HIV TESTING WEB APPLICATION

A Thesis
Presented to the
Faculty of
San Diego State University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Wenjun Zhu
Spring 2016
SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the

Thesis of Wenjun Zhu:

HIV Testing Web Application

Roger Whitney, Chair
Department of Computer Science

Carl Eckberg
Department of Computer Science

Janet Bowers
Department of Mathematics and Statistics

May 3, 2015
Approval Date
Copyright © 2016
by
Wenjun Zhu
All Rights Reserved
DEDICATION

I would like to dedicate this work to my girlfriend and my parents. This could not have been possible without them.
ABSTRACT OF THE THESIS

HIV Testing Web Application
by
Wenjun Zhu
Master of Science in Computer Science
San Diego State University, 2016

Currently, some HIV testing agencies still use paper forms to collect clients’ HIV test information. The data collected from paper forms has to be manually input into a database. Afterwards, data can be analyzed and reports can be generated. This thesis project is developed to convert the data collection method from paper forms into a web application. The web application allows users to use the application as long as the users have computer access and internet connection. As for some of the areas where internet is not always available, the application can be setup as a local/intranet application. In this application, all of the questions and options are highly customizable, which means the application has the potential to be easily adopted by different HIV testing agencies. The application also supports multi-languages, so it can be used at foreign locations of these HIV agencies.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>xii</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Benefits of the Web Application</td>
<td>1</td>
</tr>
<tr>
<td>1.3 Outline</td>
<td>2</td>
</tr>
<tr>
<td>2 FRAMEWORK, PROGRAMMING LANGUAGES AND DATABASE</td>
<td>3</td>
</tr>
<tr>
<td>2.1 ASP.NET Framework</td>
<td>3</td>
</tr>
<tr>
<td>2.2 Entity Framework</td>
<td>3</td>
</tr>
<tr>
<td>2.3 Bootstrap Framework</td>
<td>4</td>
</tr>
<tr>
<td>2.4 HTML and CSS</td>
<td>4</td>
</tr>
<tr>
<td>2.5 JavaScript</td>
<td>4</td>
</tr>
<tr>
<td>2.6 C#</td>
<td>4</td>
</tr>
<tr>
<td>2.7 SQL Server Database</td>
<td>5</td>
</tr>
<tr>
<td>3 REQUIREMENTS ANALYSIS</td>
<td>6</td>
</tr>
<tr>
<td>3.1 Functional Requirements</td>
<td>6</td>
</tr>
<tr>
<td>3.1.1 Business Need</td>
<td>6</td>
</tr>
<tr>
<td>3.1.2 New Clients’ Profile Creation</td>
<td>7</td>
</tr>
<tr>
<td>3.1.3 Test Result Input</td>
<td>7</td>
</tr>
<tr>
<td>3.1.4 Referral Information Input</td>
<td>7</td>
</tr>
<tr>
<td>3.1.5 Authentication</td>
<td>8</td>
</tr>
<tr>
<td>3.1.6 Positive Client Search</td>
<td>8</td>
</tr>
</tbody>
</table>
3.1.7 Site and Counselor Information Management ..........................................8
3.1.8 Questions Management .............................................................................8
3.1.9 User Registration ......................................................................................9
3.2 Non-Functional Requirements ...........................................................................9
  3.2.1 User Friendliness ......................................................................................9
  3.2.2 Response Time ........................................................................................10
  3.2.3 Extensibility ............................................................................................10
  3.2.4 Maintainability ........................................................................................10
  3.2.5 Safety ......................................................................................................10
  3.2.6 Backup ....................................................................................................11
  3.2.7 Browser Portability .................................................................................11
3.3 Work Flow .................................................................................................11
4 APPLICATION DESIGN ............................................................................................13
  4.1 Three-Tier Architecture Design ....................................................................13
    4.1.1 Presentation Tier .....................................................................................14
    4.1.2 Business Tier ...........................................................................................14
    4.1.3 Data Tier .................................................................................................14
  4.2 SQL Server Database Design .........................................................................14
    4.2.1 Application Type (OLTP OR OLAP) .............................................................14
    4.2.2 Database Diagrams Design ......................................................................15
    4.2.3 Database Table Design ...........................................................................16
5 APPLICATION IMPLEMENTATION ........................................................................19
  5.1 ASP.NET Web Forms ....................................................................................19
    5.1.1 Default Page ............................................................................................19
    5.1.2 Test Information Page .............................................................................24
    5.1.3 Search Page .............................................................................................26
    5.1.4 Referral Page ...........................................................................................27
    5.1.5 Administrator Page ...................................................................................28
    5.1.6 Master Page .............................................................................................33
    5.1.7 Auto Translation of Question ..................................................................33
  5.2 Error Handling .............................................................................................36
    5.2.1 Page-Level and Code-Level Error Handling .............................................36
LIST OF TABLES

