Feasibility Study and Project Plan

CIS 4911 – Senior Project U01

Virtual Job Fair 2.0

**Members:**

Luis Irizarry

Jorge Fernandez

Luis Benjumea

**Instructor:**

Masoud Sadjadi

**Mentor:**

Juan Caraballo

**Date:**

10/28/2013

Copyright © Florida International University 2013. All Rights Reserved. No part of the Virtual Job Fair Project or documentation may be reproduced or modified without the express consent of Florida International University. Redistribution or commercial use is strictly prohibited.

**EXECUTIVE SUMMARY**

The Final Deliverable of the Virtual Job Fair 2.0 is meant to provide the reader with all of the relevant documentation and information regarding the project. Virtual Job Fair 2.0 is a website that connects students and employers and allows them to participate in interviews, even if they are physically distant. The project was initially implemented by a group of students FIU in a Senior Project class spring 2013. The task of Virtual Job Fair 2.0 is to add functionality that makes the system resemble even more an on-site physical interview. Throughout the document, explanations and descriptions of system, design and object structures are described.

Contents

[1. Introduction 8](#_Toc374351231)

[1.1 Problem definition. 9](#_Toc374351232)

[1.2 Scope of system. 9](#_Toc374351233)

[1.3 Overall development methodology 9](#_Toc374351234)

[1.4 Definitions, acronyms, and abbreviations 10](#_Toc374351235)

[1.5 Overview of document 11](#_Toc374351236)

[2. Feasibility Study 12](#_Toc374351237)

[2.1 Description of current system. Identify limitations and constraints 12](#_Toc374351238)

[2.2 Description of alternative solutions considered. 13](#_Toc374351239)

[2.3 Recommendation with explanation of why the solution was selected. 15](#_Toc374351240)

[3. Project Plan 18](#_Toc374351241)

[3.1 Project Organization 18](#_Toc374351242)

[3.1.1 Project Personnel 18](#_Toc374351243)

[3.1.2 Hardware and Software Resources 19](#_Toc374351244)

[3.2 Identification of Tasks, Milestones and Deliverables 20](#_Toc374351245)

[3.3 Cost of the Project 21](#_Toc374351246)

[4. System Requirements 22](#_Toc374351247)

[3.1 Functional and Nonfunctional Requirements 22](#_Toc374351248)

[4.2 Requirements Analysis 34](#_Toc374351249)

[4.2.1 Use case model 34](#_Toc374351250)

[4.2.2 Static model 34](#_Toc374351251)

[4.2.3 Dynamic model 35](#_Toc374351252)

[5. System Design (i.e., overall system design) 36](#_Toc374351253)

[5.1 Overview 37](#_Toc374351254)

[5.2 Subsystem Decomposition 37](#_Toc374351255)

[5.3 Hardware and Software Mapping 41](#_Toc374351256)

[5.4 Persistent Data Management 42](#_Toc374351257)

[5.5 Security/Privacy 59](#_Toc374351258)

[6 Detailed Design 60](#_Toc374351259)

[6.1 Overview 60](#_Toc374351260)

[6.2 Static model – detailed description of the structure for each subsystem. 62](#_Toc374351261)

[6.3   Dynamic model 66](#_Toc374351262)

[6.4 Code Specification 68](#_Toc374351263)

[7 System Validation 73](#_Toc374351264)

[7.1 Subsystem Tests 73](#_Toc374351265)

[7.2 System Tests 76](#_Toc374351266)

[7.3 Evaluation of Tests 76](#_Toc374351267)

[8. Glossary 77](#_Toc374351268)

[9. Appendix 78](#_Toc374351269)

[9.1 Appendix A 78](#_Toc374351270)

[9.2 Appendix B 78](#_Toc374351271)

[9.3 Appendix C 129](#_Toc374351272)

[Log In Using FIU SCIS Credentials 132](#_Toc374351273)

[9.4 Appendix D 133](#_Toc374351278)

[9.5 Appendix E 134](#_Toc374351279)

[Static model 135](#_Toc374351280)

[Dynamic model 138](#_Toc374351281)

[9.6 Appendix F. 141](#_Toc374351282)

[9.6 Appendix G 141](#_Toc374351283)

[9.7 Appendix H 159](#_Toc374351284)

[10 References 169](#_Toc374351285)

# Introduction

      The introductory chapter gives some background information about the Virtual Job Fair system. Firstly, the chapter states the current problem with the interview process of companies, and some background on this problem is provided, including its scope. Next, the design methodology used is identified. This methodology includes the software process models and the types of models used. Moreover, definitions, acronyms, and abbreviations of terms that will be used in this deliverable are introduced and explained. Finally, it contains an overview of the whole project, which explains the information contained on each chapter.

## 1.1 Problem definition.

Employers looking for talent are always interested in filling out positions with the best possible people. In order to accomplish this task, the most effective method to date is to tap local talent, whether it is at universities or job fairs. Given the increasing globalization trend, and the fact that not all employers have the financial or logistical capabilities to seek for potential candidates in different locations, the current solution provided by universities and job sites is less than ideal.

