That Group

Irving Arredondo, Daniel Childs, Herald Chris Villarreal, Nestor D. Almanza, Luis Santos

Major Responsibilities

Daniel Childs

XML Instance, Django Template, Export, Import , XML Schema, GAE Models, Merge Import,

Herald Chris Villarreal

HTML Template, Django Template, Export, Import, XML Schema, GAE Models, Merge Import

Nestor Almanza

Validation, Django Template, Export, Import, XML Schema, GAE Models, Merge Import

Irving Arredondo

Technical Report, Django Template, Export, Import, XML Schema, GAE Models, Merge Import

Luis Santos

UML Models, Django Template, Export, Import, XML Schema, GAE Models, Merge Import

Executive Summary

IMDB, the popular internet movie database, uses an effective database model to store and present information that exhibit a many-to-many relationship to their users. In this paper, we show our process of designing and implementing our own database model concerning world crisis, using IMDB as a model and a source of inspiration. Our website hopes to be able to present its contents in a coherent and aesthetically pleasing manner that can be enjoyed by our target users.

Table of Contents

Introduction

The problem

We (the members in *That Group*) have been given the task of creating a website that emulates the website imdb.com. However, instead of movies, shows, actors, etc., we were to choose four crises, four people, and four organizations that are interrelated. Each crisis has at least one person and

organization that are involved in the crises, but one crisis can have more than one person or organization that is involved. As a result, we have to design a database that supports a "many-to-many relationship", in which a parent row might have many children. For example, say we are trying to relate books and authors. A book, "X", might have many authors and an author, "Y", could have certainly authored many books, yet "X" and "Y" could still be related. This production of this website has used many tools a few of them being a UML designer, GAE unit tests, GAE Task Queue, GAE Django Templates, and GAE Datastore (just to name a few). We realize that even at the end of the three phases, the website will not be complete, and it might not ever be. To produce a website of high quality and efficiency takes a lot longer than a couple of weeks; however, we hope to have produced a quality website that is both aesthetically appealing and can easily be navigated.

Importance

There are many reasons that we see importance in this project that we have been assigned. First, we are creating a website (specifically a website that is served by the Google App Engine) that is a database of many crises that are occurring in the world. As stated earlier, each crisis has at least one person and organization involved, and each crisis has a way that it can be helped, whether it be signing a petition or donating money to the cause. Each person has a social networking website (Twitter, Facebook, etc.) that can be used to contact that person, or in the least voice one's opinion on the matter that they are involved with. In addition, each organization has contact information that can be also be used to voice one's opinion or to find out ways to help or join the cause that it is fighting for. With all this information in a convenient website, we are raising awareness to crises that might have not gotten the attention that they deserve!

Another reason of importance is that we are learning valuable skills that will help us create similar databases, allowing us to organize information in a coherent manner. There is an endless amount of information out there in the world, and being able to organize a specific instance of that information in a manner that is presentable to the general public is a vital skill that can prove to be favorable as technology continues to advance.

Difficulties

Even though this project has been broken in to three phases, the first two phases have already proven to bring many challenges. In the first phase, none of us in *That Group* had much experience with any of the tools that were vital and necessary to complete the project and all of its requirements. All of our time in the beginning was concentrated on trying to learn the software that was being used and how to get the software to do what we wanted! This has been the most frustrating part of the project thus far. In the second phase, we are now having trouble understanding exactly how to implement a database. We are now more comfortable with the software required for this project, however, being comfortable with software and knowing how to effectively use the software to implement your designs is a big jump. Hopefully, we will be able to finish all of the requirements given and deliver a product (in this case a website) of high quality.

Another difficulty that arose was our choice of using Python 2.7 as opposed to Python 2.5 in programming our website. If a person uses Python 2.7 to program their website, then our many changes that one has to do in order for their website to function accordingly. We did not know this at the start, and had a very difficult time at the beginning knowing when and what to change for our website to be functional. These changes also transcended into using the GAE unit tests to test our website, as we

could not get them fully functional until well into phase 1. In retrospect, we should have researched the differences in using either language or how it would affect what we were trying to accomplish.

