of searching over the internet was to download the OpenWave v.7 simulator from a 3<sup>rd</sup> party website.



Fig. 5.3 Openwave V.7

## **CHAPTER 6**

### **TESTING**

---

#### **6.1 Introduction**

The purpose of the testing is the practical implementation of the web application and hence verifying if the project's requirements, aims and specification have been met.

Firstly the system configuration / requirements for the optimal performance of the system will be outlined.

Secondly all the features of the system will be used, demonstrating the output/result for each action. The testing process and its result are illustrated by the appropriate screenshots.

#### **6.2 Optimal Performance**

This section takes a look at the optimal performance requirements for the website; this is based on the observations from the testing of the website once the implementation was completed.

The key requirements observed during the testing are:

1. Java Script and CSS enabled Web Browser – (Internet Explorer 7)
2. Screen resolution – 1024 x 768 – (Recommended higher)
3. Microsoft SQL Server Express 2008
4. Internet enabled mobile phone (Java script enabled browser)



## 6.3 Functional Testing and Evaluation

In this section all the functions of the website are tested and the results are illustrated with appropriate screenshots.

### 6.3.1 Login

The default page to open when accessing the MobiCollect website is the login page, it functions as shown in (figure 6.1) below.

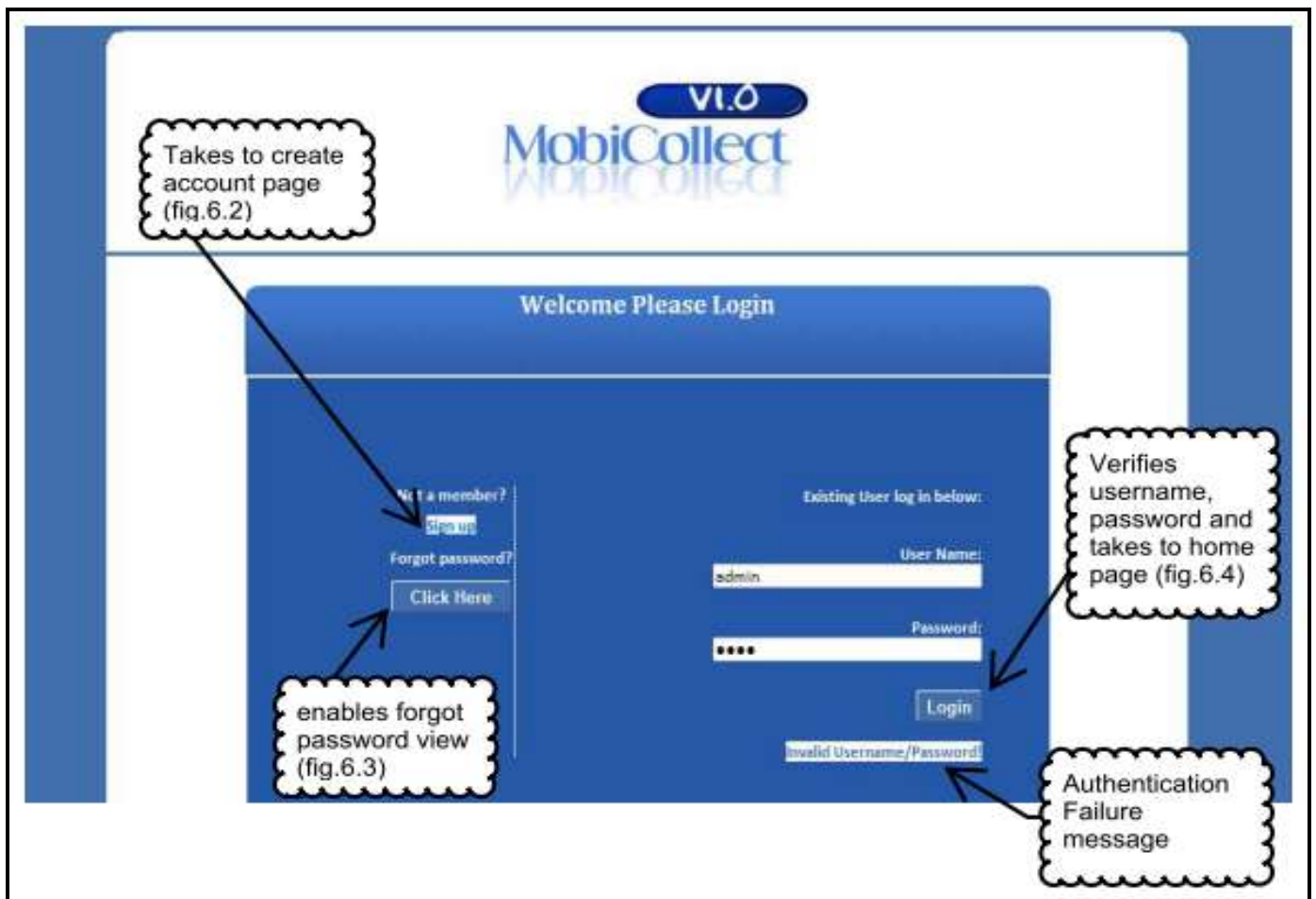


Fig. 6.1 Login Page

### 6.3.2 Registration

The page for signup as a new user for the website, it is validated appropriately; also checks and gives an error message if the username is taken already.

The registration page is shown in (figure 6.2 below).

The screenshot shows a registration form with the following fields and values:

Field	Value
First Name:	Shripal
Last Name:	Parekh
User Name:	testuser
E-mail:	shripal2006@gmail.com
Password:	••••••••
Confirm Password:	••••••••
Secret Question:	What was your first car? ▼
Secret Answer:	•••••
Captcha:	2K6 T6
Captcha Input:	2k6t6

Buttons: Create User, Reset

Validation Messages: A callout box labeled "Validation Messages" points to the right side of the form, indicating where error messages are displayed.

Fig. 6.2 Signup

The above test created a new user account for MobiCollect test purpose, this account will be used to login and continue the testing of the website.