Table 3.1. Desktop computer hardware specifications ............................................................10
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 3.1</td>
<td>Home page of the application</td>
<td>9</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>Workflow flowchart</td>
<td>12</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>Three Tier architecture design</td>
<td>13</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>Database diagram</td>
<td>15</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>Clientinfo table</td>
<td>16</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>Questions table</td>
<td>17</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>Sites table</td>
<td>17</td>
</tr>
<tr>
<td>Figure 4.6</td>
<td>Counselor table</td>
<td>18</td>
</tr>
<tr>
<td>Figure 4.7</td>
<td>InitialINFO table</td>
<td>18</td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>Default page</td>
<td>20</td>
</tr>
<tr>
<td>Figure 5.2</td>
<td>Test date</td>
<td>21</td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>C# Code: initialize culture</td>
<td>22</td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>Load questions and options with different languages</td>
<td>22</td>
</tr>
<tr>
<td>Figure 5.5</td>
<td>Generate different types of controls</td>
<td>23</td>
</tr>
<tr>
<td>Figure 5.6</td>
<td>Error message</td>
<td>23</td>
</tr>
<tr>
<td>Figure 5.7</td>
<td>Disabled pregnant and prenatal questions for male clients</td>
<td>24</td>
</tr>
<tr>
<td>Figure 5.8</td>
<td>Test information page</td>
<td>24</td>
</tr>
<tr>
<td>Figure 5.9</td>
<td>Button style</td>
<td>25</td>
</tr>
<tr>
<td>Figure 5.10</td>
<td>Completed test information page</td>
<td>25</td>
</tr>
<tr>
<td>Figure 5.11</td>
<td>Search page</td>
<td>26</td>
</tr>
<tr>
<td>Figure 5.12</td>
<td>C# code: search HIV positive clients</td>
<td>27</td>
</tr>
<tr>
<td>Figure 5.13</td>
<td>Referral page</td>
<td>27</td>
</tr>
<tr>
<td>Figure 5.14</td>
<td>Log in page</td>
<td>28</td>
</tr>
<tr>
<td>Figure 5.15</td>
<td>C# Code: authenticate and check ID</td>
<td>28</td>
</tr>
<tr>
<td>Figure 5.16</td>
<td>Administrator page</td>
<td>29</td>
</tr>
</tbody>
</table>
Figure 5.17. Site management page........................................................................................................29
Figure 5.18. Add a new site page.............................................................................................................30
Figure 5.19. HTML code: update records in GridView control..............................................................30
Figure 5.20. HTML code: insert a record in DetailView........................................................................31
Figure 5.21. Question management page...............................................................................................31
Figure 5.22. Counselor management page..............................................................................................32
Figure 5.23. Add a new counselor page..................................................................................................32
Figure 5.24. Register a new user page.....................................................................................................33
Figure 5.25. Define header and footer in the master page.................................................................33
Figure 5.26. Question dropdown list........................................................................................................34
Figure 5.27. Download translation from Google Translate website...................................................34
Figure 5.28. Translated question and its options in 4 other languages................................................35
Figure 5.29. Display successful message...............................................................................................35
Figure 5.30. Code: update question table...............................................................................................36
Figure 5.31. C# code: handle page-level error.......................................................................................37
Figure 5.32. C# Code: handle SQL error..................................................................................................37
Figure 5.33. Code in Global.asax file: handle application-level error.................................................38
Figure 5.34. Code in web.config file: handle application-level error....................................................38
ACKNOWLEDGEMENTS

I would like to express my thanks to my thesis advisor Professor Roger Whitney for his expert advice, support and unfailing patience. I would also like to thank Professor Carl Eckberg and Professor Janet Bowers for their time and effort as well as support for my thesis.
CHAPTER 1

INTRODUCTION

This chapter gives an introduction to HIV test information, the reason to build this web application and the benefits of this web application.

1.1 BACKGROUND

HIV testing agencies are required to collect certain HIV testing information from clients. HIV test information includes pretest information, test result information and referral information. Pretest information needs to be filled out by a client before a test. Generally, pretest information includes demographic information, HIV risk-assessment information, previous HIV test result and HIV test survey. Test result information includes test technology, test time and current test result. Test result information will be filled out by HIV test counselors once HIV tests come out. Referral information includes the facility a HIV-positive client is referred to, referral date, medication start date, medication, treatment date and treatment information. Referral information is required only if both results of two HIV tests are positive.

Some HIV testing agencies collect data from paper forms first and manually input the collected data into a database system. Then, these agencies will analyze data and generate reports through the database system. Sometimes, these agencies struggle with the extra time and resources spent on this process. This web application is developed to help these agencies effectively collect and store data into databases.

1.2 BENEFITS OF THE WEB APPLICATION

There are five reasons for replacing paper forms with a web application. First, web application is easy to access. There is no need for installation of client-side application on each individual computer at HIV testing locations. The web application can be easily access
through a web browser by different types of devices, such as PC, smart phones and tablets. Second, this web application uses SQL server database. Therefore, it can generates reports and it supports electronic data transmission. Data can be analyzed using T-SQL. Third, it is more efficient to collection data using this web application. It eliminates the step of manual data input. Forth, data collected will be more accurate. The application will validate input of a user. For example, a user cannot proceed unless all required fields are filled. In addition, there won’t be any data transposition error from data entry. At last, web application is a cross-platform application. The application supports all the major operating systems on the market, such as windows, OSX, IOS and Android.

1.3 OUTLINE

The paragraph below is an outline of this thesis, which helps to develop a better understanding of the thesis structure.

- Chapter 2 introduces the frameworks, programing languages and database used to deploy this web application.
- Chapter 3 discusses functional requirements, non-functional requirements and workflow of the web application
- Chapter 4 provides the details of the web application’s architectural and database design.
- Chapter 5 gives detailed page-by-page implementation and application error handling.
- Chapter 6 describes future enhancements.
- Chapter 7 provides a conclusion of the thesis.
CHAPTER 2

FRAMEWORK, PROGRAMMING LANGUAGES
AND DATABASE

This web application uses Microsoft ASP.NET and the Bootstrap framework. Programming languages used are: HTML, CSS, JavaScript, and C#. HTML, CSS, and JavaScript are used for client-side scripting. C# is used for server-side scripting. The web application also uses SQL server for database development.

2.1 ASP.NET FRAMEWORK

ASP.NET is a back-end framework, which is developed by Microsoft. It is an open-source and free server-side web application framework [1]. This project uses .NET framework 4.5 to develop the web application. ASP.NET includes integration with SQL server database and web services. ASP.NET offers three different frameworks for creating web applications: Web Forms, ASP.NET Web Page and MVC. This project uses Web Forms.

2.2 ENTITY FRAMEWORK

Entity framework enables the web application to access relational databases with minimal code. It is an open source object-relational mapping (ORM). It enables programmers to work with relational data using domain-specific objects [2]. Before the introduction of the entity framework, developers have to first create a separate entity for each data element. Then, developers have to create a relationship based on business logic and create a connections of the entities in order for the database to read and store data. However, with the entity framework feature, developers only need to create data access classes using C#. Then the application will be able to communicate with database and manipulate data elements through data classes.
2.3 Bootstrap Framework

Bootstrap is a front-end framework of HTML, CSS and JS. It used by this project as a web application creation tool. It is open source and is free to use. It includes templates for Gridview, Textbox, Button, Table, Date/Time picker and other controls. The latest versions of major browsers such as, Internet Explorer, Chrome and Firefox support Bootstrap. Bootstrap supports responsive web design. It is able to adjust layout of a web application to mobile device such as tablets and smart phones with different screen resolutions [3].

