**Louisiana Workforce Commission:**

**IT Awareness Project**

**Final Project Report**

University of Louisiana at Lafayette

Computer Science 453: Software Methodology

Team Name: Name

Scott Roddy

Joshua Delahoussaye

Bradley Milliman

Andrew Colvin

Stephen Mader

Alex Tiemann

**Executive Summary**

This content of this document describes the development and its multiple stages of the product for the Louisiana Workforce Commission: IT Awareness Project. The document will also portray the choices made during development and inform why they were chosen.

**Table of Contents**

*1.Introduction…………………………………………………………………………….4*

*1.1. Purpose and Scope………………………………………………………..4*

*1.2. Product Overview………………………………………………………….4*

*1.3. Structure of the Document………………………………………………..4*

*2. Project Management Plan………………………………………………………...5-8*

*2.1. Project Organization……………………………………………………….5*

*2.2. Lifecycle Model Used……………………………………………………...5*

*2.3. Risk Analysis……………………………………………………………….5*

*2.4. Hardware and Software Requirements………………………………..5-6*

*2.5. Deliverables, Schedule…………………………………………………....6*

*2.6. Monitoring, Reporting, and Controlling Mechanisms…………………..6*

*2.7. Professional Standards…………………………………………………....6*

*2.8. Evidence All the Artifacts Have Been Placed Under Configuration Management…………...………………………………………………………..7*

*2.9. Impact of the Project On Individuals and Organizations……………7-8*

*3. Requirement Specifications………………………………………………………....8*

*3.1. Use Case Model For Functional Requirements…………………….…..8*

*3.1.1. Graphic Use Case Model……………………………………….8*

*3.1.2. Textual Description for each use case…………………….8-11*

*3.1.2.1. Veteran Site Access………………………………...8-9*

*3.1.2.2. Teenage Site Access………………………………….9*

*3.1.2.3 Child Site Access…………………………………...9-10*

*3.1.2.4 Admin Site Access………………………………..10-11*

*3.3. Rationale For Use Case Model………………………………………....11*

*3.4. Non-Functional Requirements………………………………………….11*

*4. Architecture……………………………………………………………………...12-15*

*4.1. Architectural Style(s) Used……………………………………………...12*

*4.2. Architectural Model…………………………………………………...12-14*

*4.3. Technology, Software, and Hardware Used…………………………..14*

*4.4. Rationale For Architectural Style and Model…………………………..15*

*5. Design…………………………………………………………………………....16-20*

*5.1. GUI (Graphical User Interface) Design………………………………...16*

*5.2. Static Model - Class Diagrams………………………………………….16*

*5.3. Dynamic Model - Sequence Diagrams………...……………………....17*

*5.4. Rationale For Detailed Design Model………………………………….18*

*5.5. Traceability From Requirements To Detailed Design Model…….19-20*

*6. Test Plan…………………………………………………………………………20-23*

*6.1. Requirements / Specifications-Based System Level Test Cases…...20*

*6.2. Traceability of Test Cases To Use Cases………………………….20-21*

*6.3. Techniques Used For Test Generation…………………………….21-22*

*6.5.1. Cause-Effect Graph………………………………………..21-22*

*6.5.2. Error Guessing………………………………………………….22*

*6.5.3. Black/White Box Testing………………………………………22*

*6.5.4. Criteria For Measuring the Quality of Tests…………………22*

*6.4. Assessment of the Goodness of the Test Suite……………………….23*

1. *References………………………………………………………………………....23*

**List of Figures**

*2.8.1. Google Drive……………………………………………………………………..7*

*2.8.2. GitHub…………………………………………………………………………….7*

*3.1.1. Use Case Diagram……………………………………………………………....8*

*4.2.1. MVC Pattern Diagram………………………………………………………….12*

*4.2.2. Web Layout FSM Diagram…………………………………………………….13*

*5.1.1. GUI Design……………………………………………………………………...16*

*5.2.1. Class Diagram………………………………………………………………….16*

*5.3.1. Sequence Diagram…………………………………………………………….17*

*5.5.1. Traceability Model……………………………………………………………...19*

*6.5.1. Casue-Effect Graph…………………………………………………………....21*

1. **Introduction**
   1. **Purpose and Scope**

This document will give a description of project management, requirements, architecture, detailed design, and the test plan for the “Louisiana Workforce Commissions: IT Awareness Project” web application. It will outline the purpose and definition of the system, explain each part of the system and the interactions between these parts to form a functional system, and provide an overview of what is required to develop the web application. The document will also define the design and coding standards whilst also defining the correct test plans and their effectiveness.