Key Components of Approach and Results

The first thing we did in trying to solve this problems was navigate IMDB and the websites of groups in the past that created the same website. We noted what we liked about each website, but more importantly we noted were we could improve in our design, to make it easier for the user to navigate or understand the material that is being presented. From there we began to experiment and tried making more simple websites that were served by the appspot. We wanted to first understand how the appspot worked, before we tried to venture out into creating the appspot for our project. From that point, there has been a continuing exchange of ideas amongst us in the overall direction of our website. Every member of *That Group* has had influence in the overall design of the website. There have been clashes, but we have democratically settled the disputes. Every change that someone wants to make in the website has to be run through the whole group, and the group must vote on its approval. Also, we make sure to divide tasks accordingly, so no one person has to carry the load for other. Now that the websites of other groups have been posted, we are using those as a guide to help us improve our main direction and our design.

Limitations

There are not many limitations put in place by the specifications in this project. Professor Downing has given us a wide variety of tools to use and has left it to our discretion on how to use them. However, a limitation in this who process has been and continues to be our inexperience with the software and tools that have been provided for us. We have many ideas on how to implement our

solution, but we have met dead ends, as we have not been able to figure out how to convert many of those ideas into code. We have been working tirelessly to overcome the learning curve and finally be able to see our ideas come into fruition. Another limitation has been the amount of time that everybody (the members of *That Group*) can commit to the specific project. Most of us are taking other classes and have other commitments to attend to, that have made scheduling difficult.

Design

Design Decisions

Our initial website design in the first phase was very basic. Our main page simply listed our 4 organizations, crisis, and people which each had a hyperlink that would take you to its own individual page. In the first phase we were to create a static HTML page for each crisis, organization and person so we used the same html design for our 12 pages with their own data. The title of each page was centered at the top of the page and then we used a side panel to have links to the other 11 pages and then listed the data in the middle. The design was very simple but we knew that we were going to change it later and in this phase we wanted to make it a specific focus to clearly show all the data. This simplistic design allowed us to insure many things about that the website. It proved to us, that there existed navigation between the static HTML pages. We looked and tested for any broken links. Also it allowed us to experience a horribly layout of our website! We noted what we wanted to improve and how we wanted to go about improving it.

One of the biggest nuisances was having all the link to the 12 pages on the splash page. We knew this would be impractical, since in the later phases we were going to import the information of all

the other groups in the class. Our original website also listed the links to the 12 other pages on the side as you were navigating through each static HTML page. We knew that our new websites could simple not just list links to every page if we wanted to present it in a coherent manner. In phase 2 we were now required to make a design for our website that made it presentable.

From the start, we have taken the emulation of IMDB to heart, and decided that IMDB would be the heart and soul of our newly designed website. From scratch, we started making a template that resembled the website design of IMDB. We are a bit astonished at how similar we have gotten our website to look like the actual IMDB website, although there are major differences since we have to make our template to the xml schema.

Our original xml schema, we felt, was a very good one. The only attributes that we made as "required" were those that were stated on the project website. For each crisis, person, and organization we listed the attributes and the information for each one. Below is a sample for one crisis in our original schema:

```
<crises>
  <crisis id="abduction">
    <name>Abduction/abuse of children by the Lord's Resistance Army</name>
    <kind>Child abuse</kind>
    <location>Uganda, South Sudan, Democratic Republic of the Congo, Central African Republic</location>
                                <date>1987 - 2012</date>
    <humanImpact kind="Killed" note="since December 2009">1105</humanImpact>
    <humanImpact kind="Missing" note="since December 2009">2519</humanImpact>
    <humanImpact kind="Displaced" note="since 1987">2000000</humanImpact>
    <economicImpact note="unknown">0.0</economicImpact>
    <resourceNeeded>money, health care, military</resourceNeeded>
    <wayToHelp link="https://www.stayclassy.org/checkout/donation?eid=14711">Donate</wayToHelp>
    <image>
      <link>http://tinyurl.com/7zc5ows</link>
    </image>
                                <video>
                                        <link>http://youtu.be/-ip08pjKngI</link>
                                        <title>The Lord's Resistance Army Hunts Children in Sudan</title>
                                </video>
                                <social kind="twitter">
```

```
                                        <link>https://twitter.com/#!/Pulitzercenter/lords-resistance-army</link>
                                </social>
    <extLink>
        <link>http://www.lracrisistracker.com/</link>
                                        <title>Lord's Resistance Army Crisis Tracker</title>
    </extLink>
    <organizationRef>invisible</organizationRef>
    <personRef>Joseph_Kony</personRef>
</crisis>
```