2.4 HTML and CSS

HTML (HyperText Markup Language) is a standard markup language which can be used to build web pages. Tags enclosed in angle brackets are the key elements. HTML tags usually comes in pairs like <title> and </title>. They are known as opening tags and closing tags [4]. HTML5 is the fifth revision of the HTML standard of World Wide Web Consortium (W3C). It is used for structuring and presenting content for World Wide Web. It has added many new features the project takes advantage of. HTML5 allows applications to get a gauge to adjust parameter. It adds email inputs. It allows applications to validate email address input without extra code. It also adds required attribute so that required input boxes can be set up without any extra JavaScript code or client-side code [5].

2.5 JavaScript

JavaScript is a dynamic programming language. It is a client-side scripting language. It can interact with users, validate controls, change the action of the controls, alter documents and communicate asynchronously. All major browsers support JavaScript. JavaScript has dynamic typing. It is an object based language [6]. JQuery, a cross-platform library of JavaScript, is used to simplify HTML. It is open source and free. This project uses JQuery to 1) find, display and hide a document, 2) choose and manipulate Document Object Model (DOM) elements, 3) create events, and 4) validate controls [7].

2.6 C#

C# is a server-side programming language. It is strong typing and object-oriented. It is integrated with .NET library and has automatic garbage collection. The definition of its classes or functions can be defined in different orders. Developers can define a class inside
another class, but cannot define global variables or functions. With support of Microsoft, different libraries of C# for different purposes can be easily found on line [8].

2.7 SQL Server Database

SQL Server is a relational database management system from Microsoft. It runs on Transact-SQL (T-SQL), which is a Microsoft’s and Sybase’s proprietary extension to SQL. T-SQL supports exception and error handling, transaction control and declared variables [9]. This application uses SQL Server 2012 express to build its database. SQL Server Management Studio is used to make changes to the database and to backup database.
CHAPTER 3

REQUIREMENTS ANALYSIS

System requirement is a critical part of application development. This chapter introduces both functional requirements and non-functional requirements of this web application. This chapter also describes workflow of this application in the end.

3.1 FUNCTIONAL REQUIREMENTS

Functional requirement describes a system’s function, which is what a system can do [10].

3.1.1 Business Need

Currently, clients of some HIV testing agencies need to fill out a paper pretest information form before a test. The paper form includes a list of questions asking about demographic information, HIV risk-assessment information, previous HIV test result and a HIV test survey. If these agencies need to analyze data or generate reports, they need to hire data entry personnel to manually input information from hundreds of paper forms into a database. There are several drawbacks these agencies are trying to avoid: (1) Data errors from data entry process, (2) Data entry cost and inefficiencies, and (3) incomplete forms or forms with illegible handwriting.

This HIV test web application is designed to improve this data collection process. Instead of writing on a paper form, clients will use either computers or tablets to answer all the questions. Then information entered will be stored into a database at the same time. Data entry by the agent is no longer required. Data errors and data entry costs can be reduced. In the meanwhile, with data validation function, all required information needs to be filled out before proceeding to the next step. In addition, the application improves clients’ on-site experience. It supports five different languages: English, Spanish, Chinese, Korean and
Japanese. This function reduces language barriers of clients and can be used in foreign locations of the HIV testing agencies.

### 3.1.2 New Clients’ Profile Creation

Once a client arrives at a test location, the client needs to create his/her own profile. The web page will display current location and counselor information at the top. Counselor will make sure the correct location and counselor information is selected. Clients can choose any of the five languages listed on the top. Then, clients can click the start button to start answering questions. There are five questions on each screen. Once all the questions are completed, clients can click save button to create their profiles.

### 3.1.3 Test Result Input

After a client clicks the save button, test result page will show. The client will then take a HIV rapid test in a private room on site. The test looks for antibodies to HIV in blood or oral fluid samples. If a client is exposed to viruses like HIV, antibodies produced by their immune system will make the HIV rapid test positive. Usually, it takes less than 30 minutes to know the test result [11]. If the test result is negative, then the client does not need to take a second test. Otherwise, the client need to take a second test. Once the test result comes out, an on-site counselor will input test result for the client. On the test result page, a counselor can input test result for 2 tests. For each test, the following information needs to be entered: begin time, end time, test result and test technology (test technology describes which technology is used for this test.), current result decided by 2 tests.

Once the test result is entered into the web application, the application will generate a client ID. Counselor will give the client ID to the client. Since the application is designed for anonymous HIV tests due to confidentiality purpose, client ID, instead of client name, will be used to identify a client during any follow-up visits.

### 3.1.4 Referral Information Input

If the test result is negative, no referral information is needed. If the test is positive, a counselor is required to input referral information. Referral information page includes the facility a HIV-positive client is referred to, referral date, and other follow up information,
such as medication start date, medication, treatment date, and treatment. This will be the last step of the work flow for one client.

### 3.1.5 Authentication

To access the referral and administration pages, the application requires a counselor’s username and password. Once the username and password are verified, the counselor can access to the referral page and administration page. Counselors can also change their own passwords after login.

### 3.1.6 Positive Client Search

Since HIV positive clients need to be referred to one of the HIV facilities for treatment, counselors need to be able to find HIV-positive positive clients. HIV-positive clients can be easily found by selecting test site and counselor in a dropdown list. Since HIV test results will come out within thirty minutes [11], all HIV positive clients will be referred to various facilities on the same day of the HIV test. Only current date’s HIV-positive clients will show through this search function.

### 3.1.7 Site and Counselor Information Management

Since this application is used at multiple locations, it is important to choose the correct site and counselor. After the site and counselor information is selected on a device once, the information will be saved in the database associated with current device’s profile. Therefore, the device will remember and display previously selected site location and counselor information once the selection is made.

Counselor has the right to edit site name, edit site address and creation date, and change default language and country. Counselors can also add a new site to the system as needed.

Counselors also have the right to edit counselor’s first name, last name, and created date. Counselors can also add a new counselor to the system if needed.