* 1. **Product Overview**

The end product is capable of displaying the correct data through simplistic means on a web page conforming to the LWC standards. Scenarios fully tested and planned for in the project encompass all age groups and the specific needs of veterans. Other scenarios tested for include slow or fast internet connections and user ease of use.

* 1. **Structure of the Document**

The document is structured in a manner to allow for easy identification of project management, specifications, architecture, detailed design, and the test plan used. Each section will be properly formatted for readability with orderly subsections as required. All figures will include a title and description if needed. All sections and subsections will be listed in a table of contents while figures will be listed in a list of figures all with their appropriate pages they appear upon.

1. **Project Management Plan**
   1. **Project Organization**

Joshua has been tasked with maintaining communication continuity within the team as well as the link between the team and employer. Alex and Andrew have the most functional knowledge of the languages and scripting languages we will be using for the website and will be able to provide help and direction to our other coding members Scott, Bradley and Stephen. Each member will be expected to code but our member’s with more experience will help us break down the tasks between members as well as guide us on the directed learning each member will need to complete their specific task.

* 1. **Life Cycle Model**

The site we are making is going to display information regarding current and future jobs for the next 5 years. So the current estimated lifecycle will be projected for those 5 years. The plan is to code the site in a way that would facilitate future changes and improvements however so that with a little work and tweaking the lifecycle can be indefinitely extended should the owner choose to invest more time into it.

* 1. **Risk Analysis**

There is little in the way of risks for the employer because no private or secure information will be put into the hands of our team or the end product. The risks the team faces all primarily affect its ability to produce the final product in working order and in the allotted time. Acts of god, injuries and downright laziness are all factors which can contribute to a higher rate of failure but with proper task and time management we hope to have everything finished ahead of the due date to allow for extra time to polish any problems caused by the previously listed risks.

* 1. **Hardware and Software Requirements**

The platform we have chosen should run seamlessly across all platforms and devices. It will all be hosted by the company’s existing servers but a single web page as we will have it designed will require very minimal storage and should not have much, if any impact on their throughput since data is not added or affected by the user. We are going to use tools such as Github, and Trello for organization and version control. Google docs will be used for joint documentation and file sharing outside of the code itself. And we will primarily use the Netbeans IDE and eclipse for general coding and testing.

We have made use of the jsoup API. This API provides us with a scraper which we can implement to gather the information needed from laworks.net. The API also provides features for sorting through HTML and generating DOM trees. The API is written in java, which was one of the primary reasons java was chosen to write the backend of the web application. Javascript and HTML will be used to create the front end of the application. Javascript integrates smoothly with java so connecting the two halves of the project together.

