International Journal of Computer System (ISSN: 2394-1065), Volume 02– Issue 06, June, 2015
Available at http://www.ijcsonline.com/

Design and Development of a Web Application Framework for Alumni Profiling

Nelia Ereno, Kurt Junshean Espinosa and Ryan Ciriaco Dulaca

Sciences Cluster, University of the Philippines Cebu, Cebu City, Philippines

Abstract

Profiling systems are primarily used in decision-making. In universities, they can be used to assess the curriculum of the program to make it relevant in meeting the market demand. However, they are implemented in different ways. In this paper, we present an integrated web application framework for alumni profiling that may be used by any university. With WAF, many aspects in web applications that are crucial to data integrity and the system itself are already inherently provided. A prototype was created and details of the design and some statistical analyses of the data gathered are presented. Initial results showed that the integrated web application framework developed can be used effectively in profiling systems. It also showed that the prototype may be used readily by any university and may be customized based on their needs. We recommend improving the prototype by integrating an automated statistical system for faster decision-making.

Keywords: *Alumni Tracker, Management Information Systems, Profiling Systems.*

I. INTRODUCTION

Profiling is the ability to assess a comprehensive amount of information about a person [4]. It is widely used in business (consumer profiling), criminology (criminal profiling), clinical psychology (profiling of patients), among others. It is also used in universities (faculty profiling or student profiling or alumni profiling). In fact, most universities maintain the profiles of their faculty and alumni either locally (hard copies or local systems) or even publicly, through their websites. For local systems and/or websites, they use password-protected database profiling. Such type of database profiling provides the users with a more accurate, complete and up-to-date informative profile. It also protects some information which need not be disclosed to the public.

Keeping track of professional profiles is an effective way to assess and evaluate curricular programs of a university. Professional capacity of the alumni can be determined and thus, the relevance of the curriculum of the degree program they graduated can be validated. It can also be a source of information to identify geographic diversity of graduates, that is, what foreign countries do the graduates got employed. The success of the alumni serves as the best market value of a degree program and attracts those who are looking for a university where they intend to pursue graduate studies.

On the other hand, profiling paves the way for the alumni to stay connected with the university. It also serves as an avenue for them to give back to their alma mater. Moreover, profiling can enhance each other's personal and professional networks which are vital for placement opportunities that can be explored by the university.

There are a number of ways to implement a good profiling system. One of which is to implement using a web application framework (WAF). WAF provides inherent support for security, scalability, database access, and other features. Without the framework, all these features must be handled by the developer from scratch, which can take up time and might not be aligned to the best practices in software engineering. Universities should no longer be burdened by the design and development of the profiling system. Providing it, the universities can focus on the more important part of profiling, which is analysis.

In this paper, we introduce the use of web application framework for creating the profiling system. We discuss the requirements of the system, the database design, the web application design, and the implementation. We then present the results and discussion on the said system.

II. RELATED LITERATURE

A. Profiling System

Technology is best used to collect information. Tracking and profiling graduates are common in several universities abroad. Sydney University of Technology [6] has done an exploratory study on the tracking and profiling of their IT graduates. They gathered data using instruments such as the Course Experience Questionnaire (CEQ) and an internal Student Satisfaction Survey. They discovered that their graduates were not just technically competent but also displayed emotional intelligence in their workplace.

Nunes, Silva, Santos, Queiroz and Leles [9] surveyed 633 registered Brazilian dentist who graduated from the Federal University of Goias. Using the two-step cluster method based on job-related variables they were able to identify

professional profile subgroups based on job-related variable.

B. Web Application Frameworks

The evolution of web application frameworks started from single-tier model where dumb terminals are directly connected to the mainframe. It was a centralized model. The presentation, business logic, and data access are intertwined in one monolithic mainframe application. Since clients are dumb terminals and therefore do not have any processing logic whatsoever, there is no client management required. Furthermore, because the mainframe application has complete control on the data access logic, consistency is easily achieved. However, with single-tier model, there is intertwining of the different aspects (i.e. presentation, logic, data access) of the web application such that changes to one aspect may imply changes in another aspect which is a big maintainability issue.