### 3.1.8 Questions Management

Counselors are able to edit demographic questions, HIV risk-assessment questions, and test survey questions and edit options of these question. Counselors can also edit the
facilities referred to on the referral page. Other information on the referral page and test results questions are fixed and cannot be modified or changed by any counselors. It is designed this way because rarely counselors will need to update these questions. However, these questions can be edited by programmers if needed.

### 3.1.9 User Registration

Counselors have the right to create a new counselor account. A new email and password are required for each registration. Email will be used as login username. Password can be changed after login.

### 3.2 Non-Functional Requirements

A non-functional requirement defines how the application works or behaves. It is the quality attribute of an application [12].

#### 3.2.1 User Friendliness

The theme color of the application is mainly blue and white. Large size blue buttons are used. Users can easily read texts and click each button. (Figure 3.1) Since there will be more and more users use tablets to get access to the application, date picker, which is tablet user-friendly, is incorporated during the application design stage. Multiple-selection buttons, larger text fonts and larger-size buttons become part of the design for the convenience of users.

![Figure 3.1. Home page of the application.](image)
3.2.2 Response Time

Application performance is an essential part of user experience. Home page takes about one second to load. The application takes less than one second to switch between home page, Site page, Referral page and Admin page. When a client switches from one language to another, it will take about one second. At last, searching positive clients by counselors will take about one second to show result. Response time test was performed five times on a desktop computer. The hardware specifications of the desktop computer is shown in Table 3.1.

Table 3.1. Desktop computer hardware specifications

<table>
<thead>
<tr>
<th>Processor</th>
<th>Intel Core i7 - 4790</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard Drive</td>
<td>128 GB SSD</td>
</tr>
<tr>
<td>Memory</td>
<td>8 GB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Windows 10 Home 64-bit English</td>
</tr>
<tr>
<td>Browser</td>
<td>Chrome Version 49.0.2623.87 m</td>
</tr>
<tr>
<td>Internet Connection Bandwidth</td>
<td>50 Mbps</td>
</tr>
</tbody>
</table>

3.2.3 Extensibility

The application uses a standard template as a master page. When a new page with new functions needs to be added to the application, the new page can be easily added using this master page. Since the master page contains the navigation bar and footer, new pages created will have a uniform look.

3.2.4 Maintainability

The application code uses descriptive variable, method and class names to make future maintenance easier.

3.2.5 Safety

The application uses HTTPS protocol for secure communication on the internet. In addition, counselor account passwords must meet complexity requirements. The setup requires at least one capital letter, one letter, one number and one special character. The length of each password must be more than 7 characters. The password is stored in the SQL server database and is encrypted using Hash algorithms. The SQL server database of the
application is encrypted through BitLocker Drive Encryption. BitLocker works at the volume level. It encrypts data using the AES algorithm.

### 3.2.6 Backup

Depending on the traffic condition of each HIV testing location, the SQL server database of the application will either have daily full backups or transaction log backups.

### 3.2.7 Browser Portability

The application uses JQuery, Boostrap, HTML5 and CSS3. The application was developed under Google Chrome. Google Chrome is the recommended browser.

### 3.3 WORK FLOW

The flowchart below depicts the work flow of this application. (Figure 3.2)
Figure 3.2. Workflow flowchart.
CHAPTER 4

APPLICATION DESIGN

4.1 THREE-TIER ARCHITECTURE DESIGN

This web application uses three-tier architecture design. (Figure 4.1) The three tiers are presentation tier, business tier and data tier [13]. There are two advantages of choosing three-tier architecture comparing to a 2-tiered architecture (Client tier and Data tier) for this web application. First, there is better security control at the database tier. Client side will not have direct access to the database since they only need to access to the application. Second, developers can make changes on presentation tier without affecting business tier and data tier.

![Diagram of Three Tier Architecture Design]

Figure 4.1. Three Tier architecture design.
4.1.1 Presentation Tier
Presentation tier is the top level of the application. It is an interface that users use to interact with the application. Presentation tier includes web forms, user controls and shared UI code. In this application, default page, tester page, referral page and administration page are all web forms. Language user control in this project provides the feature to switch between 5 different languages on default page. Master page in this project contains shared UI code.

4.1.2 Business Tier
Business tier is the middle level of the application. It includes business rules and data rules. It also exchanges and processes the data between presentation tier and data tier. It validates the input, the validation represents business rules.

4.1.3 Data Tier
Data tier is the bottom level of the application. This tier is the SQL server that stores the application’s data. It includes database tables and stored procedures.

4.2 SQL Server Database Design
The database design is based on the understanding of the requirements of the application. This database design includes analyze application type, design database diagrams, and create database table.

4.2.1 Application Type (OLTP OR OLAP)
This application’s type is On Line Transaction Processing (OLTP), because users are interested in creating, reading and updating records of clients instead of data analysis. The application needs to insert new records to store each client’s demographic and risk information constantly. There are 36 questions to collect demographic and risk information for each client. The application will need to insert 36 rows into one table to store the information [14].
4.2.2 Database Diagrams Design

This application uses SQL server database diagram designer to design and visualize database. Figure 4.2 is the database diagram of this application. A relationship line between two tables represents a one-to-one relationship if there is a key symbol at the endpoint of a relationship line. A relationship line represents a one-to-many relationship if there is an infinity symbol at the endpoint of a relationship line [15].

Figure 4.2. Database diagram.
4.2.3 Database Table Design

Clientinfo table (Figure 4.3) stores clients’ HIV test results and referral information. It is a larger table. Table fields contain details of test results and referral information. ID is the primary key. ID has to be an integer. It is also the identity field of the table and its identity increment has been set to 1. Clientinfo table contains a few foreign keys. CounselorID is the foreign key linked to the ID of Counselor table. SiteID is the foreign key linked to the ID of Sites table.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Allow Nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>CounselorID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>SiteID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>HIVstatus</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test1tech</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test1Begin</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test1End</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test1Result</td>
<td>nvarchar(50)</td>
<td></td>
</tr>
<tr>
<td>Test2tech</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test2Begin</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test2End</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Test2Result</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>ReferredTo</td>
<td>nvarchar(10)</td>
<td></td>
</tr>
<tr>
<td>DateofReferral</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Medication</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>MedicationDate</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Treatment</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>TreatmentDate</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>TestingDate</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Createdate</td>
<td>datetime</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>nvarchar(50)</td>
<td></td>
</tr>
<tr>
<td>Touchscreencomplete...</td>
<td>datetime</td>
<td></td>
</tr>
<tr>
<td>Testinginfocompletetime</td>
<td>datetime</td>
<td></td>
</tr>
<tr>
<td>Referralcompletetime</td>
<td>datetime</td>
<td></td>
</tr>
<tr>
<td>LoginID</td>
<td>nvarchar(128)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.3. Clientinfo table.