Our solution to this problem intends to provide a more transparent interview process that allows employer and potential employee to interact as closely as possible and provide a better interview experience.

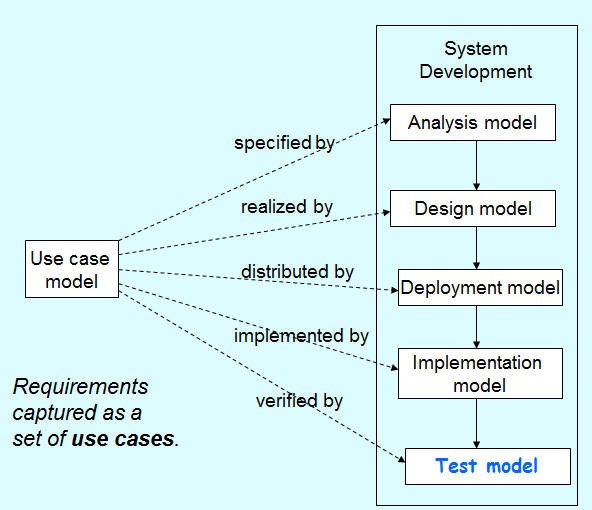
## 1.2 Scope of system.

One of the main purposes of the Virtual Job Fair is to expand the scope of the original. While the original project was geared towards Computer Science majors at FIU, our project expands on that by focusing on the global population. In our project, basically a student in any given university in the world can be interviewed by any employer located anywhere. This helps solve the global problem of employers being unable to interview students who might be potential employees who are located remotely.

## 1.3 Overall development methodology

We will take advantage of the Unified Development Process in order to develop the new system and add the necessary functionality.  Since USDP iterative and use case driven in nature; we will focus our attention to adhering as much as possible to the functional requirements and gathered uses cases. It is very important to follow the rule imposed by this methodology in order to deliver a quality product that conforms and exceeds all standards.

We have found the iterative process to be very simple since it allows us to make small but concise progress towards the completion of the project. It is also important to understand the importance of the Unified Modeling Language (UML) and the significance of the models used to represent the design. We rely heavily in diagrams in order to visualize complicated concepts and devise solutions that are both efficient as well as robust.



## 1.4 Definitions, acronyms, and abbreviations

**Definitions**

-          **Student**: an individual who is currently enrolled in the School of Computing & Information Sciences FIU

-          **Job:** an activity done in exchange for payment

-          **Full-time:** requiring 40 hours or more hours per week

-          **Part-time:** requiring less than 40 hours per week

-          **Paid internship:** an internship for which a student will receive compensation

-          **Unpaid internship:** an internship for which the student will not receive compensation

-          **Benefits:** non-salaried compensation for employees, such as insurance, tuition reimbursement, and retirement benefits

-          **Work authorization:** current legal work status of a student. Categories include U.S. Permanent Resident and U.S. Citizen

-          **Grade point average:** a number out of 4.0 which gives a representation of a student’s grades in his/her classes throughout his/her college career

**Acronyms**

-       **VJF**: Virtual Job Fair

-       **FIU:** Florida International University

-       **GPA:** Grade point average

-       **SCIS:** School of Computing & Information Sciences

-       **USDP:** Unified Software Development Document

**Abbreviations**

As of right now, there are no abbreviations for this project.

## 1.5 Overview of document

This document is divided in several sections. Section 2 goes over the current system and its limitations as well as the possible solutions and alternatives. Section 2.5 goes over the solution and a detailed explanation is given stating the reasons why this approach is significantly better than the other ones. Section 3 describes the project plan as well as all resources needed such as the cost of the project, software and hardware requirements, etc. Section 4 goes over the System requirements and includes diagrams that describe our current plan and implementation details. Section 5 goes over the System design section and goes into more detail about design patterns and subsystem decomposition. Section 7 discusses all the testing approaches that we used as well as more details about the tools we used and the test cases.

# Feasibility Study

The feasibility study chapter explores the idea of a virtual job fair from a practical point of view. Firstly, it considers the limitations of the current system, which was the project developed last spring. Also, it explains the purpose of the Virtual Job Fair, explaining how the features of VJF will improve on current problems. Then, high-level user requirements are described. Moreover, alternatives to certain aspects of VJF are considered and analyzed, with quantitative data used to support the fact that these alternatives were not used.

## 2.1 Description of current system. Identify limitations and constraints

The current system was implemented in the spring 2013 semester. As it is, it allows students and employers to engage in interviews through online video chats. It also allows employers to post jobs for students to browse and apply. Moreover, it offers a messaging system, through which employers and students can communicate privately. Also, the current system allows users to log in using their LinkedIn, Google or FIU account.

The current system also has standard functionality, allowing users to log in and log out, to register in the system, to edit their profiles, to change their passwords and to retrieve forgotten passwords.