When the xml schema for the class has been picked, we saw where our original schema could have been improved. The class xml schema was more modularized. Below is an example for the same crisis shown above in the chosen class schema:

```
<crisis id="abduction">
        <name>Abduction Of Children By The Lord's Resistance Army</name>
        <info>
                <history></history>
                <help>Donations</help>
                <resources>Money, Health Care, Military</resources>
                <type>Child abuse</type>
                <time>
                        <time></time>
                        <day>1</day>
                        <month>4</month>
                        <year>1987</year>
                        <misc>Approximate/Ongoing</misc>
                </time>
                <loc>
                        <city></city>
                        <region>South Sudan</region>
                        <country>Uganda</country>
                </loc>
                <impact>
                        <human>
                                <deaths>1105</deaths>
                                <displaced>2000000</displaced>
                                <injured>50000</injured>
                                <missing>2519</missing>
                                <misc>Figures are approximate</misc>
                        </human>
                        <economic>
                                <amount>100000000</amount>
                                <currency>US Dollars</currency>
                                <misc>Unknown (Figures are estimates)</misc>
                        </economic>
                </impact>
        </info>

        <ref>
                <primaryImage>
                        <site></site>
                        <title></title>
                        <url></url>
                        <description></description>
                </primaryImage>
```

```
                <image>
                        <site>The Huffington Post</site>
                        <title>Limbaugh Lord's Resistance</title>
                        <url>http://images.huffingtonpost.com/2011-10-18-danzcolor4921.jpg</url>
                </image>
                <video>
                        <site>YouTube</site>
                        <title>The Lord's Resistance Army Hunts Children in Sudan</title>
                        <url>http://youtu.be/-ip08pjKngI</url>
                        <description>One of the world's most brutal rebel groups, Joseph Kony's Lord's Resistance Army
(LRA), is on the move from the Congo, terrorizing civilians</description>
                </video>
                <social>
                        <site>Twitter</site>
                        <title>Lords Resistance Army</title>
                        <url>https://twitter.com/#!/Pulitzercenter/lords-resistance-army</url>
                        <description>A public list by Pulitzer Center</description>
                </social>
                <ext>
                        <site>LRA CRISIS TRACKER</site>
                        <title>Lords Resistance Army Crisis Tracker</title>
                        <url>http://www.lracrisistracker.com</url>
                        <description>The LRA Crisis Tracker is a real-time mapping platform and data collection system
created to bring an unprecedented level of transparency to the atrocities of the Lord's Resistance Army.</description>
                </ext>
        </ref>

        <misc></misc>

        <org idref="invisible" />

        <person idref="Joseph_Kony" />
</crisis>
```

This modularization has made it easier to create the data models that we use to store the information from the xml schema (Datastore for more information on the models). As is shown, the new schema made our information much longer than before. We do disagree with some of the changes that we were forced to make to conform to the new schema. The class xml schema required more than what is listed in the project specifications. Some of these new required elements we simple could not information for in regards to our selected people, crisis, and organizations, so we simply chose to list for those elements "information was not collected". We would have preferred is those elements were optional as they are not general enough to apply to a wider base. However, we had to conform to the new schema and make the transition, which was not too difficult. Our schema had the majority of the elements in place. The

majority of the switch was the modularization of our already existing schema. We should have thought about the schema as a type of program with functions, so we could have included the modularization from the beginning.

Our UML models are a direct representation of the schema. The way that the schema was modularized is the same way that our UML models are modularized. We felt that designing our UML models in this way would give us the easiest time in importing information to them from our schema, as we started to try and fill in the datastore from an import. In addition, since everyone now has to conform to the new schema, then our models will also be a direct representation of the way that their information is stored in their instance of the schema. When we tried to make our pages dynamic, however, we ran into a couple of problems. Our first design of the models was not specific enough and they left a lot of variables unread. Now, however, we iterate through the entire schema, searching for the correct name of the information that we want to store. We think that there may exist and easier way to fill in our models, but we have not been able to figure it out.

Special Features Added

To this point in our project, we have not added any special features other than those that are required by the project specifications. We have first tried to implement our solution in a matter of successive steps, starting simplest solution. As we have progressed through our steps, we have hit major roadblocks (elaborated more in the section Difficulties) and have not had the time or luxury of thinking of special features available to the user. However, even if by then end of phase 3 we have not added any extra features, we hope to add features after the course is complete. The website is going to continue to be something that the members of *That Group* will continue to improve. We hope to show this website to future employers, so that they can have a visual and physical representation of the work that we are

capable of completing. As time passes and our knowledge expands, our skills will improve and we hope that our website will continue to reflect this improvement.