Questions table (Figure 4.4) contains demographic questions, HIV risk assessment questions and the options of these questions. ID is the primary key and has to be an integer.
The table has fields to hold five different languages, including English, Spanish, Chinese, Korean and Japanese. Nvarchar datatype is used for all the question fields and option fields. Nvarchar takes twice space that Varchar takes. However, it allows the database to store unicode characters and multilingual data. Nvarchar uses 2 bytes per character and can hold a maximum of 4,000 characters [16]. Status field uses bit data type, since it is either active or inactive.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Allow Nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>bit</td>
<td></td>
</tr>
<tr>
<td>EnglishQuestion</td>
<td>varchar(200)</td>
<td></td>
</tr>
<tr>
<td>EnglishOptions</td>
<td>varchar(2000)</td>
<td></td>
</tr>
<tr>
<td>SpanishQuestion</td>
<td>nvarchar(200)</td>
<td></td>
</tr>
<tr>
<td>SpanishOptions</td>
<td>nvarchar(2000)</td>
<td></td>
</tr>
<tr>
<td>ChineseQuestion</td>
<td>nvarchar(200)</td>
<td></td>
</tr>
<tr>
<td>ChineseOptions</td>
<td>nvarchar(2000)</td>
<td></td>
</tr>
<tr>
<td>KoreanQuestion</td>
<td>nvarchar(200)</td>
<td></td>
</tr>
<tr>
<td>KoreanOptions</td>
<td>nvarchar(2000)</td>
<td></td>
</tr>
<tr>
<td>JapaneseQuestions</td>
<td>nvarchar(200)</td>
<td></td>
</tr>
<tr>
<td>JapaneseOptions</td>
<td>nvarchar(2000)</td>
<td></td>
</tr>
<tr>
<td>[Order]</td>
<td>decimal(18, 1)</td>
<td></td>
</tr>
<tr>
<td>QuestionType</td>
<td>varchar(200)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.4. Questions table.

Sites table (Figure 4.5) contains basic information of each test. ID is the primary key and ID has to be an integer. Nvarchar is the datatype for SiteName, SiteAddress, language and Country fields. These fields may contains unicode characters and multilingual data. Datatype “bit” is used to hold sites’ status, such as active or inactive.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Allow Nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>SiteName</td>
<td>nvarchar(500)</td>
<td></td>
</tr>
<tr>
<td>SiteAddress</td>
<td>nvarchar(300)</td>
<td></td>
</tr>
<tr>
<td>CreateDate</td>
<td>date</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td>nvarchar(90)</td>
<td></td>
</tr>
<tr>
<td>Country</td>
<td>nvarchar(150)</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>bit</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.5. Sites table.
Counselor table (Figure 4.6) contains each counselor’s basic information. ID is the primary key and it has to be an integer. Nvarchar is used as the datatype for FirstName, LastName and Fullname. These fields might contain unicode characters and multilingual data. Datatype “bit” is used to hold counselors’ status, such as active or inactive.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Allow Nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>FirstName</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>LastName</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>Fullname</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
<tr>
<td>created</td>
<td>date</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>bit</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 4.6. Counselor table.**

InitialINFO table (Figure 4.7) contains users’ choices of sites and counselors. The primary key is ID.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Allow Nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>SiteID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>CounselorID</td>
<td>int</td>
<td></td>
</tr>
<tr>
<td>TestDate</td>
<td>varchar(50)</td>
<td></td>
</tr>
<tr>
<td>Computername</td>
<td>nvarchar(200)</td>
<td></td>
</tr>
<tr>
<td>Stename</td>
<td>nvarchar(200)</td>
<td></td>
</tr>
<tr>
<td>Counselorname</td>
<td>nvarchar(100)</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 4.7. InitialINFO table.**
CHAPTER 5

APPLICATION IMPLEMENTATION

5.1 ASP.NET Web Forms

ASP Web Forms is a programming model that programmers can use to create ASP.NET web applications. In this application, web forms include client script, HTML, server controls, and C# code. Upon an user’s request, web forms will be compiled and executed by .NET framework on the server. .NET framework will then generate HTML markup that web browsers render [17].

Web forms are generated through the following steps in this application. First, create an empty web form associated with the master page in Visual Studio. Second, add web controls such as Label, Textbox, Button and Table. Third, set up property, method, and event for each control. Size, color and layout of each control can be adjusted by either property or HTML/CSS code. JavaScript and C# are used to handle events of web controls. Page and control events used are Init, Load, PreRender, Upload, DataBinding, Disposed and Error.

Web forms can separate HTML, CSS and JavaScript code from application server-side code C#. Web forms use view state and server-based forms to store value. Therefore, data already entered by a user can be retained by web forms. In addition, stored value can be accessed by code, which enables users to switch between different pages without losing any information entered previously.

5.1.1 Default Page

When a user gets access to this application through URL in a browser, the user will first see the default page (Figure 5.1). Default page is open to every client. Thus, default page is not protected by username and password.
On the top of the default page, there is a navigation bar. The navigation bar contains link buttons to other main pages. The navigation bar is from the master page. Therefore, the bar will be displayed on all other pages. Each page uses the same code for the navigation bar. If there are any changes made on the navigation bar in the master page, updated navigation bar will be automatically applied to all other pages.

Right below the navigation bar, there are language buttons: English button, Spanish button, Chinese button, Korean button and Japanese button. A user can use these buttons to select a language for his/her questionnaire. Once a user click a language button, the default page will be refreshed and be displayed in the selected language.