* 1. **Deliverables, Schedules**

Project Management Plan: 9/30

Requirements Documentation: 9/30

Architecture Documentation: 10/9

Detailed Design Documentation: 10/30

Test Plan: 11/10

Final Report: 11/27

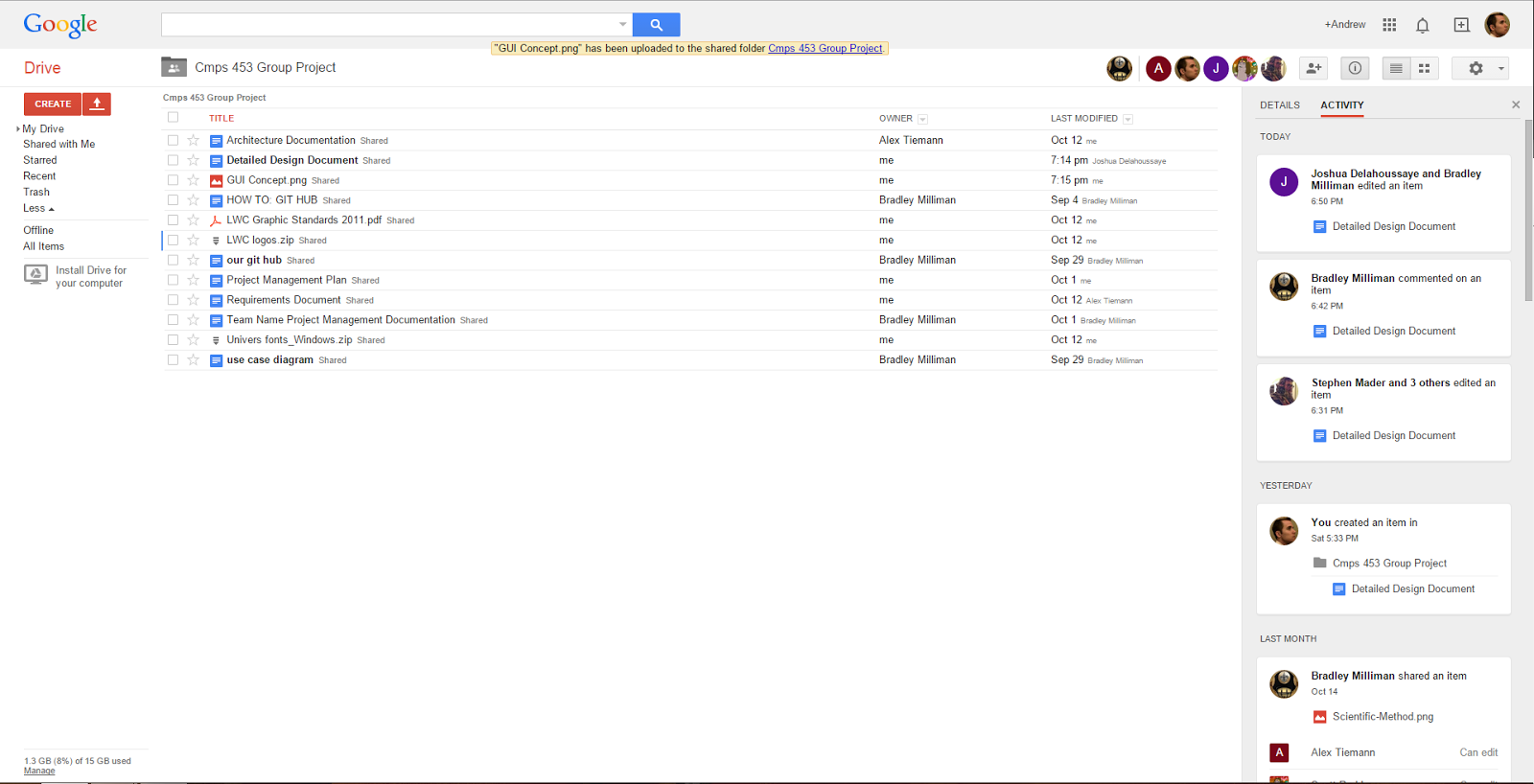
* 1. **Monitoring, Reports, and Controlling Mechanisms**

Progress can be viewed and monitored via github. If a mistake is published accidentally it has functionality to go back to a previous working state to prevent a few changes from ruining working parts.