**Limitations and Constraints**

As implemented, the current system has the following limitations/constraints:

**- Aimed towards FIU Computer Science students**: the scope of the system is geared towards students who are both FIU students and Computer Science majors. This leaves out two major groups of people:

a) Students who study at FIU but whose major is not Computer Science

b) College students who do not study at FIU, but at other universities instead

Therefore, it is geared toward a very narrow group of people, even when potentially any college student could benefit from this service

**- Limited interview functionality:** as implemented, the system only allows for video interviews and live chat, limiting the interaction between students and employers to a very basic level

**- Limited Document Collaboration functionality:** the current system has a very limited collaboration ability. It lacks the ability to have users of the system collaborate on documents in real-time which is a crucial part of an interview process.

**- Limited Priority for FIU Seniors:** even though the current system does not restrict any user from signing in, Priority should be given to FIU Seniors, given that this project was born from an FIU SCIS Senior course and it is has been strongly suggested by the faculty to include this feature in the system.

**- Inability to share images between students and employers:** the current system lacks an image-sharing feature which allows students and employers to exchange pictures which can enhance the interview experience

- **No drawing feature:** the current system does not have any type of drawing feature that may allow students and employers to brainstorm, exchange drawings or jot down ideas while interviewing

**- Reminder system:** the current system lacks a way of reminding students or employers of upcoming interviews

**- Users are not allowed to delete their accounts:** once registered, students and/or employees are not allowed to remove themselves from the system’s database

**-Employers are not able to contact students through other means other than by email:** The current implementation makes it very hard to keep both students and employers connected outside of it.

**-Students and employers are not reminded of important deadlines:** The current system does not have any functionality that alerts users of new events

## 2.2 Description of alternative solutions considered.

Below, alternative implementations for this project are mentioned and discussed:

**New System’s Features Alternatives**

**Collaborative Text Editor**

**Alternative 1**

Make use of the Google Drive API ( https://developers.google.com/drive/ ) in order to implement via their API some of the collaborative editing features required, and extend the features this solution does not provide, which are essential for the project.

**Alternative 2**

Make use of the Etherpad Collaborative Editor ( http://etherpad.org/ ) in order to provide some of the collaborative editing features required, and extend the features this solution does not provide, which are essential for the project.

**Alternative 3**

Make use of the Zoho Editor API, specifically the Zoho Remote API ( https://apihelp.wiki.zoho.com/) in order to implement via their API some of the collaborative editing features required, and extend the features this solution does not provide, which are essential for the project.

**Whiteboard Feature**

**Alternative 1**

In the case that the original whiteboard becomes unfeasible to implement, if time permits, integrate a different online whiteboard, such as Scriblink (http://www.scriblink.com). This might require learning a new API and completely changing the implementation, use cases and test cases. For this reason, this would only be considered if sufficient time is available.

**Alternative 2**

In the case of a problem with the whiteboard API calls or that there is no time for the first alternative, integrate a single, non-shared whiteboard for each user to allow them to draw during an interview. Then, also implement an image-sharing system which allows users to upload images to the system (including whiteboard drawings) to allow the other party in the interview to view the images.

**Blocking Feature**

**Alternative 1**

In the case the blocking feature becomes unfeasible, since it is a minor feature, the feature will be scraped from the project in order to allow for resources to be directed to the major features of the project.

**Screen Sharing Feature**

In the case that the screenleap is unreliable or unsuitable for the project, we would focus our attention on the screen sharing product developed by easyRTC. In the case that the screen sharing functionality become unfeasible a simpler alternative could be implemented such as one where only employers are able to share their screens.

**SMS messaging functionality**

In case that the twilio api is too complicated to implement or prove to be unreliable then we could change to a different service provider such as clockwork which also provides plenty of documentation and customer support. In the case where we lack time to implement all features, we could eventually scrap the notification functionality.

**Environment Software Suite Alternatives**

**Alternative 1**

Our current implementation strategy, which involves a LAMP stack.

**Alternative 2**

For the second alternative, instead of using the open source solution (LAMP Stack), we would implement the system using a closed source solution (.NET framework, IIS).

**Alternative 3**

For the third alternative, instead of using the open source solution (LAMP stack), we would implement the system using a different open source solution (Tomcat stack)

## 2.3 Recommendation with explanation of why the solution was selected.

**New System Feature Analysis**

**Collaborative Text Editor**

The following section contains analysis of the Collaborative Text Editor feature alternatives that were proposed on 09/09/13 and which have been revised at several points throughout the lifetime of the project.

**Alternative 1**

Google Drive API: After careful analysis and reevaluation of this alternative, we have concluded that the restrictive nature of this solution fails to provide the simplicity and integration necessary for the end user to be able to interact with the feature, which is one of the main goals of the project. The reasons for this are:

1. **Google Drive API**: Provides no means other than to have a Google Administrator Account ( Google Apps for Business ) to dynamically create, delete and manage users. Which would be required in order to dynamically provide a Virtual Job Fair user with a Google account in order to make use of the Google Collaborative Editor ( Implemented via the API ) This solution was discarded for The following Reasons:
2. **Google Administrator Account** ( Google Apps for Business ) has a cost of $5.00/user/month which proves extremely cost prohibitive for a Senior Project, as well as being unscalable. As the Virtual Job Fair system grows in users. The cost becomes, again, extremely high.
3. **Google Drive API:** Requires users to have a Google account. This problem was suggested to be overcome by creating a pool of Google user accounts. One of these would be selected for a Virtual Job Fair user at the time of login. This way that the Virtual Job Fair user could make use of the Google Collaborative Editor ( Implemented via the API ) This solution was discarded for The following Reasons:
4. Unmanageable/Unscalable as the Virtual Job Fair system grows in users.
5. Violates Google’s code of service.
6. Very high risk of having the pool of accounts ( or a subset of ) be marked as inactive and therefore deleted.