In the middle of the page, there are site information, counselor information and test date. A counselor will make sure correct site and counselor are selected. If any of the information is wrong, a counselor can correct the information by clicking the “Click here to change site” link. Correction can also be made by clicking the “Site” button at navigation bar. Clients can change test date by clicking the date box and picking a date from the date picker (Figure 5.2) [18].
Before loading the default page, the application will override and pull culture information from cookies (Figure 5.3). Culture contains information of both the language and the region. For example the culture code “en-US” indicates that the language is English and the region is United States. Based on the culture from cookies, the web application will determine which language to use for the default page. If the web application fails to find any culture information from cookies, it will set the default culture to “en-US” (English, U.S.) and save culture “en-US” to http cookies. Once the web application decides which language to use, it will load questions and their options from database in this language. (Figure 5.4).
protected override void OverrideCulture()
{
    string language = String.Empty;
    HttpCookie cookie = Request.Cookies["CurrentLanguage"];
    if (cookie != null && cookie.Value != null)
    {
        language = cookie.Value;
        CultureInfo culture = CultureInfo.CreateSpecificCulture(language);
    }
    else
    {
        language = "en-US";
        CultureInfo cul = CultureInfo.CreateSpecificCulture(language);

        HttpCookie cookie_new = new HttpCookie("CurrentLanguage");
        cookie_new.Value = language;

        // Set cookie expiration period to 6 months
        cookie_new.Expires = DateTime.Now.AddMonths(6);
        Response.SetCookie(cookie_new);
    }

    base.OverrideCulture();
}

Figure 5.3. C# Code: initialize culture.

int counter = 0;
foreach (Question q in questions)
{
    tr = new TableRow();
    tc = new TableCell();
    tc.Width = Unit.Percentage(30);

    switch (language)
    {
        case "English":
            question = q.Text;
            option = q.Options;
            break;

        case "Spanish":
            question = q.SpanishQuestion;
            option = q.SpanishOptions;
            break;

        case "Korean":
            question = q.KoreanQuestion;
            option = q.KoreanOptions;
            break;

        case "Chinese":
            question = q.ChineseQuestion;
            option = q.ChineseOptions;
            break;

        case "Japanese":
            question = q.JapaneseQuestions;
            option = q.JapaneseOptions;
            break;

        default:
            question = q.Text;
            option = q.Options;
            break;
    }
}

Figure 5.4. Load questions and options with different languages.
In this application, there are different types of controls (Figure 5.5) on different types of questions. Examples of controls are date picker box, single text box, multi-line text box, checkbox, checkbox list, multi-select, radio button, and radio button list.

```java
if (q.QuestionType.ToLower() == "multiselect")
{
    ms1 = new ListBox();
    ms1.ID = "ms1_" + q.ID + "multiselect";
    ms1.Attributes.Add("multiple", "multiple");
    ms1.Attributes.Add("form-control input-lg multiselect selectpicker");
    ms1.Width = Unit.Percentage(65);
    if (!string.IsNullOrEmpty(option))
    {
        string[] options = option.Split(',');
        foreach (string o in options)
            ms1.Items.Add(o.Trim());
    }
    tc.Controls.Add(ms1);
}
if (q.QuestionType.ToLower() == "checkbox")
{
    if (!string.IsNullOrEmpty(option))
    {
        string[] options = option.Split(',');
        int length = options.Length;
        for (int i = 0; i < options.Length; i++)
        {
            cb[i] = new CheckBox();
            cb[i].ID = "cb_" + q.ID + i;
            cb[i].ValueChanged = Unit.Percentage(80);
            cb[i].Text = options[i].ToLower();
            cb[i].Width = Unit.Percentage(100);
            cb[i].LabelAttributes.Add("data-toggle", "toggle");
            cb[i].LabelAttributes.Add("class", "checkbox-inline");
            tc.Controls.Add(cb[i]);
        }
    }
}
```

Figure 5.5. Generate different types of controls.

Answer validation is a very important part of the default page. For example, age has to be a positive integer. If a client inputs a letter or a negative integer in the textbox, an error message will show up (Figure 5.6). Additionally, if a client chooses “Male” as his gender, questions related to pregnant and prenatal care will be disabled and options will be greyed out (Figure 5.7). Once clients finish answering all of the questions, “Save” button will pop up. Once the save button is clicked, all the answers will be saved to the database. The application will go to test information page.

![What is your age?](Please enter a valid number.)

Figure 5.6. Error message.
5.1.2 Test Information Page

Test information page (Figure 5.8) allows test counselors to input two HIV test results for each client. For each test result, test counselors need to fill out test technology, test begin time and end time, and test result.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test 1</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Begin Time:</strong></td>
<td>20:27</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>End Time:</strong></td>
<td>20:27</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Result:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Negative</strong></td>
<td></td>
<td><strong>Positive</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Test 2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Begin Time:</strong></td>
<td>20:27</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>End Time:</strong></td>
<td>20:27</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Result:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Negative</strong></td>
<td></td>
<td><strong>Positive</strong></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.8. Test information page.

The button styles are from Bootstrap. The “TestingTechnology1” button’s style is “btn-primary active” class. “TestingTechnology2” button’s style is “btn-primary” class.
(Figure 5.9). Since both “TestingTechnology1” button and “TestingTechnology2” button are in the same radio button list, selecting one button will automatically make the other button deselected. “TestingTechnology1” button is the pre-selected button in the radio list.

![Figure 5.9. Button style.](image)

After a test counselor enters both test results and clicks the “Complete” button, the application will generate a ClientID (Figure 5.10) for this client and save test information to the database.

![Figure 5.10. Completed test information page.](image)

If a client’s first HIV test result is negative, the test counselor will input the first test information in this page. This client’s HIV testing process is complete. If the client’s first HIV test result is positive, the client will need to take a second HIV test. If the second test
result is negative, the test counselor will enter both test results and suggest the client to go to a healthcare center to take an additional HIV test to confirm test result. If both of the HIV test results are positive, the test counselor will ask the client to write down his or her Client ID.

**5.1.3 Search Page**

Counselors need to fill out referral information for HIV positive clients. The search page enables counselors to search these clients (Figure 5.11). Search page is password protected. Only counselors with authorized login information can log into search page.