Implementation

Solution for Frameworks and Tools

In the process of creating and improving this website, we used a very wide variety of resources, tools, and frameworks. The most obvious tool that we used was the Google Appspot Engine. Through the Google App Engine we created our appspot which serves our website. The appspot itself can be programmed in different three different languages: Python, Java, and GO (an experimental language created by Google). As required by the project specifications, our appspot had to be programmed in Python, although we were given the option of either programming in Python 2.5 or Python 2.7. We chose to program in Python 2.7 since more recent features would be available to be used at our disposal.

Professor Downing gave us many links to helpful tools that were at our convenience and benefit to use; however, we found no tool more helpful than Google and Youtube. We were not taught how to use or program the appspot, and all of us members in *That Group* had no experience in using the appspot beforehand. As a result, we walked into this project blindly and struggled mightily during the first days of starting this project. We did find a wide variety of help on Youtube (a website that hosts a large amount of user submitted videos) as Google even had a channel (a collection of videos from the same user) dedicated to providing help with the Google App Spot. Also, whenever we faced a problem that we could not figure out, we simply searched the internet with the state of our problem. As it turns out, many people have run into the same types of problems that we had and we hope the same is true to those problems that will continue to rise.

Another reliable tool was Piazza, which is a website that every member in the class is a part of. In Piazza students post questions and other members of the class, the teaching assistants, or Professor Downing respond with an answer, or something that helps them to the right direction. As a result of the majority of the class never using the Google App Engine before, most of us were struggling in the beginning. In addition, along the road we kept facing the same roadblocks as other, so it was really helpful when the solution to our problem was already solved on Piazza.