**Account Details:** Username: *testuser*, Password: *testpass*

### 6.3.3 Forgot Password

The following screenshot (figure 6.3) shows the interface for helping a user reset their password in case forgotten.

The screenshot shows a web interface for password recovery. The main heading is "Forgot Password". On the left side, there is a "Mistaken?" label and a "Go Back" button. On the right side, under the instruction "Please enter the details below:", there are several input fields: "User Name", "Secret Question" (a dropdown menu currently showing "In what city where you born?"), "Secret Answer", "New Password", and "Confirm Password". A "Confirm" button is positioned at the bottom right of the form area.

Fig. 6.3 Forgot Password

### 6.3.4 Homepage

The next set of screenshots demonstrates the testing of the homepage (user is redirected upon successful login).

The screenshot displays the homepage for a user named "testuser!". The header includes a welcome message "Welcome testuser!" and navigation options: "Home", "Account", and "Logout". Below the header, there are four menu items: "Create Form", "View Forms", "Create Collector", and "View Collectors".

Fig. 6.4 Homepage

### 6.3.5 Create Form

In this section of testing a form with 1 question from each available type will be used to create "TestForm" (As shown in figures 6.5-6.10).

The screenshot shows the 'Create Form' interface. At the top, there is a blue header with the text 'Create Form'. Below the header, there are two input fields: 'Form Name:' with the value 'TestForm' and 'Date:' with the value '14-06-2010'. The main area is titled 'Design your Form:' and is divided into three sections: 'Questions Added:', 'Question Type:', and 'Information:'. The 'Questions Added:' section is currently empty. The 'Question Type:' section has a list of options: 'Label', 'Text', 'Number', 'Multi-choice', and 'Date'. The 'Label' option is highlighted with a blue background. An arrow points from a callout box labeled 'Question Type: Information Label' to the 'Label' option. The 'Information:' section contains a text area with the text 'This is a form created for testing purposes.' and two buttons: 'Add Question' and 'Clear'. At the bottom of the interface, there are two buttons: 'Save' and 'Cancel'. A callout box labeled 'Stores the question' points to the 'Add Question' button. Another callout box labeled 'Form & DB Table Name' points to the 'Form Name:' field.