When the personal computer era started, the two-tier model emerged. In this model, clients were named "fat" as they do most of the processing. It is the client that sends queries to back-end database in the form of database access protocol like SQL. While this is a great advantage over the first tier since it creates database independence, on the other hand, this consumes much bandwidth. Furthermore, clients maintain the presentation, business logic, and data model such that changes in any aspect of the model have to be replicated in all the clients which is a great bottleneck issue on maintainability.

With the bottleneck in the client-side came the advent of the three-tier model, wherein a new tier - the middle-tier, was introduced. In this model, the presentation is separated in the client, while the business logic and data model is transferred to the middle tier. The advantage of this is that many clients connect to one middle tier and reuse the same business logic and data model. To make this possible, the middle tier employs a common interface which clients can interact. However, this also creates bottleneck wherein the middle-tier has to handle concurrent requests from many clients. Hence, it needs some mechanisms of concurrency control, threading, transaction, security, and so on. In short, while this model introduces separation in the components, it also creates a bottleneck in the middle-tier level.

When the internet era came, the browser became the pervasive platform of user interface at the client side. The browser provides a common presentation logic by means of HTML and it communicates to the server via HTTP, another standard. And the business logic and data models are captured by dynamic content generation technologies such as ASP, PHP, JSP which interact with the backend database. However, the bottleneck in the middle-tier remains. The solution to this is the creation of web application frameworks which handle all the tasks and complexity in the middle-tier [7]. Since then, many web application frameworks were developed with different features and capabilities [14].

Web application frameworks are usually categorized according to the programming language in which they are built [13]. Usually, WAFs are compared as to what and how they support the following features: Ajax, MVC frameworks, MVC push-pull, internationalization, Object-Relational Management (ORM), testing frameworks, DB migration frameworks, security frameworks, template frameworks, caching frameworks, form validation frameworks [13]. The choice of the specific WAF depends on the need of the application.

III. PROPOSED SYSTEM

The proposed system was developed using rapid prototyping. This is the process of rapidly mocking up the future state of a system by simulating a few aspects of the system and having the client or users or even developers validate the prototype. Because of the multiple iterations of prototyping, reviewing, and refining, there is rapid feedback early on in the project development. In the process, the design of the system is improved and major changes in the system during the development process are reduced.

A. Requirements Gathering

1) Information needed to be gathered: The information collected from the alumni respondents were personal information, educational background and employment history.

a) Features needed: The system should support two types of users: the alumni, and the administrator. The alumni should be able to add and edit his/her personal, educational and employment information only. The administrator should also be able to add an alumni and edit the alumni's data. The system should also allow the administrator to do data cleaning.

b) Rules in the system: The system provides a tracker form where the alumni can enter the needed information. This means that the alumni can add themselves in the system's database. In instances where the alumnus has no access to the tracker form and has provided the administrator the needed information, the administrator can use the same tracker form to add the alumnus. The administrator can also edit the profile of the alumni for purpose of data cleaning.

2) Database Design

The system has three (3) main entities, namely, the alumni, company and the program (see Fig. 1). The alumni entity has student number, last name, first name, middle, present address, permanent address, email address, contact number in present address, and contact number in permanent address as attributes. The alumni's student number is its primary key. The alumni's names and the present address are required attributes. Since this is an alumni tracker and universities would want to know where their alumni are working, an entity for the company is created. Its attributes include company name (this is the primary key), address, contact information and the type of industry where the company belongs. It will be used to counter check whether the alumni's employment is in line with the program that he/she graduated from. Finally, the last entity is the program. The program entity represents all the programs offered by the University since its founding. Its attributes are program ID (this is its primary key), degree, major and year instituted.