**Alternative 2**

1. **Etherpad**: This alternative was given initial extensive consideration, as it provides a simple solution that has a high capability to be integrated with the system. However, upon further examination, it was decided against for the following reasons:

1. Relatively new, which is evidenced by lack of robustness and feature set, compared to the other alternatives.
2. Lack of Security features. Documents created by etherpad can easily be shared between users via a single URL, giving potentially anyone the possibility of viewing a document. This represents a security concern for Virtual Job Fair users, given the delicate nature of the information shared.

**Alternative 3**

**1. Zoho Docs API:** After careful analysis of all the alternatives for implementing the Collaborative Text Editor feature, we have ascertained that Alternative #3 ( our current implementation ) is the one best aligned with the goals of the project. It is the most robust, reliable, feature rich, and security oriented of all the alternatives examined. Reasons for choosing this alternative are ( among others )

1. Seasoned, robust and reliable implementation, with a mature API which has been available for years.
2. Feature Rich, given that Zoho Docs provides 2 different API’s in order to interact and implement their features/service.
3. Highly manageable, given that the API imposes no restriction with regards to requiring a user to have a Zoho account to interact with the editor.
4. Highly Scalable, given that the API imposes no restrictions on the amount of simultaneous users supported.
5. Security oriented, given that one of the API’s ( the one chosen for our implementation ) requires documents to be hosted on the Virtual Job Fair servers directly.
6. Security oriented, given that the API does not allow users that are not specifically approved to interact with a shared document to view or edit them.

**Whiteboard Feature - Final Outcome (as of 11/25/2013)**

Two weeks before the implementation phase was to end, we noticed that the calls to the API of A Web Whiteboard were extremely unreliable. In fact, in average, 1 out of 50 calls to API methods were successful. After careful deliberation, and after weighing both alternatives to the whiteboard implementation, we decided to go for alternative 2.

The main reason for this was the time constraint that comes with only having two weeks to start an implementation from scratch. We believe that 2 weeks is not enough time to learn a new whiteboard system, integrate into the system and test. Moreover, nothing prevents this new system from failing as well.

For that reason, alternative 2 was taken. This requires creating several methods in PHP and JavaScript, of creating upload/view buttons with HTML and CSS, and of using AJAX calls and SQL in order to retrieve URLs of images uploaded into the system. Also, it requires file upload functionality with PHP.

**Blocking Feature - Final Outcome (as of 11/25/2013)**

As described in the whiteboard feature section, given the fact that API calls became unreliable, an alternative to the whiteboard had to be developed. For that reason, resources were directed towards the new implementation of the whiteboard (a major feature), instead of to the blocking feature (a minor feature). At this point, the blocking feature was scraped from the project in order to develop, test and document the alternate solution to the original real-time whiteboard.

**Screen Share Feature**

After multiple testing and debugging of the easyRTC screen sharing tool, we have decided to go ahead and focus our attentions onto the ScreenLeap service. ScreenLeap seems to be a better alternative since all the media streaming takes place between the service provider and the user, there is no need to have a powerful environment that can handle all the bandwidth necessary. By using ScreenLeap we will be able to future proof our system for multiple users and concurrent interviews.

**SMS Feature**

Through various discussion and careful deliberation we have made the final decision between clockwork API and the Twilio. The Twilio service has a better infrastructure and plenty of documentation; additionally we have found the customer service to be unmatchable. The benefits of a 24/7 customer service team can prove to be of great use for the development of this functionality.

**Environment Software Suite Analysis**

After careful analysis of all alternatives, we have ascertained that Alternative #2 (our current implementation) presents a more cohesive and well-supported solution than Alternative #3. It suffers from scalability problems given the nature of it being a closed-source, proprietary, and expensive software solution.

On the other hand, while Alternative #3 does not suffer from these problems, given the fact that it is open-sourced (like our current implementation), it does not present any additional benefits. Moreover, it represents a technical challenge to our developers given the fact they are not familiar with the stack, incurring in higher costs, deviations from the schedule, and an overall less efficient system.

Please refer to Appendix 2 for a feasibility matrix, with each alternative scored by different aspects. As seen in the feasibility matrix, for the environment suite, our original alternative had the highest score out of all three alternatives.

Moreover, our developers have a strong familiarity with the LAMP stack, which will allow for the efficient coding of solutions. Given its open-source nature, our familiarity with it, and high score in our feasibility matrix, we decided to use the LAMP stack for our project.