![Search page](image)

**Figure 5.11. Search page.**

After logging in, a test counselor needs to choose a site and a counselor. A list of today’s HIV positive clients will be listed on the search page. The counselor can select a client from the list to go to his or her referral page. Search page gets data from database through entity framework and binds the data set to the GridView control on the page. (Figure 5.12)
5.1.4 Referral Page

Test counselors use referral page to input referral information for HIV-positive clients. It includes which facility the client has been referred to and date of referral. It also includes medication start date, medication, treatment date and treatment information. (Figure 5.13) This page is designed to be used by counselors, therefore, it requires log in information. If a user trying to get access to this page’s URL without a login, the application will automatically redirects the user to log in page (Figure 5.14).

![Referral Information](image)

**Figure 5.13. Referral page.**

```csharp
protected void New_Tick(object sender, EventArgs e)
{
    string today = DateTime.Now.ToString("yyyy-MM-dd");
    var list = (List<ClientInfo>)Session["clientlist"];  
    foreach (var item in list)
    {
        if (item.DateOfReferral == today)  
        {
            item.DateOfReferral = null;
            item.DateOfReferral = null;
            item.DateOfReferral = null;
        }
    }
}
```
The code in Figure 5.15 checks whether a user has logged in when the user is trying to access to referral page. Also this page expects an ID passed from the test information page. It uses the ID to pull the client’s information that is input into the web application earlier. If there is no value found by this page, the application will automatically redirect the user to default page to avoid exception.

```csharp
protected void Page_Load(object sender, EventArgs e)
{
    if (!this.Page.User.Identity.IsAuthenticated)
    {
        //FormsAuthentication.RedirectToLoginPage();
        Response.Redirect("~/Account/login");
    }
    else
    {
        if (Request.QueryString.Keys.Count <= 0)
        {
            Response.Redirect("~/Default");
        }
        else
        {
        }
    }
}
```

Figure 5.15. C# Code: authenticate and check ID.

5.1.5 Administrator Page

The administrator page is used to manage configuration and register new users.

(Figure 5.16)
Figure 5.16. Administrator page.

Figure 5.17 is site management page. Users can use it to update site information or add a new site. (Figure 5.18)
Since only test counselors can update site information or add a new site, both pages require login to access. Site management page uses Gridview control to list all the sites. Users can update site information through the Gridview control. Code in Figure 5.19 defines the template (which columns to display), SQL data source and SQL command for updating the site information in GridView.

Figure 5.18. Add a new site page.

Figure 5.19. HTML code: update records in GridView control.
Add a new site page uses DetailView control. It allows users to add a new site. The code in Figure 5.20 defines the template, SQL data source and SQL command for adding a new site in the DetailView.

Figure 5.20. HTML code: insert a record in DetailView.

Figure 5.21 is question management page. Users can change the text of the questions and their options in five languages. The code behind this page is similar to site management page.

<table>
<thead>
<tr>
<th>QuestionType</th>
<th>EnglishQuestion</th>
<th>EnglishOptions</th>
<th>SpanishQuestion</th>
<th>SpanishOptions</th>
<th>ChineseQuestion</th>
<th>ChineseOptions</th>
<th>JapaneseQuestions</th>
<th>JapaneseQuestions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text radiobutton</td>
<td>What is your Current Gender?</td>
<td>Female, Male, Transgender (Male), Transgender (Female)</td>
<td>¿Qué es su género actual?</td>
<td>Hombre, Mujer, Transgénero (Hombre), Transgénero (Mujer)</td>
<td>女性、男性、他者身分（男性）、他者身分（女性）</td>
<td>性別は何ですか</td>
<td>あなたはどのような性別ですか</td>
<td>あなたはどのような性別ですか</td>
</tr>
<tr>
<td>Text radiobutton</td>
<td>What is your SFQ?</td>
<td>1, 2, 3, 4, 5</td>
<td>¿Qué es su SFQ?</td>
<td>1, 2, 3, 4, 5</td>
<td>1, 2, 3, 4, 5</td>
<td>1, 2, 3, 4, 5</td>
<td>1, 2, 3, 4, 5</td>
<td>1, 2, 3, 4, 5</td>
</tr>
<tr>
<td>Text radiobutton</td>
<td>Are you Pregnant?</td>
<td>Yes, No, Don't know, Prefer not to say</td>
<td>¿Estás embarazada?</td>
<td>Sí, No, No lo sé, Preferir no decir</td>
<td>産者は同様の質問</td>
<td>あなたは妊娠していますか</td>
<td>あなたは妊娠していますか</td>
<td>あなたは妊娠していますか</td>
</tr>
<tr>
<td>Text radiobutton</td>
<td>Are you in Previous Care?</td>
<td>Yes, No, I don't know, Prefer not to say</td>
<td>¿Estás en atención preexisitente?</td>
<td>Sí, No, No lo sé, Preferir no decir</td>
<td>前期ケアを受けていますか</td>
<td>あなたは前回ケアを受けていましたか</td>
<td>あなたは前回ケアを受けていましたか</td>
<td>あなたは前回ケアを受けていましたか</td>
</tr>
<tr>
<td>Text radiobutton</td>
<td>Have you had a blood transfusion?</td>
<td>Yes, No</td>
<td>¿Habías una transfusión de sangre?</td>
<td>Sí, No</td>
<td>献血を受けましたか</td>
<td>あなたは献血を受けましたか</td>
<td>あなたは献血を受けましたか</td>
<td>あなたは献血を受けましたか</td>
</tr>
<tr>
<td>Checkboxes</td>
<td>If so, when?</td>
<td>Si es así, ¿cuándo?</td>
<td>Sí, cuándo?</td>
<td>日付を入力してください</td>
<td>あなたは何時</td>
<td>あなたは何時</td>
<td>あなたは何時</td>
<td>あなたは何時</td>
</tr>
</tbody>
</table>

Figure 5.21. Question management page.
Counselor management page can be used to update counselor information. If a test counselor needs to be removed, the test counselor can be set as inactive through unchecking the active box (Figure 5.22).