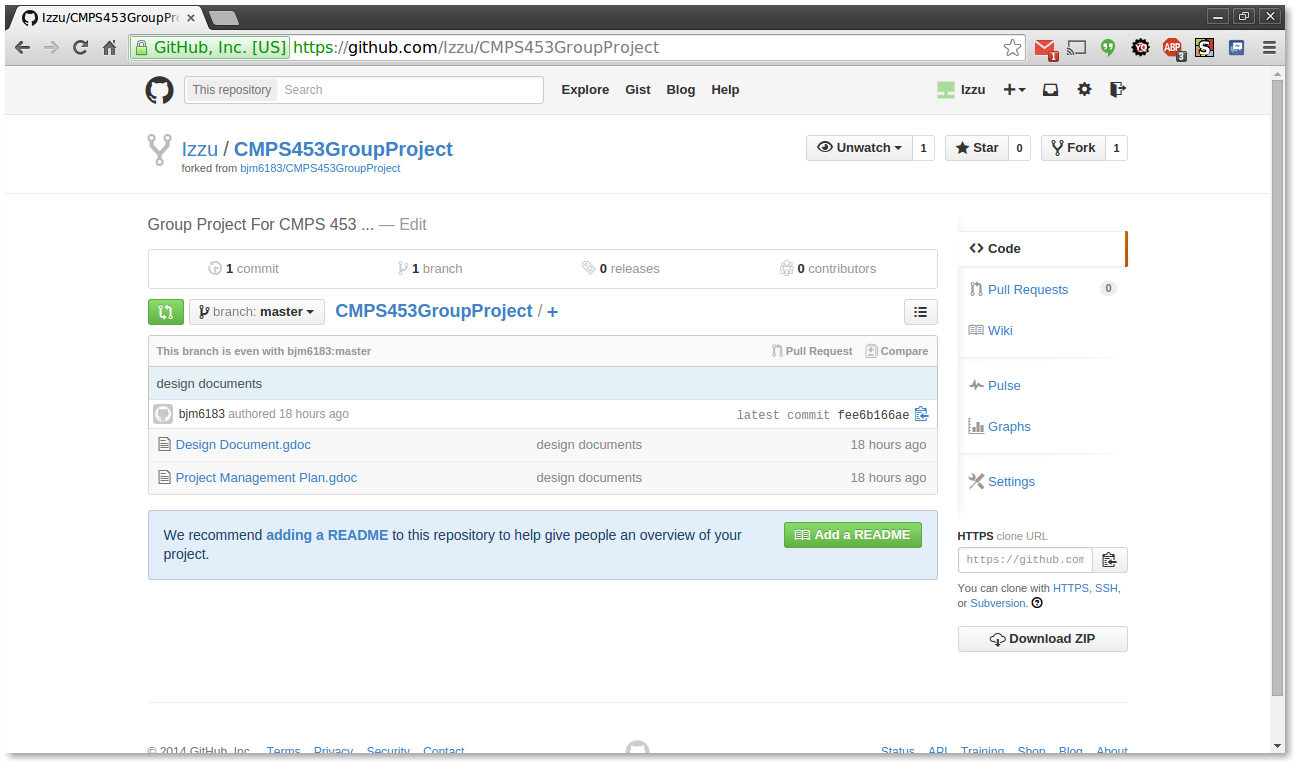
* 1. **Professional Standards**

Each team member is expected to take part in all tasks leading up to and during software production. We are all required to clearly communicate our wishes and abilities. Respect and understanding are of the utmost importance during any and all interactions involving team members with the employer and amongst ourselves. Good professional relationships are the surest way to get your foot in the door at any company. It also expedites the process of hearing about and obtaining future employment and opportunities so building strong and positive relationships now is an invaluable investment in your own future.

* 1. **Evidence All the Artifacts Have Been Placed Under Configuration Management**



*figure 2.8.1 Google Drive*

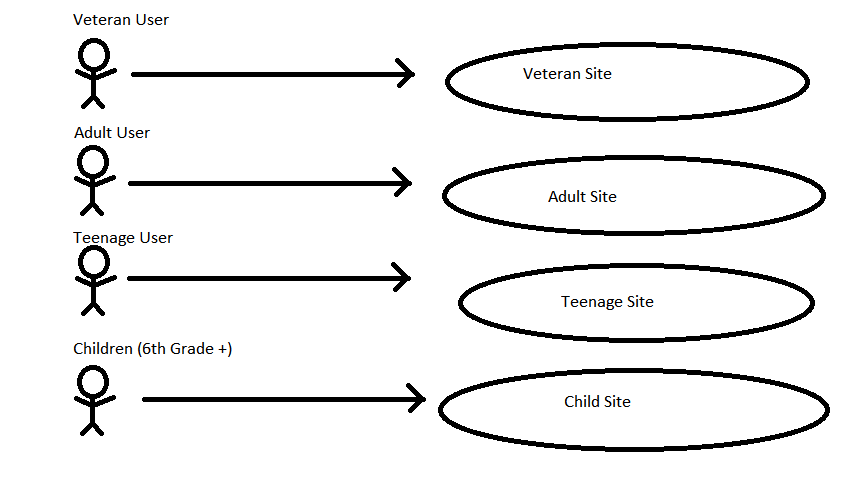


*figure 2.8.2 GitHub*

* 1. **Impact of the Project On Individuals and Organizations**

Our project will have an outstanding impact on the individuals seeking a career in the field of Information Technology, along with the companies seeking new talented professional, and the institutions that can provide the proper training and education for these individuals. An increase in employment could be a possible outcome as the information regarding upcoming IT related careers is made readily available to the public. Businesses and large corporations will begin to choose Lafayette as the new home for its branches and headquarters. We can expect to see in an increase in the number of new students enrolling in the computer science and informatics classes at the local colleges and university.