Fig 6.5 Create Form – Label

The screenshot shows the 'Create Form' interface with the 'Text' question type selected. The 'Form Name:' field still contains 'TestForm' and the 'Date:' field still contains '14-06-2010'. The 'Questions Added:' section now contains one entry: 'This is a form created for testing purposes.' with a 'Remove Question' button below it. The 'Question Type:' section has the 'Text' option highlighted. The 'Information:' section has a 'Question Text:' field with the text 'What is your Name?'. Below this is a 'Data field name:' field with the text 'fName'. There is a checkbox labeled '(Should be unique)' which is checked, and a label 'Answer Details: Required Field' next to it. At the bottom, there are 'Add Question' and 'Clear' buttons. Callout boxes provide additional information: 'Question type: Text' points to the 'Text' option in the 'Question Type:' list; 'Validation Option' points to the 'Required Field' checkbox; and 'Already added questions' points to the 'Remove Question' button.

Fig 6.6 Create Form – Text

**Create Form**

Form Name: TestForm Date: 14-06-2010

**Design your Form:**

Questions Added: This is a form created for testing purposes. What is your Name? Remove Question

Question Type: Label, Text, Number, Multi-choice, Date

Question Text: What is your Age?

Data field name: TestAge

(Should be unique)

Answer Details:  Required Field

Min.: 18 Max.: 40

Add Question Clear

Save Cancel

Question type: Number

Validation Options

Fig 6.7 Create Form – Number

**Design your Form:**

Questions Added: This is a form created for testing purposes. What is your Name? What is your Age? Remove Question

Question Type: Label, Text, Number, Multi-choice, Date

Question Text: Select Gender:

Data field name: SEX

(Should be unique)

Answer Details:  Required Field

Type: Enter the Choice: (One on each add)

Female Male Add Clear

Add Question Clear

Save Cancel

Question Type: Multiple Choice

Radio (selected) / Checkbox Dropdown.

Possible Answer choices

Fig 6.8 Create Form – Multiple Choice

**Create Form**

Form Name: TestForm

Date: 14-06-2010

**Design your Form:**

Questions Added:

This is a form created for testing purposes.  
What is your Name?  
What is your Age?  
Select Gender:

Remove Question

Question Type:

- Label
- Text
- Number
- Multi-choice
- Date**

Question Text: Choose Reservation Date:

Data field name: rDate

(Should be unique)

Answer Details:  Required Field

Add Question Clear

Save Cancel

Question type: Date

Fig 6.9 Create Form – Date

**Create Form**

Form Name: TestForm

Date: 14-06-2010

**Design your Form:**

Questions Added:

This is a form created for testing pur  
What is your name?  
What is your age?  
Select Gender:

Remove Question

Question Type:

- Label
- Text
- Number
- Multi-choice
- Date

Save Cancel

Success!

Click when all questions added

Result: Success in creating the form

Fig 6.10 Form save – Success



### 6.3.6 Create Collector

This section is used for creating data-collector accounts by the user.

The form is shown in the below screenshot (figure 6.11).

The screenshot shows a web form titled "Create Collector" with the following fields and values:

First Name:	TestfName
Last Name:	TestlName
User Name:	testcollector
E-mail:	testing@mail.com
Password:	••••
Confirm Password:	••••

Below the fields are two buttons: "Create" and "Reset". To the right of the password field, the text "Strength: Average" is displayed in a yellow highlight. Two callout boxes with arrows provide additional information: one points to the "Create" button, stating "Creates a new collector account, that is used to login from Mobile for data collection"; the other points to the "Strength: Average" text, stating "Form validation messages".

Fig. 6.11 Create Collector

A collector account was created during testing in order to test this section as well as data collection features of the system by using the account from a mobile device.

**Account Details:** Username: *testcollector*, Password: *testpass*

### 6.3.7 View Collectors

This section of the homepage displays the existing collector accounts for the user and allows deleting them (as shown in figure 6.12).