# Project Plan

The project plan chapter introduces VJF 2.0 from a project management perspective. Firstly, the project organization is described, with the roles for each member listed. After that, milestones, tasks, and deliverables will be listed. Finally, a cost estimate for whole project is presented in terms of a feasibility matrix.

## 3.1 Project Organization

For this project, each of three members will be in charge of adding at least two new pieces of functionality to the system.

Luis Benjumea will be responsible for integrating a collaborative text editor that will be available during the interview. Also, he will be working with the API integration of the Senior Project website.

Jorge Fernandez will be responsible for screen-sharing functionality to be used during interview, allowing students and employers to share screens during interview. Also, he will be handling the integration of an SMS/email notification system to remind students and employers of upcoming interviews.

Luis Irizarry will be responsible for integrating a virtual whiteboard into the interview process. Also, he will be working on an image-sharing feature which allows users to dynamically share images during a live interview.

### 3.1.1 Project Personnel

Below is a table which represents the roles of each of the members:

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Primary Task** | **General Task** |
| Jorge Fernandez | Front End Developer  Tester  Timekeeper | Additional shared tasks |
| Luis Irizarry | Team Leader  Developer  Database Manager | Additional shared tasks |
| Luis Benjumea | Back End Developer  Configuration Manager  Document Editor | Additional shared tasks |

### 3.1.2 Hardware and Software Resources

**Hardware**

In order to start developing the project, our team will need computers with at least the following

specifications:

**- Processing Power:** Pentium IV 2.0 GHz processor or better

**- RAM Memory:** 1GB 133MHz SDRAM

**- Available space on hard drive:** 5GB

**Other Devices**

- **Input devices**

a) Standard wired/wireless K120 keyboard

b) Standard wired/wireless trackball/optical mouse

- **Output devices**

a) Standard VGA/DVI/HDMI monitor display

**Software**

**1) Google Chrome 30.8:** last, most updated version of the Chrome browser that will be used to test Virtual Job Fair

**2) StarUML:** UML software platform that will be used to create diagrams for the document

**3) Yii Framework 1.1.14:** last, most updated version of Yii Framework, an MVC-based, PHP framework used for development. It will be used to develop the front-end and back-end of Virtual Job Fair

**4) phpMyAdmin:** a DBMS (integrated into Yii) that will be used to manually manipulate the database when necessary

**5) Adobe Dreamweaver (IDE):** software platform for front-end design that will be used to create backbone of the user interface

**6)** **VMWare:** virtual machine software that will be used for the deployment of the software

## 3.2 Identification of Tasks, Milestones and Deliverables

Below is a list of all different tasks, milestones, and deliverables for the project:

**Tasks**

Requirement Elicitation

Requirement Analysis

Framework Setup

Deliverable 1 (Feasibility Study and Project Plan)

Database Design

Database Implementation

System Design

Deliverable 2 (Requirements Document)

Object Design

Deliverable 3 (Design Document)

System Implementation

Test Case Design

Test Case Implementation

Integration and System Testing

Deliverable 4 (Final Deliverable)

Documentation

Project Deployment

**Milestones**

Analysis Milestone

Design Milestone

Testing Milestone

Documentation Milestone

**Deliverables**

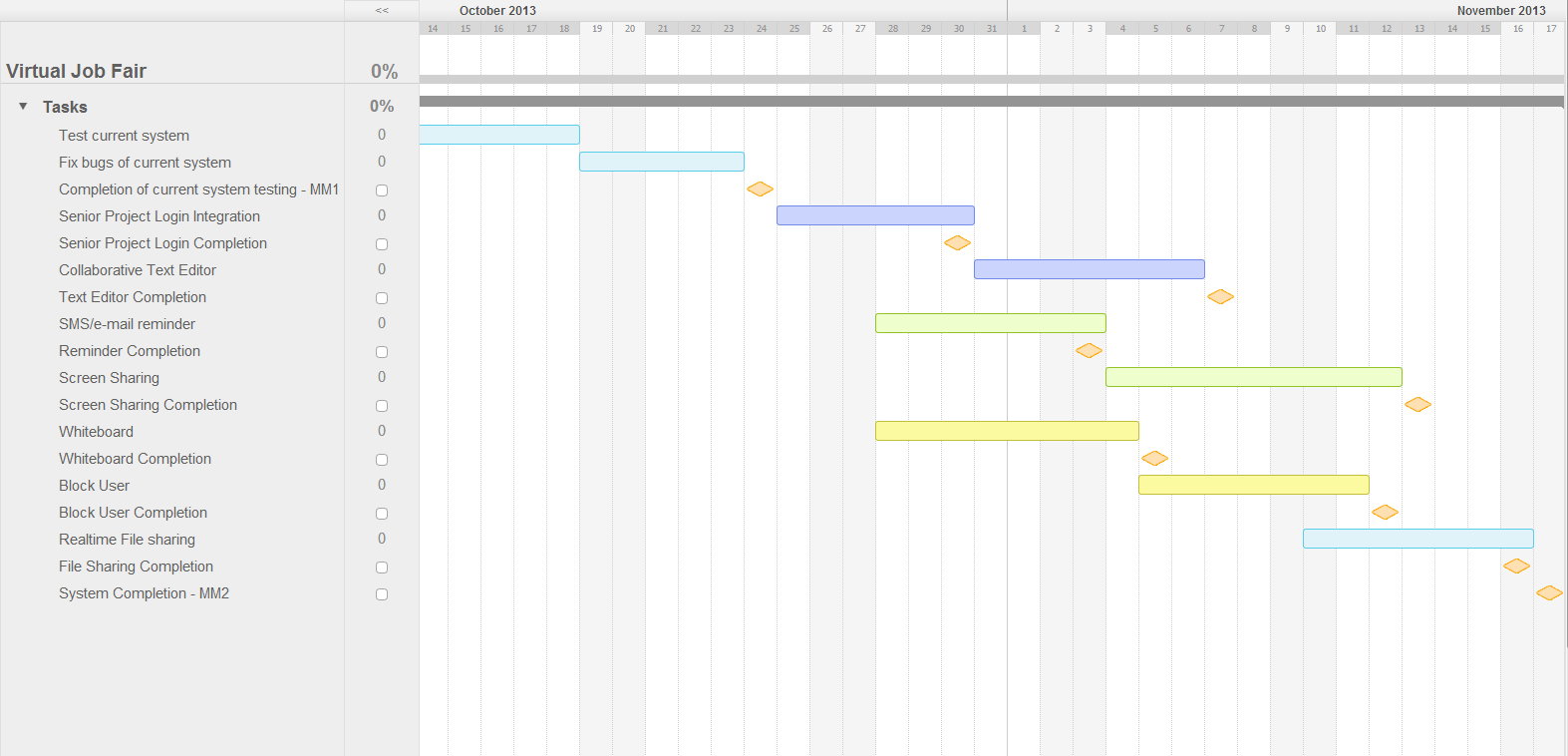
- **Deliverable #1 (Feasibility Study & Plan):** September 9th, 2013