1. **Requirement Specifications**
   1. **Use Case Model For Functional Requirements**
      1. *Graphic Use Case Model*

figure *3.1.1. Use Case Diagram*

* + 1. *Textual Description*
       1. *Veteran Site Access*

*Participating Actors*

Veteran User

*Entry Condition(s)*

The Veteran User enters the Veteran User portal on the main site.

*Normal Flow of Events*

The Veteran User enters the site. They are presented with information regarding the various features of the site, and methods with which to access them. From this point, the Veteran User may access and search for information, charts, and videos.

*Exit Condition(s)*

The Veteran User enters the main site again, or closes the browser.

*Special Requirements*

* + - 1. *Teenage Site Access*

*Participating Actors*

Teenage User

*Entry Condition(s)*

The Teenage User enters the Teenage User portal on the main site.

*Normal Flow of Events*

The Teenage User enters the site. They are presented with information regarding the various features of the site, and methods with which to access them. From this point, the Teenage User may access and search for information, charts, and videos.

*Exit Condition(s)*

The Teenage User enters the main site again, or closes the browser.

*Special Requirements*

* + - 1. *Child Site Access*

*Participating Actors*

Child User

*Entry Condition(s)*

The Child User enters the Child User portal on the main site.

*Normal Flow of Events*

The Child User enters the site. They are presented with information regarding the various features of the site, and methods with which to access them. From this point, the Child User may access and search for information, charts, and videos.

*Exit Condition(s)*

The Child User enters the main site again, or closes the browser.

*Special Requirements*

Readable by children.

* + - 1. *Admin Site Access*

*Participating Actors*

Admin

*Entry Condition(s)*

The Admin logs into the admin portal on the main site.

*Normal Flow of Events*

The Admin enters the site. They are presented with the site, and methods with which to access them. From this point, the Admin may access and search for information, charts, and videos, the same as a user, and also change content on the site.

*Exit Condition(s)*

The Admin logs out, or closes the browser.

*Special Requirements*

* 1. **Rationale For Use Case Model**

After a meeting with the Louisiana Workforce Commissions office and discussing what they would like to achieve with this project we have come to the conclusion that they would like a functional website for each of the three different user types within one main website as shown in the use case model. Also assuming they would want to edit this site in the future, a fourth user would be needed for editing permissions.

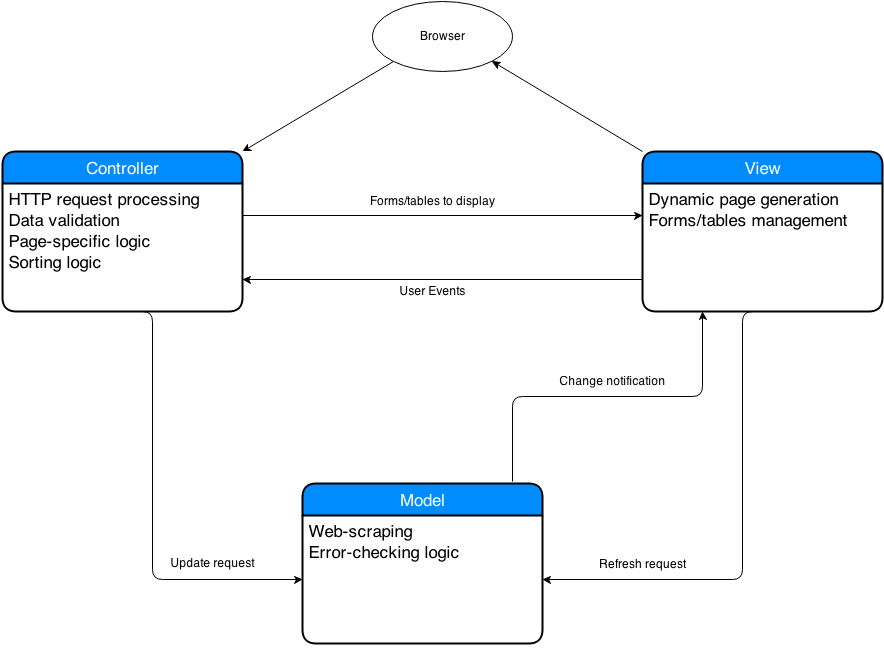
* 1. **Non-Functional Requirements**

The final product needs to be stable, easy to use and navigate. The Child User section of the website needs to use basic english and larger fonts than the other sections of the website. The final product needs to be maintainable, and the information presented needs to be updatable by the administrator of the site.