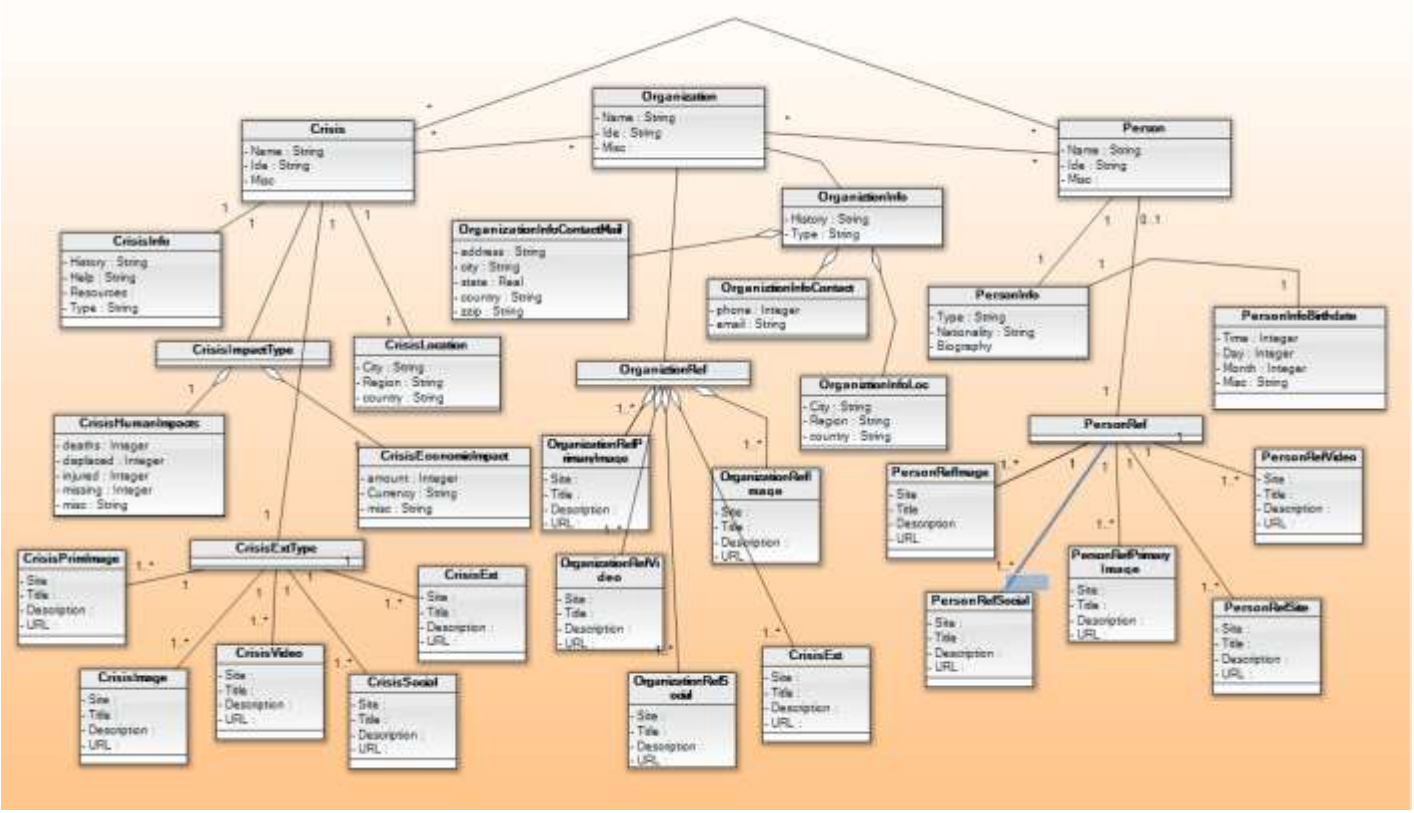
Solution for Datastore

In phase one we weren't required to use the datastore so we simply imported the .xml file into the *elementTree* data structured and spewed it back out with the export button. Now that we were required to use GAE datastore we had to figure out a model diagram for us to store the data imported and later be able to retrieve and export from the GAE models. We decided that we were going to let organization, person and crisis have their own class and then have them be parents of other subclasses that are required in the schema. This would also allow for the each Organization/Person/Crisis to have a many to many relation with each other. The crisis class used location, impact, extension and info as a subclass which then each had their own smaller subclasses. The organization class only used Info and Ref as a subclass. Organization's Ref subclass was very similar to Crisis' extension subclass in that it had similar subclasses with the exact same attributes. Even though we could have joined them in one class and have Organization/Peron/Crisis have separated associations we decided to have each "Parent" class have its own separate hierarchy. We did this in order to make the importing of the xml data to these models much easier to iterate and save to the datastore. The person class used only Info and Ref as a subclass with Info using Birthdate as a subclass and Ref being the same as OrganizationRef. Once

our models were set we then used *ElementTree* to go through the imported xml and create and save our class instances into the datastore.

Solution for Import/Export

With the help of *elementTree* we were able to import our .xml file into a tree data structure and then simply print it back out with the .out.write() function giving it the imported xml file back in a string format. At first we had trouble on how it was being displayed in the browser but found an easy fix by setting the .content_type equal to “text/xml” from the hint on the project page. Since we weren’t supposed to use the datastore in the first phase we were able to accomplish this fairly easily. By saving the tree in a global variable and using this tree across our import/export functions we were able to output the xml that we imported as long as we did not close the browser. If you were to import the xml and then close the browser, the export function would not have worked in a new browser since there was no global variable initiated in that instance. In the second phase we were to use the datastore and save the imported xml into GAE models that were then to be saved to the datastore. Below is a sample of our models:



The models are a direct resemblance to the chosen XML schema. We found that making the models this way would make it easier to store the information and in turn use that information to import and export. Now, whenever we import an xml file it saved and we can close the browser and export it at a later time without having to import first.

Evaluation

Testing Solution

With the papers and that have been assigned to us in class, it has become very apparent to us, that we must test first and code second. We have taken this message to heart and have made it a point in our group to test everything that goes into our final version of code. Of course, we are not yet experts at this technique so bugs that might have been caught by effective testing have fallen through the crack and we

have paid dearly in the form of time because of those errors. The GAE unit testing is a helpful tool that has caught many bugs for us. However, it was very difficult for us to learn how to use this tool.