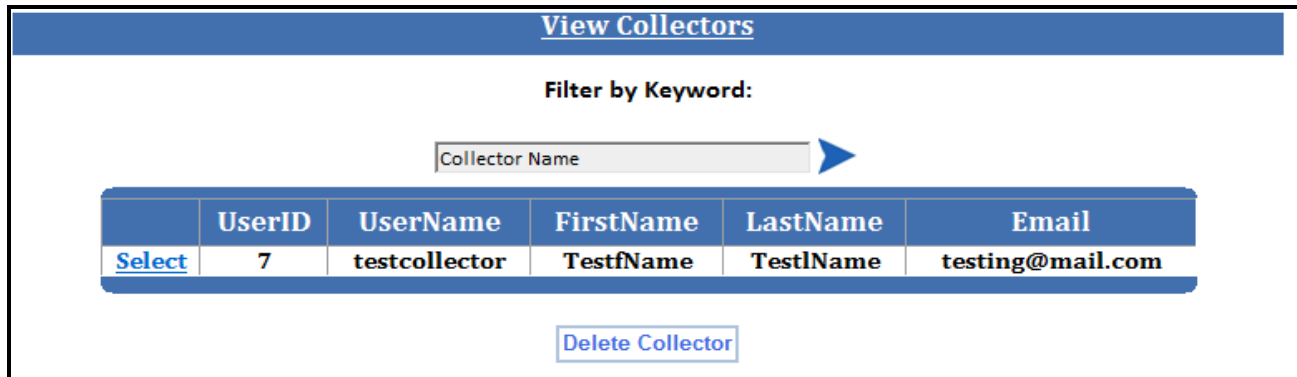


Fig.6.12 View Collector

### 6.3.8 Data Collection

This section shows the testing of the data collection done by login in to the collector account (created in section 6.6.2 of Testing) from mobile phone simulator and using the form (created in section 6.6.1 of Testing).

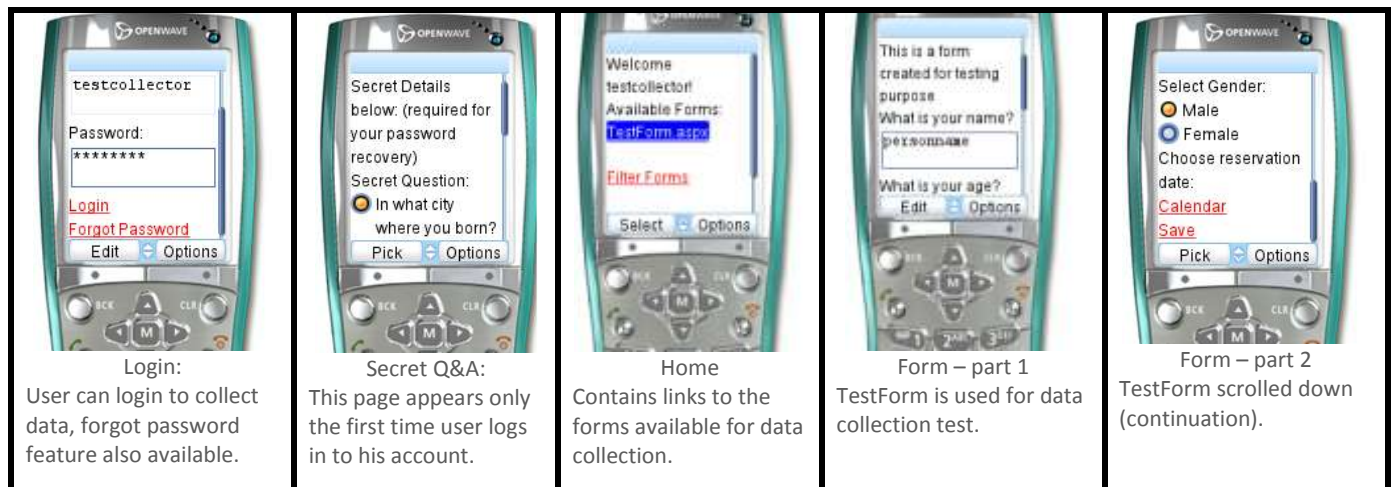


Fig. 6.13 – Mobile Web Page and Data collection



### 6.3.9 View Forms

In this section the test for “View Forms” segment of the homepage is carried out. It displays the existing forms to the user and allows deleting them or viewing the data collected for a particular form (as shown in figure 6.14).

**View Forms**

Filter by Keyword:

Form Name

ID	Form Name	Creator	Created On	Entries	Link	Last Update
Select 22	TestForm	6	14-06-2010	1	TestForm.aspx	15-06-2010

View Data    Delete Form

Delete Table from Database

ID	CollectorNo	FName	Age	Sex	RDate
1	7	personname	20	Male	18-06-2010 00:00:00

Click on view data after selecting row from above Forms Table

Appears when view data is clicked for a selected form.

Fig. 6.14 – View Forms and Form Data

### 6.3.10 Export Data

In this section the test for the export data function is done, the test and its result can be seen from below in figures 6.15 and 6.16 respectively.