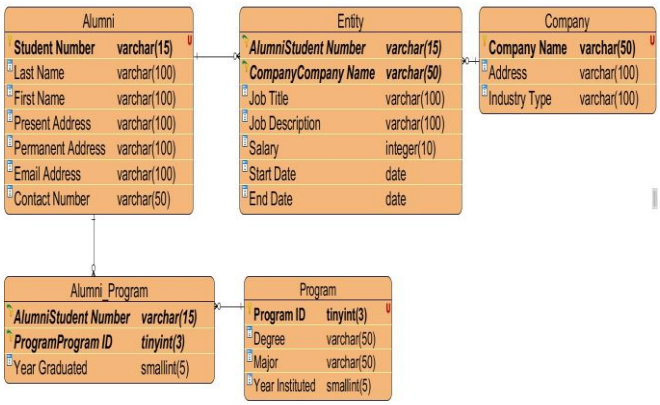

Figure 1. Entity-Relationship Diagram (ERD)

The entities are related to each other. The alumni graduated from a program. Also, it is interesting to note that the alumni may have graduated from more than one program. The cardinality of the relationship, then, between the alumni and the program is many-to-many. In the same manner, the cardinality of the relationship between company and alumni is many-to-many. Since the aforementioned relationships are many-to-many, tables (entities) must be created for them. For the alumni-program relationship, its attributes include the primary keys of alumni and program and the additional attribute year graduated. For the company-alumni relationship, its attributes include the primary keys of company and alumni, and the following additional attributes - job title, job description, salary, start date and end date.

3) Web Application Design: The web application has two users, the alumni and the administrator. The following use cases (see Fig. 2) represent the different functionalities of the system.

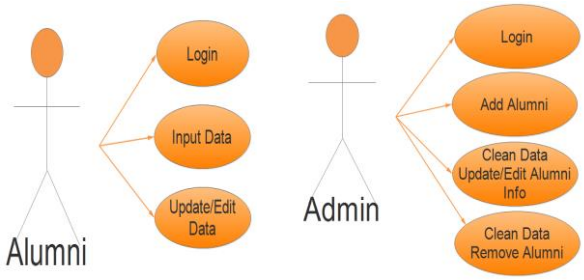

Figure 2. Use-case diagram

4) Implementation: In the implementation, there are lots of web frameworks to choose from. Basically, we have to choose a web framework that will meet the requirements and then implement the database design.
The first step is to choose a web framework that will meet the requirements. In this application, we decided to choose Ruby on Rails as the web framework. On their website [11], it says that "Ruby on Rails is an open-source web framework that is optimized for programmer happiness and sustainable productivity. It lets you write beautiful code by favoring convention over configuration (CoC)." CoC allows the programmer to spend more time in business logic rather than in setup and configuration. Rails (shortcut for Ruby on Rails) also supports the standard Model-View-Controller (MVC) which allows separation of the data model, the presentation logic and the business logic.

Another feature of Rails is support for "Don't repeat yourself (DRY)". DRY principle was formulated by Andrew Hunt and David Thomas in their book Pragmatic Programmer [1]. With DRY principle, a modification in one part of the system automatically cascades changes in another logically related part of the system. In Rails this is done internally using code generators, automatic build systems, and scripting languages [5].

Rails also implements another software engineering pattern which is active record pattern. According to Fowler [8], an active record is a data source architectural pattern. He defined it as an object that wraps a row in a database table or view, encapsulates database access, and adds domain logic on that data. With this behavior, any update on the object is automatically reflected in the database. This automatically encapsulates the extra work of generating queries to run the respective queries to the database.

Rails also provide easy integration with UI. In this system, we use the Twitter Bootstrap for UI [3]. All these features make Rails an ideal agile development framework where changes along the way can easily be handled.

The second step is to implement the database design. In this system, we use MySQL [10]. Rails supports active record as the Object-Relational Mapping (ORM) that provides interface between MySQL tables and Ruby application objects or entities. To create the active record files in Rails, you will issue a command at the top-level of the application directory following conventions such singular form for entities. The format of the command is the following [12]:

```
bin/rails generate model
Usage:
  rails     generate     model     NAME
[field[:type][:index]
field[:type][:index]] [options]

...

Active Record options:
   [--migration]    # Indicates when to
generate migration
                    # Default: true

...

Description:
    Create   rails   files   for   model
generator.
```

Then, you can create the associations and validations. The associations are indicated by adding declarations to models such as has_one, has_many, belongs_to, and has_and_belongs_to_many. CoC principle is also used here when for example, you indicate the relationship that an alumni has many companies, it can be written like this:

```
class Alumni < ActiveRecord::Base
```

```
        has_many :Companies
    end
```

Note that the Companies is in plural form.
The validations are also implemented in the Rails model [4]. For example, the code below indicates that email is a required field.

```
class Alumni < ActiveRecord::Base
        validates :email, presence:
true
    end
```

There are lots validation options such as length, numericality, presence, absence, uniqueness, conditional validations and custom validations. All these options can be found in their guide [2].