We spent a lot of time during the first phase in learning how to set up The GAE unit tests so that we could use them. We searched the Internet for help and we always ended up at the same information, which was given in the GAE unit testing README. It said that after adding a couple of lines to the app.yaml file and then making a directory named "test" in the same directory that the gaeunit.py file was located, then the testing would be available for use. However, this did not work for us as it kept giving us a 404 error. We first figured out that in the directions it says to add gaeunit.py in our app.yaml file as the script for the url "/test.*", but this was incorrect. We are coding our solution in python 2.7, and the instructions were meant for python 2.5. As a result, we had to change many things in order to correct this situation. First, instead of the script being gaeunit.py, we instead wrote gaeunit.app as the script. Then we changed the many references of webapp in the gaeunit.py to webapp2. After we got it working, we made tests for everything that we already had in place, which in turn helped us catch many headache inducing errors that had been plaguing us for quite some time. Now in phase 2, we were able to enjoy this tool from the start and have followed the suggested model of making tests, and then coding.

Critiques

Our Own Critique

What did we do well? We believe that our website can be easily navigated. It is not clumped together and it can be enjoyable to the user. Going through some of the past projects, we saw that some of their websites looked cluttered at times. Those groups tried to be flashy and we think that they lost

sight of simplicity and made it difficult for the user at the cost of being more aesthetically pleasing. We kept it simple, and made sure that the user would have a good experience.

What did we learn? There are so many tools that coming into this project we had never heard of before. Now, however, we have added these tools to our toolbox. Each member got some time and effort in with every tool assigned to us, giving exposure to each member, and allowing each member to gain experience, something that will be beneficial in the work place. We also learned how to work as a team. Most of us had not had a great amount of experience working with other, but this experience certainly gave us that experience and much more. At first communication was not that great, but as we went along in the process we learned how to work with each other, and more importantly to trust each other's skill.

What can we do better? We don't think that our solution to the dynamic pages is efficient, in fact, we believe it is pretty slow. However, we wanted to first implement a simple solution and if we then had time, go back and try to optimize our code. Also, although our design is simple, it does lack in the "flash" and "wow factor" that other groups' websites possess.

What puzzles us? The search function has given us much trouble. We are still working hard and diligently to try and come up with a solution. Also, none of us is very experience with javascript, so we do not have the necessary experience to be able to add some of the effects that other groups' websites showcase. We hope to add more effects as special features before it is our time to present.

"All The Single Ladies" Critique

What did they do well? The home page is simple but it looks very presentable. We like how they list the home Crisis, Organizations, and People on their main website. The slideshow on the homepage

also looks very nice and the whole style has a modern feel to it. The individual pages have large text and different color text that makes for an attractive page.

What did we learn from their app? We learned how they were able to use the slideshow using the Primary image to scroll around. The dropdown list on the header was very attractive but it might get too big when you import all the data. It's pretty easy to flow throughout the website. We have also taken into consideration their asthetics as they have made us question some of the choices that we have made.

What can they do better? At the time of the critique the search wasn't working as specified. Also the rest of the groups' information was still not uploaded, as per the specifications of the project. In addition, they seemed to have too much information, instead of a short summary that quickly informs the reader about the crisis at hand.

What puzzles us about their app? We believe that if they implement search better than it should be a very attractive website. We are really curious on how they will handle the search and if the specifications will be met. We are also wondering how they are able to make the slideshow on the homepage, as we would like to implement a similar feature on our own website.

Conclusion

This project has not been an experience. However, it has made realize what lies beyond the trivial programs that we are assigned in our computer science classes. One of the most valuable lessons, has been that of learning how to work in a team. For the majority of the group, this project was our first time ever being a part of a group. We have always been taught to work individually on our assignment, so it was a bit of an adjustment to be able to cooperate with peers. It was a welcome adjustment, though. To be able to collaborate and grasp the best ideas out of a group was a welcomed experience.

Individually this task would have taken many more hours, and would have let to many more hours of frustration. The image that a person has of a computer scientist is usually a man alone at his desk, programming his life away. However, it is clear from projects like these that computer science is a team driven field, where interpersonal relationship and teamwork are essential to a successful venture.

The pain that we experienced, trying to figure out how to use the many tools given to us was horrifying but invaluable. We spent hours and hours reading through documentation, watching tutorials, asking other people, trying to find help from every corner of the earth so that we could implement our designs through code. This was our greatest experience in this project, for in the future, we are not guaranteed to be working with software or tools that we are familiar with. Future employers are looking for someone who is self-driven and can quickly learn how to use new software. Without these skills, a prospective employee who only has one skill and has the inability of picking up others will be of no value to any employer.

After this experience, we feel that we are more prepared and ready for what lies ahead our career path. The product that we have ended up with is dear to us, but the experience that we have gained through this process will prove invaluable down the road.