**- Deliverable #2 (Requirements Document):** September 23rd, 2013

**- Deliverable #3 (Design Document):** October 7th, 2013

**- Deliverable #4 (Final Deliverable):** December 9th, 2013

The following GANTT chart contains the project schedule for the semester:



## 3.3 Cost of the Project

The following feasibility matrix represents an estimate of the items and labor required for the project. These estimated costs are accurate as of Monday, September 23rd, 2013.

|  |  |
| --- | --- |
| **Item** | **Item Cost** |
| Yii Framework | $0.00 |
| Twitter Bootstrap | $0.00 |
| Easy RTC | $0.00 |
| Shared Whiteboard | $120.00 |
| Hardware | $1200.00 |
| Development | $0.00 |
| Testing | $0.00 |
| Contingencies | $260.00 (~20% of total cost) |
|  | **Total:**$1570.00 |

# System Requirements

The proposed system is Virtual Job Fair 2.0, which would expand upon the already-implemented Virtual Job Fair. The main idea behind Virtual Job Fair 2.0 is to add new dynamic functionality to the interview process in order to make it more dynamic. This chapter will introduce the functional and non-functional requirements of the system and the requirements analysis phase of the system.

3.1 Functional and Nonfunctional Requirements

**Current System’s Functional Requirements**

The system shall…

· **Allow students and employers to register**

- **Usability**: The register form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 2 seconds.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to view respective profiles**

- **Usability**: Data displayed in profiles is easy to follow. Students are only able to see their own profile and the employer ones. Employers can see all student profiles.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to edit their basic profile information**

- **Usability**: The edit form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to take part in a video interview**

- **Usability**: Starting a video interview is simple and understandable.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 3 seconds when connecting.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to upload a resume and video resume**

- **Usability**: The upload form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1-5 seconds, depending on the file size.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to upload an image for their profile**

- **Usability**: The upload form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to associate skills to their profile**

- **Usability**: The ability to add skills to a profile is simple and understandable. It can be done by using LinkedIn connect or adding them manually.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within one 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to add and delete education information**

- **Usability**: The corresponding form is easy to complete and follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to add and delete experience information**

- **Usability**: The corresponding form is easy to complete and follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to integrate with their LinkedIn account to provide education and experience information (security)**

- **Usability**: The connection with LinkedIn should be easy to follow. Users will enter their LinkedIn credentials and get appropriate data that the user allowed.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to apply to open job postings and provide a cover letter**

- **Usability**: Students are presented with a user-friendly interface that is easy to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to reply to an employer’s message**

- **Usability**: Students are presented with a clear and simple interface to send messages.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to search for jobs based on skills**

- **Usability**: The search form is easy to follow and complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to post jobs**

- **Usability**: The post job form is easy to understand and complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to close a job posting**

- **Usability**: The closing of a post is easy to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to associate skills to a job posting**

- **Usability**: the addition of skills to a post is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second when adding each skill.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to search for students based on skills**

- **Usability**: The search form is simple to submit.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to view student profiles**

- **Usability**: The view of a student profile is easy to understand.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within one 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to send messages to students**

- **Usability**: Employers are presented with a clear and simple interface to send messages.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to give students a “virtual handshake” to show interest in the student**