IV. Results and Discussion

A. Web-based Profiling System

The screenshots in Fig. 3-5 show the prototype system for the alumni profiling, which we call Alumni Tracker. It contains the features described above in the requirements. Fig. 3 shows the homepage which allows the users to login. Users can be admin or alumni. Fig. 4 shows a listing of all the alumni. In that page, the admin can create or add, view, update or edit, and delete alumni. Fig. 5 shows a sample page where the alumnus information is edited.

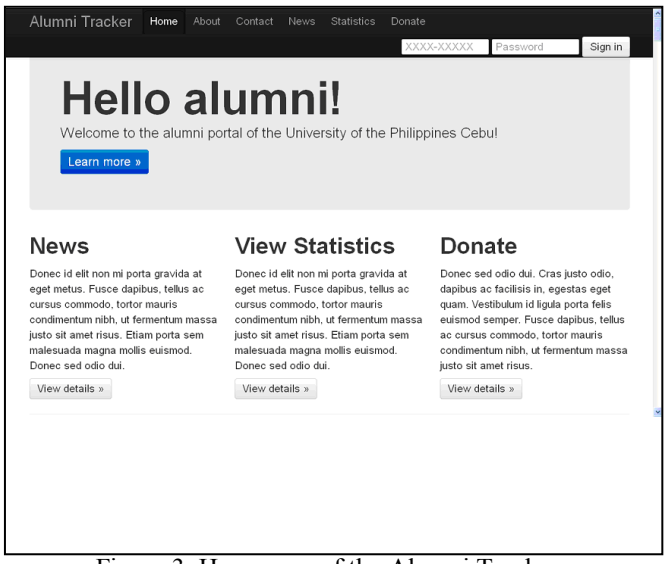

Figure 3. Homepage of the Alumni Tracker

Figure 4. Alumni Page with the CRUD (i.e. create, read, update, delete) Features

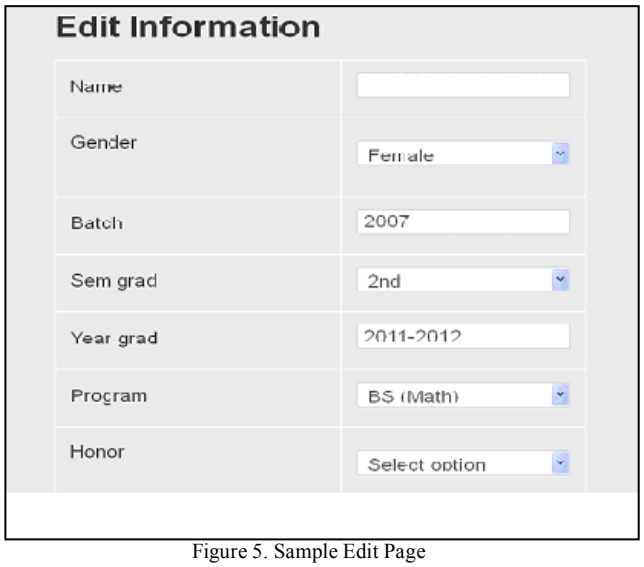

Figure 5. Sample Edit Page

B. Statistical Analysis

The constructed web-based system for alumni profiling was validated using alumni profiles of Math Program of the University of the Philippines Cebu. Since definitions of "alumni" vary from one university to another, we are defining our alumni as graduates of the Math Program of UP Cebu.

About 74.19% among the accomplished tracker form was analyzed and interesting results were noted.