1. **Architecture**
   1. **Architectural Style(s) Used**

As a whole the architecture helps to support the flow of user control. Ensuring that the user is on the correct path and that there is no confusion for the user. It also helps to support the maintaining of the website in an organizational standpoint. The architecture helps to simplify the process of designing and structuring the website by laying the foundations on which to build upon and around.

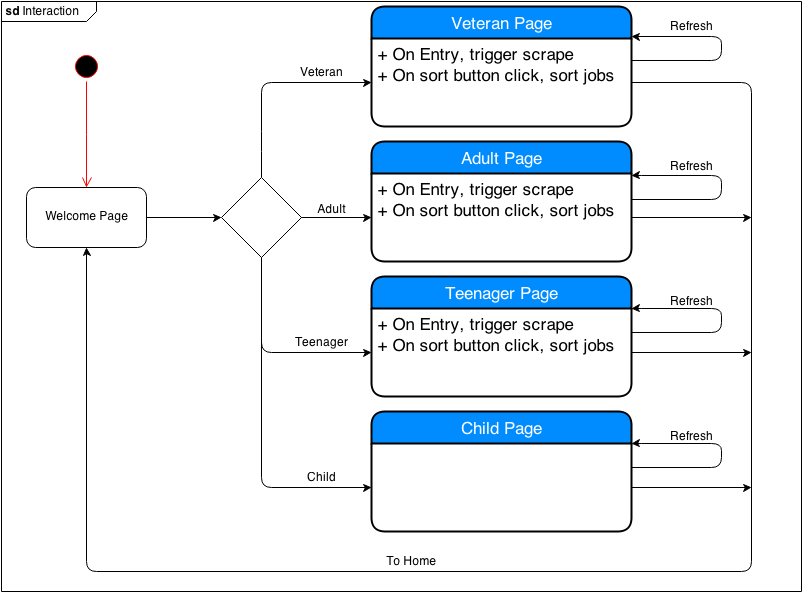
* 1. **Architectural Model**



*figure 4.2.1. MVC Pattern Diagram*

*Figure 4.2.1 Description*

The controller tells the view which forms and tables to display. It also handles the sorting and page-specific logic of the site. The model is polled by the controller for the jobs that are available and scrapes the LA Works site for information. The view handles the presentation of the tables and forms sent to it by the controller, generating html to display it neatly on the page.



*figure 4.2.2. Web Layout FSM Diagram*

*Figure 4.2.2 Description*

The welcome page will link to one of four subpages. On the veteran, adult, and teenager pages, on loading the page, the LA Works website will be scraped for information, which will be presented differently based on which page the user navigated to. The user will then be able to sort the information according to their preferences. The child page will be a static informational page.

* 1. **Technology, Software, and Hardware Used**

The web page will be implemented through use of Java and Javascript as the primary coding and scripting languages. To solve the issue of version control we plan to use the git and/or github which allows each member to work individually and automatically generates backups of all previous saves. This way any possible mistakes can be dealt with and fixed rapidly. When it comes to development environments, Eclipse is a very powerful tool for java related applications and Alex Tieman is already proficient in its use which will be a great asset for other members who have questions.

Since our employer only wants a straight forward and simplified version of their current site, we do not foresee ourselves needing to use any database. The application will be stored on their current servers and will require little to no added bandwidth overhead since the data we take in will be as simple as them clicking their age group and the data output will be a shortened form of what their site would display normally. This “less is more” approach has been chosen to keep the user from getting bored, giving them only the information they need and ask for, and help lessen the headache the Workforce Commission workers currently go through when trying to find information for their own use and when talking users through the process.

* 1. **Rationale For Architectural Style and Model**

We chose to use the Model View Controller (MVC) pattern to describe the architecture of our system because it best fits the needs our application. The system is structured into three logical components, the model, the view and the controller, that interact with each other. The MVC pattern appears to be the most useful in describing web based applications such as this project, where there is no interaction with other systems or servers, with the exception of the web server to host the application. The advantages of the MVC pattern work well for the agile programming style that our development team is using. Some of these advantages include the ability to allow the data to change independently of the way it represented and the other way around.