- **Usability**: The virtual handshake form is easy complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow an administrator to disable an account (security)**

- **Usability**: Disabling a user is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow an administrator to close a job posting**

- **Usability**: Closing a job post is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow an administrator to validate an employer registration (security)**

- **Usability**: The validation of an employer is done by one click and is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Require a username and password to log into the system (security)**

- **Usability**: This is required for a user to log in. Form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Encrypt the user password before storing into the database (security)**

- **Usability**: Storing user password in a secure way without user intervention.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Require login before viewing user profiles (security)**

- **Usability**: Security measure for system. Interface is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

**New System’s Functional Requirements**

**· Allow users to create a new shared document.**

- **Usability:** The document creation interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to open a document.**

- **Usability:** The document open interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to share a document.**

- **Usability:** The invitation should be transparent to the users upon document creation. The invitation interface on new documents should be easy to use.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to delete a document.**

- **Usability:** The document deletion interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 second.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to import a document.**

- **Usability:** The import document interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to export a document.**

- **Usability:** The export document interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to rename document.**

- **Usability:** The document rename interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to save a shared document.**

- **Usability:** The document saving interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Maintain access boundaries between non-collaborating temporary accounts (security).**

- **Usability:** Temporary accounts not related to a collaborating session shall have no access to that session’s documents.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 second.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow FIU Computer Science Seniors to login using their FIU SCIS credentials**

- **Usability:** The system should provide an easy and integrated login process for FIU SCIS Seniors using the school UNIX account.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 second.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

· **Allow users to share their screens**

- **Usability**: User should be able to share their screens solely with mouse clicks

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should start streaming content in under 3 seconds

- **Supportability**: Screen sharing page should be supported by Google Chrome versions 29 and up.

· **Allow users to view shared screens**

- **Usability**: User should be able to watch a shared screen with single mouse click

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should start displaying the screen in under 2 seconds

- **Supportability**: Screen sharing page should be supported by Google Chrome versions 29 and up.

· **Allow users to stop sharing their screens**

- **Usability**: User should be able to stop sharing their screens with a single mouse click

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should stop the stream in under 1 second.

- **Supportability**: Screen sharing page should be supported by Google Chrome versions 29 and up.

· **Allow employers to send SMS to students**

- **Usability**: Sending an SMS should not take more than 15 seconds for a novice user.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The SMS should be sent in under 2 seconds after user presses send

- **Supportability**: Sending SMS page should be supported by IE, Firefox, Chrome and Safari.

· **Allow users to receive automatic email and SMS reminders**

- **Usability**: Not applicable

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: Users should receive a reminder 30 minutes before scheduled interview

- **Supportability**: Not applicable

· **Allow users to confirm their phone numbers**

- **Usability**: Confirming a phone number should take less than 30 seconds for inexperienced users

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: Sending authentication code and validating it should take less than 2 seconds respectively.

- **Supportability**: Confirm phone number page should be supported by IE, Firefox, Chrome and Safari

**· Allow students and employers to start using the whiteboard functionality**

**- Usability:** Starting the whiteboard is a matter of clicking a button while interviewing, making it a fairly easy process to do

**- Reliability:** The whiteboard tool should be available 90% of the time, since not every major/employer will use it

**- Performance:** The whiteboard should be started within 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to upload an image to share during an interview**

**- Usability:** if an image has been selected for uploading, actually uploading it to the system is as simple as click the “Submit Drawing” button, making it simple to use

**- Reliability:** Uploading images should work 99% of the time, since it is a critical function of the image-sharing subsystem

**- Performance:** Depending on its size, the maximum time it should take to upload an image should be 10 seconds

**- Supportability:** The system be supported by Chrome version 29 or above

· **Allow students to view images uploaded by the other party in an interview**

**- Usability:** viewing an image which has been uploaded already relies simply on clicking a “View Drawing” button, so it is fairly simple to use

**- Reliability:** as uploading an image, viewing an image should be working 99% of the time, since it is a critical function of the image-sharing subsystem

**- Performance:** depending on the size of the image, the maximum time it should take to view an uploaded image should be 10 seconds

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers show or restore a whiteboard session**

**- Usability:** starting a whiteboard session involves only clicking the “Whiteboard” button below the video chat section of the interview page, making it a simple process

**- Reliability:** the whiteboard should be available 99% of the time, since it is a main feature of the system

**- Performance:** displaying the whiteboard should take 1 second, since a default whiteboard is not preloaded with any information

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to select an image to upload to the server for sharing purposes**

**- Usability:** selecting an image requires basic experience with a browsing dialog, which should be fairly common knowledge, making it a simple functionality

**- Reliability:** the ability to select an image is critical to the image-sharing feature, so it should be available 99% of the time

**- Performance:** since selecting an image does not depend on image size, this functionality should take less than 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to draw using the whiteboard**

**- Usability:** drawing on the whiteboard simply requires to drag the mouse through the whiteboard’s area, which makes it a simple process

**- Reliability:** without being able to draw in the whiteboard, the whiteboard serves no purpose, so drawing in the whiteboard should be available 99.99% of the time