Based on the Alumni Tracker, most of the Math Program alumni (38%) landed in IT industry for their first employment.. Others went to academe (15%), banking (9%) and business process outsourcing (8%). There were about 30% who pursued graduate studies, self-employment or full time family life.

Some of the alumni have resigned from their first job. Apparently, the intention was to look for greener pasture and/or more self-satisfying job. Though most of the alumni are still in the IT industry. Notably, many were promoted to supervisory/managerial position and a few have put up their own company.

About 17% of UP Cebu's Math Program alumni preferred to stay abroad and a good majority of 73% stayed in our country. In abroad, our graduates were employed as Accounting Manager, Controller Designee, High School Math teachers, Senior Software Engineer, IT Consultant, Owner/Manager of an IT Consulting Company, Assistant Vice President and Software Developer.

Based on the data gathered, of the eleven (11) alumni were resident of the United States of America; nine (9) were in Japan; nine (9) were in Singapore; three (3) were in Canada, one (1) was in Australia; one (1) in Sultanate of Oman; and one (1) was in Germany.

V. CONCLUSION AND RECOMMENDATIONS

The use of the web application has sped up the collection of data for alumni profiling. It has allowed for the collection of more data from the alumni. With the electronic data, its cleaning was made easier. The prototype also showed that such a profiling system can be developed for the general use by any university.

The statistical analysis was done separately. We are recommending that statistical tools should be made available in the web application itself. This will solve problem on data migration, from the tracker's database to a format readable statistical tool that was used in the analysis (IBM SPSS version 22).

A number of information have been extracted from the collected data. But the information that would be most beneficial to the decision makers of the University, especially for those responsible for curriculum development, is that most of the alumni are employed in the IT industry. This has not significantly changed even as the alumni transferred from one job to another, that is, the new employment was still in the IT industry.

REFERENCES

[1] A. Hunt, and D. Thomas., "The Pragmatic Programmer: From Journeyman to Master. Reading", MA: Addison-Wesley, 2000.

[2] "Active Record Validations". Active Record Validation. Accessed September 08, 2014. active_record_validations.html

[3] "Bootstrap." Bootstrap. Accessed September 08, 2014. http://getbootstrap.com/2.3.2/.

[4] D. Koren, "The Art of Profiling: Reading People Right the First Time", Expanded and Revised Edition, Richardson, Texas, International Focus Press, 2012.

[5] D. Thomas, (interviewed by Bill Venners) "All Programming Is Maintenance Programming." Orthogonality and the DRY Principle. March 10, 2003. Accessed September 08, 2014. http://www.artima.com/intv/dry.html.

[6] G. Scott, and D. Wilson. Tracking and Profiling Successful IT Graduates: An Exploratory Study. Sydney Australia, 2002.

[7] I. Vuksanovic and B. Sudarevic, "Use of web application frameworks in the development of small applications," *MIPRO, 2011 Proceedings of the 34th International Convention* , vol., no., pp.458,462, 23-27 May 2011

[8] M. Fowler, "Patterns of Enterprise Application Architecture", Boston: Addison-Wesley, 2003. Print.

[9] M. Nunes, E. Silva,, L. Santos, M. Queiroz, and C. Leles, "Profiling Alumni of a Brazilian Public Dental School", Human Resources for Health , 2010.

[10] "MySQL :: The World's Most Popular Open Source Database." MySQL :: The World's Most Popular Open Source Database. Accessed September 08, 2014. http://www.mysql.com/.

[11] "Ruby on Rails." Ruby on Rails. Accessed September 08, 2014. http://rubyonrails.org/.

[12] "Ruby on Rails Guides (v 4.1.6)". Ruby on Rails Guide. Accessed September 08, 2014. guides.rubyonrails.org

[13] T. Shan and W. Hua, "Taxonomy of Java Web Application Frameworks," *e-Business Engineering, 2006. ICEBE '06. IEEE International Conference on* , vol., no., pp.378,385, Oct. 2006. doi: 10.1109/ICEBE.2006.98

[14] V. Okanovic, "Designing a web application framework," *Systems, Signals and Image Processing (IWSSIP), 2011 18th International Conference on* , vol., no., pp.1,4, 16-18 June 2011