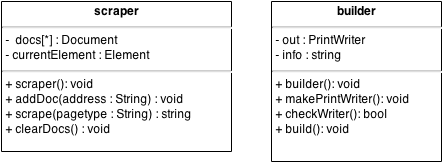
1. **Design**
   1. **GUI (Graphical User Interface) Design**



*figure 5.1.1. GUI Design*

* 1. **Static Model**

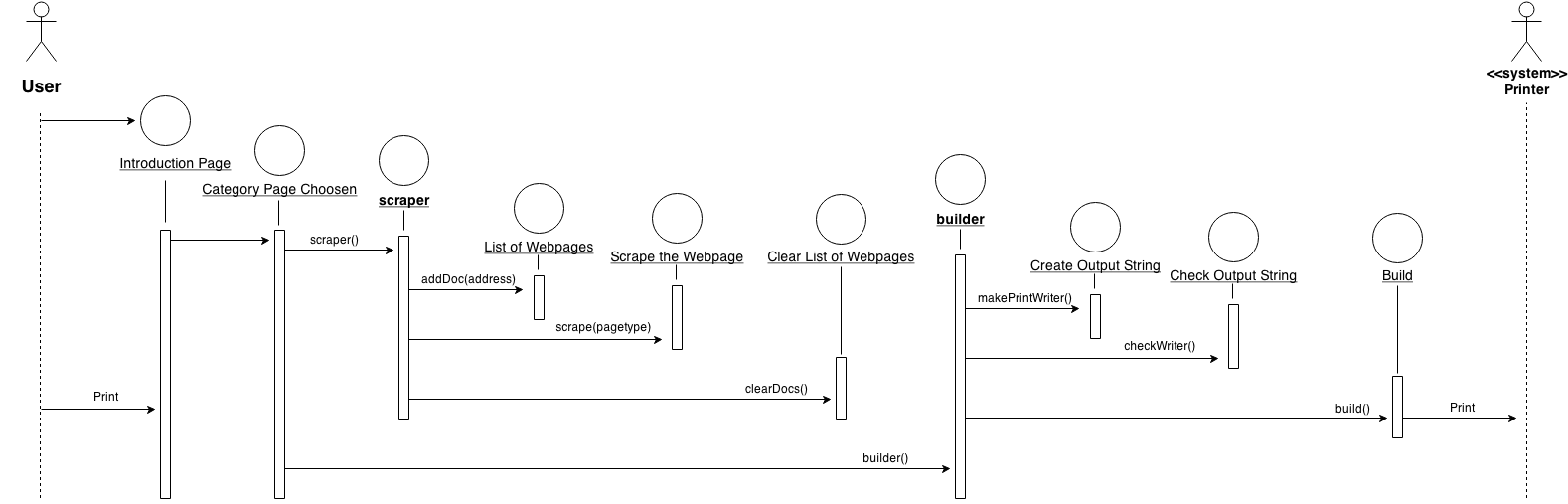
*Class Diagrams*



*figure 5.2.1. Class Diagram*

* 1. **Dynamic Model**

*Sequence Diagrams*



*figure 5.3.1. Sequence Diagram*

* 1. **Rationale For Detailed Design Model**

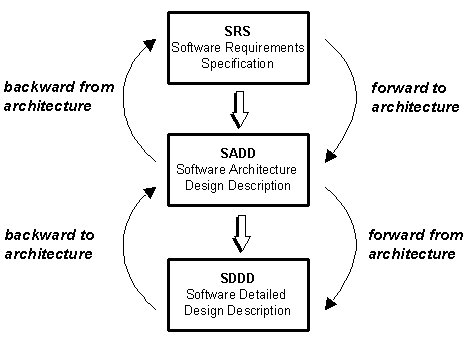
Reasons and Justification - We decided to use JSOUP for our website scraper to pull the relevant information needed to rebuild the new site. JSOUP is a Java library that works with real-world HTML. The API is convenient for extracting and manipulating data and it uses some CSS, DOM, and jquery-like methods. We decided on Java because most of our team has Java or C# in our former programming language background. Java is also an easy platform to work with websites.

Other Alternatives - Our language alternative that was weighed against using Java was Ruby. Java was chosen entirely based off the team’s background and the JSOUP library that will handle all of the website scraping needs.

Trade Offs - The common tradeoffs faced between choosing Java over Ruby were ease of use, popularity, readability of language, and the scraping library, JSOUP.

Argumentation Leading To Decision - There were no arguments leading to our decision. The team all agreed on JSOUP and Java for our platform and scraper.