![Counselor management page](image)

**Figure 5.22. Counselor management page.**

Users can also add a new counselor through add a new counselor page (Figure 5.23).

![Add a new counselor page](image)

**Figure 5.23. Add a new counselor page.**

In the register page (Figure 5.24), users can create a new user account by inputting an email and password. Email, password and confirm password fields are all required. After a new account is created, the new user will be able to access to test information page, referral page and administrator pages.
5.1.6 Master Page

The master page provides a template for all the other pages in this web application. It contains the navigation bar and footer for all other pages. Therefore, navigation bar and footer of all pages will have a uniform look. In addition, if there are any changes made to the navigation bar or footer on the master page, changes will be applied to all other pages. Code in Figure 5.25 defines navigation bar and footer for all pages.

Figure 5.24. Register a new user page.

5.1.7 Auto Translation of Question

Question management page has auto translation function. The function allows test counselors to obtain translation of questions and their options. A test counselor can select a question needs to be translated in the dropdown list (Figure 5.26). Then, the web application will send the text of the question and its options in English with target language culture code to www.google.com/translate website. Translation will be downloaded from the website (Figure 5.27). Then, the application will display the translations in Spanish, Chinese, Japanese, and Korean (Figure 5.28).
Figure 5.26. Question dropdown list.

```java
//Net Translated Response from Google Translate Website
//Method Autotranslate text parameter and culturecode parameter format en/zh if translate from English to Chinese
public static string Autotranslate(string text, string cultureCode)
{
    //Use webclient to download web page
    //Set encoding to UTF8
    WebClient googlewebclient = new WebClient();
    googlewebclient.Encoding = System.Text.Encoding.UTF8;
    //Define URL with the correct format
    string googleURL = string.Format("http://www.google.com/translate?hl=zh-CN&tl=zh-CN&sl=en&text={0}&ie=UTF8", text, cultureCode);
    string translatedText = googlewebclient.DownloadText(googleURL);

    //Download source code from google translate website using UTF8
    //Source code contains the translated text: <span title="what is your name">омомомомомо</span>
    //remove anything before what is your name

    //Get the translated text between <span title> and </span>
    int first = translatedText.IndexOf("<span title="") + 13;
    int last = translatedText.IndexOf("</span>", first);
    string subString = translatedText.Substring(first, last - first).Trim();
    string translatedText = subString.Substring(subString.IndexOf(">") + 1);    

    return translatedText.Trim().ToString();
}
```

Figure 5.27. Download translation from Google Translate website.
Figure 5.28. Translated question and its options in 4 other languages.

And the user can click “Apply translations” button, translated questions and options will be saved under Question table. The web application will display a successful message (Figure 5.29) after completion.

Figure 5.29. Display successful message.
Code in Figure 5.30 updates questions and options in the Questions table.

![Code Snippet]

**Figure 5.30. Code: update question table.**

### 5.2 Error Handling

Error handling may have a direct effect on user experience. If a web application handles errors in a user friendly way, users won’t feel panic when an error occurs. When an error occurs, this web application will display a detailed error message and ask the user to send the message to the maintenance team. In this way, users will know what to do when an error occurs and the maintenance team will know what issue needs to be fixed. In ASP.NET web application, there are 2 type of errors. They are page/code-level errors and application-level errors [19].

#### 5.2.1 Page-Level and Code-Level Error Handling

This web application uses page-level handler to handle page-level errors. When a page-level error occurs, the page-level handler will collect the error message and display the error messages on error page (Figure 5.31). No page content will be displayed at the same time, because no controls will be generated by the web application when an error occurs. The page-level handler is able to catch errors which try-catch code cannot catch.
This application uses try-catch code to handle code-level errors. A page is created to display error messages. This page is called error page. Whenever the web application reads data, inserts a new record or updates a current record in its database, there is a higher possibility that an error will occur. Therefore, the try-catch code is added to these communication processes to capture errors and pass error messages to the error page. When a code-level error occurs, the application will redirect current page to the error page and display detailed error messages. Since the web application uses entity framework to communicate with its database, it uses “DbEntityValidationException” class to capture related errors (Figure 5.32).

```csharp
try
{
    context.SaveChanges();
}
catch (DbEntityValidationException ex)
{
    var errorMessages = ex.EntityValidationErrors
        .SelectMany(x => x.ValidationErrors)
        .Select(x => x.ErrorMessage);
    var fullErrorMessage = string.Join("; ", errorMessages);
    var exceptionMessage = string.Concat(ex.Message, " The validation errors are: ", fullErrorMessage);
    Response.Redirect("error.aspx?errormessages" + exceptionMessage);
}
```

**Figure 5.32. C# Code: handle SQL error.**

### 5.2.2 Application-Level Error Handling

In order to handle application-level error, the error handler that is defined in Global.asax file is incorporated into this application (Figure 5.33).
Default errors and HTTP errors need to be handled in web.config file. The “statuscode” allows the application to redirect the current page to the error page (Figure 5.34). When “statuscode” 404 happens, the application will go to the error page with corresponding error message on it.
CHAPTER 6

APPLICATION ENHANCEMENT

The following is a list of enhancements to improve the application in the future.

- Allow counselors to create several customized series of questions. Allow each client to choose which series of questions to answer.
- Add content manager user, who can only edit question contents.
- Include multi-languages selection function to all web pages to increase user experience. Currently, only home page supports multi-language selection option.
- Allow the application automatically adapt to different devices, such as PCs, tablets, and smart phones. To achieve this goal, the application should be able to automatically adjust application resolution, change click button size, adjust layout of navigation bar and footer.
CHAPTER 7

CONCLUSION

This project creates a web application for HIV testing agencies. These agencies will be able to use the application to collect and store HIV test information electronically, analyze collected data, and generate reports from the database. These agencies will be able to collect and save a large amount of data without any manual input. The application will help to collect clients’ demographic information, HIV risk assessment information, test information and referral information. It validates client’s information input and supports five different languages. Auto translation function allows counselors to obtain translations of questions and options from google translate website. Administration function enables counselors to edit and add questions, sites information, and counselor information. It also allows counselors to register new users. In conclusion, the web application achieves the purpose of collect data electronically and is able to perform all the functions as designed.
REFERENCES