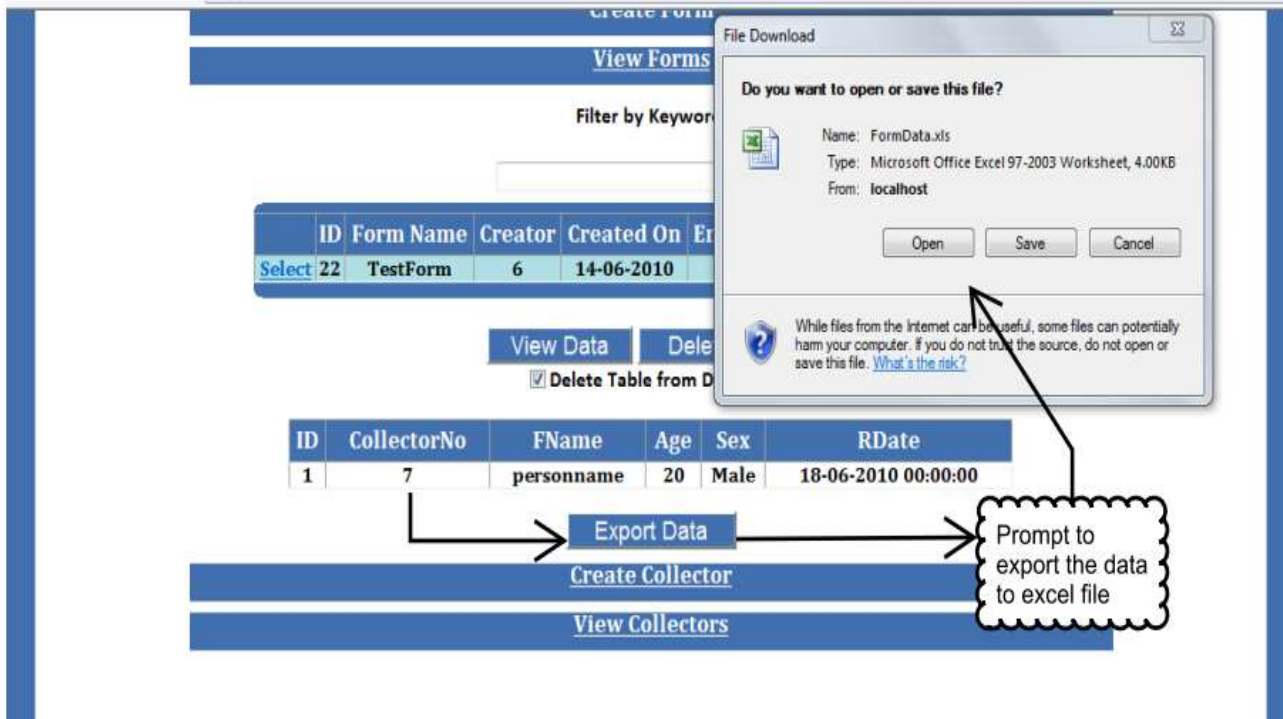


Fig. 6.15 – Export Data

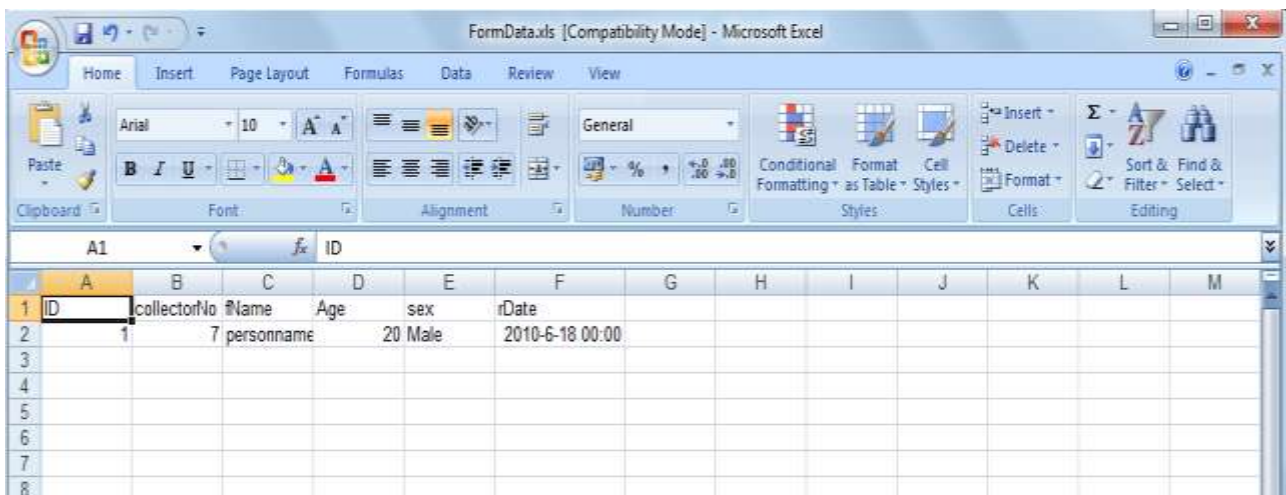


Fig 6.16 – Excel File

### 6.3.11 Account Page

In this section the test for “Account” page is done.

It has two sections; both the sections are functioning properly.

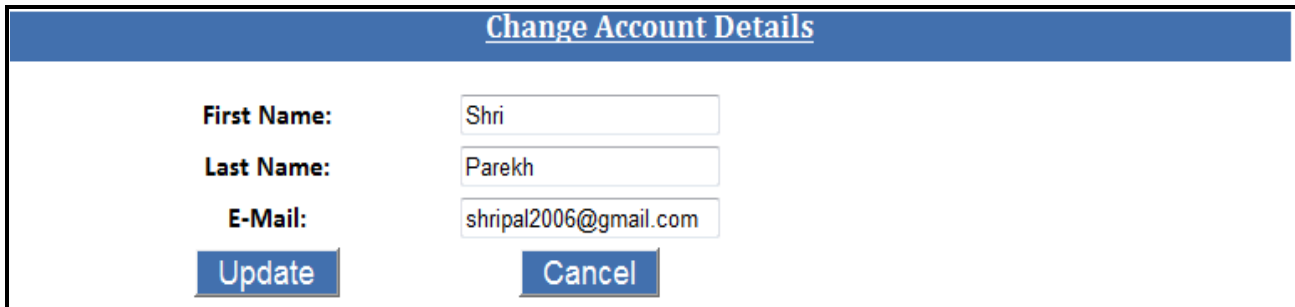


Fig. 6.17 – Change Account Details

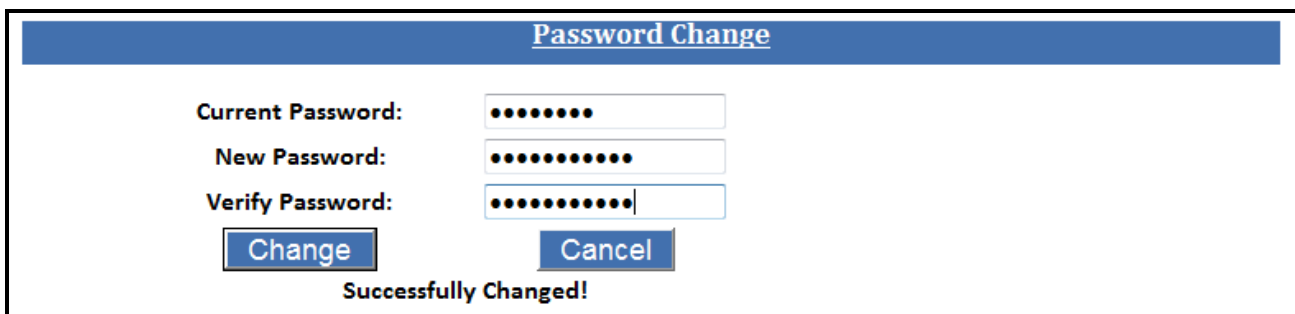


Fig. 6.18 – Change Password

Change Account Details (figure 6.15) - tested by updating the firstname to “Shri”

Change password (figure 6.16) – tested by changing the password to “newtestpass”

### 6.4 Summary

The testing phase showed the system is properly functioning as per the screenshots demonstrated, there were a couple of bugs found during the phase and were all corrected.