* 1. **Traceability From Requirements To Detailed Design Model**



*figure 5.5.1. Traceability Model*

**Backward from architecture**

* All components of the architectural design satisfy a customer requirement
* Architectural design satisfies the customers requirements by having 4 separate pages one for each of the target groups and linking each group to their appropriate page.

**Forward to architecture**

* All customer requirements have been addressed by one or more architectural components
* No customer requirements have been overlooked.

**Backward to architecture**

* All detailed design entities implement a component of the architectural design
* By scraping all of the information that we want to find, and then building the website appropriate to the target group.

**Forward from architecture**

* All architectural design components have been implemented with one or more detailed design entities
* All architectural design components have been implemented by the detailed design.

1. **Test Plan**
   1. **Requirements / Specification-Based System Level Test Cases**

The listing below identifies items that have been identified as targets for testing. This list represents what will be tested.

* 1. **User Interface Testing**

Verify ease of navigation through the screens.

Verify screens conform to GUI standards.

Verify that the access/layout of information is more user friendly than the current site.

Verify that the data is displayed in the appropriate areas.

* 1. **Performance Testing**

Verify Response/load time of data from the scraper.

Verify that the correct data is gathered with the scraper.

* 1. **Traceability of Test Cases To Use Cases**

All user will send a request to the main site before selecting which age group they are in. This request will yield the page source for the main site. People requesting to view the child site, which is also static, will have the page source of the child site returned to them. The adult, teen, and veteran sites all function in a similar manner, allowing the test cases to be identical with the exception of the page source and data being returned. The user sends a request for a particular page, the request is then sent to the scraper. Following the test cases, if successful, the page requested will be returned containing the appropriate data pertaining to the request.

* 1. **Techniques For Test Generation**
     1. *Cause-Effect Graph*

*figure 6.5.1. Cause-Effect Graph*

A cause-effect graph is a directed graph that aps a set of causes to a set of effects. The cause may be thought of as input to the program, and the effects may be thought of as the output. The following figure 6.3.1 describes the cause-effect graph for the IT Awareness Project Site. The first two cases describe static pages, in which the input is a request to see a page whose information remains constant. The program should return the page source for the requested static page. When the input consists of a request to access the adult, teen, or veteran site, the scraper is invoked. Once the scraping is completed, the generator produces the appropriate page source based upon the input request. Finally, the program returns the generated page source for the requested page.

* + 1. *Error Guessing*

Error guessing is a test method in which test cases used to find bugs in programs are established based on experience in prior testing. This is being used as a backup method to expand test cases.

* + 1. *Black/White Box Testing*

Black box testing is being used for the jsoup API portions of our scraper. We are treating the jsoup API functions as if they were black boxes, as it is not feasible to spend the man hours to dissect the API on a line by line basis. The source code existing outside of the API implementation will undergo white box testing. Our source code is more easily tested in a white box style as the program can be stepped through at each line.

* + 1. *Criteria For Measuring the Quality of Tests*

Formatting of the returned page source will be compared to our mock-up design for each site to check for usability and adherence to the Louisiana Workforce Commissions standards. The content within each site will be checked against the content displayed on laworks.net to ensure that the correct data is being scraped from the site.

* 1. **Assessment of the Goodness of the Test Suite**

The cause effect graph showed an almost exact example of the predicted out come although, some of these fields differed. The first of the errors occurred when old data was compared to the most up to date data; regarding job demand rates and jobs available rates. This error/bug as well as the others to follow were corrected, in the same manner, as the new data was provided to the team. Otherwise, the cause effect graph proved to be accurate for other tests. Error guessing was used as one of the secondary testing methods. It was implemented in the program. Errors, once thought to arise, were proved false as each possible combination of site entry "clicks" were tried. However, errors poked through. The redirects to each page needed to be recorded. Once each additional address for the given link was added to the application these errors vanished. The site now operates within its implemented functionality, though certain sorting request run slow.

1. **References**

Test Plan pdf on the CMPS453 moodle page

Cause-effect graph wiki

UML 2.0

Sommerville Lecture Slides for 9th Edition

SpinGrid Detailed Design Document (<http://wwwis.win.tue.nl/2R690/projects/spingrid/ddd.pdf>)

<http://www.chambers.com.au/glossary/traceability_analysis.php>

Sommerville Lecture Slides for 9th Edition

UML 2.0 Tutorial from Sparx