For further use of the system for testing the account created in the testing phase can be used with the credentials: Username: testuser and Password: newtestpass

## **CHAPTER 7**

### **CONCLUSION AND FUTURE WORK**

---

#### **7.1 Project Achievements**

The aim of the project has been to provide a solution that allows users to build a web application that allows user to create forms that are on the fly generated as mobile web pages and connected to the organization's database, to be accessed from mobile phones for data-collection purpose.

This has been achieved with a system that has been built and tested in the previous chapters of implementation and testing respectively.

The testing results prove that the project has been a success whilst there being the potential for the system to be developed further.

The future of the system remains unclear, it is possible the system will be deployed and made available through the internet or I shall continue working on the enhancement of the system or incorporate it with an already existing system as an enhancement.

#### **7.2 Conclusion**

The minimum requirements, objectives and deliverables of the project have been successfully accomplished. Moreover, a number of experts commend the work and believe that the project has a strong potential. No significant faults were found with the system, to enhance the system further there has been a few suggestions and will be well considered for the future work.

### **7.3 Recommendations for Future Work**

The system has a distinct scope for further enhancements, as described earlier in Section 4 of Chapter 1 in this report. The main areas I see as significant improvements are:

1. Allow user to select fields in existing database tables to insert data.

This feature would make the project a lot more practical in terms of usage, as it would allow users to relate between data being collected and would reduce the memory space consumed by new tables. It can be implemented for any organization that would use the system according to their database schema.

2. Allow uploading photos from phone memory or taken via camera.

The data currently possible to be collected is simple text based hence an addition to include images would increase the scope of the project.

3. Allow editing existing forms and backing up data already collected.

It is possible that the user would only want to change a small part of the form already existing; in such a case this new feature would save a lot of time and effort as opposed to designing a new form from the scratch.

## References:

- [1] Wireless Mobile Computing: Rugged PDAs. Retrieved February 27, 2010 from <[http://www.spiritdatacapture.co.uk/product\\_technologies\\_wirelessmobilecomputing\\_PDAs.asp](http://www.spiritdatacapture.co.uk/product_technologies_wirelessmobilecomputing_PDAs.asp)>
- [2] Creatability Concepts – FAQ. Retrieved February 27, 2010 from <<http://www.createabilityinc.com/FAQ.html>>
- [3] Ronan Cremin, et al.: *DotMobi Mobile Web Developer Guide*. mTLD, Dublin (2007)
- [4] Mathew MacDonald, Mario Szpuszta: *Pro ASP.NET 3.5*. Apress, New York (2008)
- [5] Subhasis Saha, et al. (June 2001) – “Bringing the Wireless Internet to Mobile Devices”. Retrieved March 01, 2010 from: <<http://crystal.uta.edu/~kumar/cse6306/papers/three.pdf>>
- [6] Ian Gilfillan (06.24.2002) – “Introduction to Relational Databases”. Retrieved March 01, 2010 from: <<http://www.databasejournal.com/sqlc/article.php/1469521/Introduction-to-Relational-Databases.htm>>
- [7] Alan Dix, et al.: *Human Computer Interaction*. Prentice Hall, Europe (1998)
- [8] PHYSORG.com, (04.28.2009) - “Open source mobile technology software reinventing health care in developing countries”. Retrieved March 14, 2010 from <<http://www.physorg.com/news160128864.html>>
- [9] Melisa Loudon (02.18.2009) - “Mobile Phones for Data Collection”. Retrieved March 26, 2010 from: <<http://mobileactive.org/howtos/mobile-phones-data-collection>>
- [10] Britta Stromeyer (01.22.2010) - “Text messages save lives – A Business profile on FrontlineSMS:Medic”. Retrieved March 26, 2010 from <[http://businessprofiles.suite101.com/article.cfm/text\\_messages\\_save\\_lives](http://businessprofiles.suite101.com/article.cfm/text_messages_save_lives)>
- [11] OOAD – Wikipedia. Retrieved March 26, 2010 from <[http://en.wikipedia.org/wiki/Object-oriented\\_analysis\\_and\\_design](http://en.wikipedia.org/wiki/Object-oriented_analysis_and_design)>
- [12] UML – Wikipedia. Retrieved March 26, 2010 from <[http://en.wikipedia.org/wiki/Unified\\_Modeling\\_Language](http://en.wikipedia.org/wiki/Unified_Modeling_Language)>
- [13] OpenWave Simulator. Retrieved April 06, 2010 from <<http://developer.openwave.com/download/index.html>>