**- Performance:** since the whiteboard is self-contained and does not need any API calls, the drawing should take less than half a second

**- Supportability:** The system be supported by Chrome version 29 or above

**Allow students and employers to change the color of the drawing pencil tool**

**- Usability:** changing the color is a matter of clicking on the new color desired, so it is fairly simple to do

**- Reliability:** changing the color of the drawing tool is not of the utmost importance, so it is expected to work 90% of the time

**- Performance:** since the whiteboard is self-contained and does not need any API calls, changing the color should take less than half a second

**- Supportability:** The system be supported by Chrome version 29 or above

**Allow students and employers to type text into the whiteboard**

**- Usability:** typing text into the whiteboard requires clicking a button and typing text, so it is fairly simple to use

**- Reliability:** since typing text is not the main whiteboard feature, it is expected to work 90% of the time

**- Performance:** the text should appear in the whiteboard in less than three seconds

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to clear the drawings of the whiteboard**

**- Usability:** Clearing the contents is a matter of clicking a single button, making it a fairly easy process to conduct

**- Reliability:** It should be available 99% of the time that the whiteboard is being used, given that new drawings might be required

**- Performance:** Whiteboard screen should be cleared out within 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**Allow students and employers to partially erase drawings from the whiteboard**

**- Usability:** Partially erasing from the drawing is a matter of clicking a single button, drawing the cursor and releasing the click, making it a fairly easy process to conduct

**- Reliability:** It should be available 99% of the time that the whiteboard is being used, given that a user might make a mistake while drawing or that new drawings might be required

**- Performance:** selected sections should be erased within 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

4.2 Requirements Analysis

Analysis models – contains the complete functional specification and is mainly for the designers and programmers. This section describes the diagrams in the Appendices B - D and validates the models against the use cases.

### 4.2.1 Use case model

The use case diagram describes the overall view and functionality that we will add to the system. The proposed functionality deals with the two type of users displayed in the diagram. Both student and employers are the targets of these functional requirements since the majority of use cases dealing with the administrator user type was developed by the previous team working on Virtual Job Fair V1.

### 4.2.2 Static model

**Object Diagrams Analysis**

1. **Whiteboard**

An object from the Whiteboard class is instantiated when a user first requests to use the whiteboard. The instantiated Whiteboard object has 4 properties. autoJoin is a feature that deals with automatically joining a whiteboard. “toolbar” determines whether the toolbar will be shown on the whiteboard. “smallIcons” decides whether small icons will be forced on the whiteboard. Finally, “orientation” determines the view of the whiteboard (acceptable values are “portrait”, “landscape” and “auto”.

**Document**

An object which represents the Document class gets instantiated upon initiation of the Collaborative Text Editor feature, more specifically, upon the request of any of the features that involve document manipulation. The Document Object describes 10 properties:

- id: unique identifier.

- active\_status: represents if the current document is in an active state.

- document\_id: represents a unique document, it's an internal unique identifier that represents a file.

- local\_user\_id: contains the unique user id of the user who creates a local document.

- remote\_user\_id: contains the unique user id of the user who collaborates with the document remotely.

- owner\_id: which represents which of the users collaborating in the document owns it.

- document\_path: the document path on the filesystem.

- document\_name: represents the document identifier from the point of the user, not the system.

- owner\_url: the url for the Zoho API that accesses and displays a specific document.

-viewer\_url: the url for the Zoho API that permits another user to join a specific document.

**SMS**

An object from the SMS class is instantiated when a SMS message will be sent to a particular user. The object has 6 properties which are useful for any interaction involving a text message. The id property specifies the unique identifier of this text message, the receiver and sender properties specify the identification numbers of the parties involved, the message field contains the content of the SMS, then there is the date field which stores the current date and finally subject which holds the topic of the message.

The other two objects, BasicInfo and VideoInterview, were already explained by the old group.

**Class Diagram Analysis**

The Class Diagram in Appendix C depicts the additional classes that will be added to the original system. It portrays the proposed changes to the scheme, where we build on the implementation of the previous team. The new proposed classes are colored green in order to emphasize the difference between the new and old implementations. In addition, the Diagram follows UML notation in order to facilitate the developers as well as adhere to industry standards.

### 4.2.3 Dynamic model

In our sequence diagrams, actors can be either students, employers, or any. In all of our use cases, the actors communicate with an object from the view component. Also, some sequence diagrams require communication with our database.

# 5. System Design (i.e., overall system design)

This chapter gives an insight into the architectural pattern(s) used to build the system. Virtual Job Fair was subdivided into subsystems, each one with a specific functionality that adds richness to the interview process. In this chapter, an overview of the system design is introduced. Then, the decomposition of the system into subsystems is explained. Moreover, hardware and software mapping and persistent data management aspects of the project are discussed. Finally, the security and privacy issues of the system are introduced.

The new system shall…

Allow users to create a new shared document.

Allow students and employers to start using the whiteboard functionality

Allow students and employers to upload an image to share during an interview

Allow students to view images uploaded by the other party in an interview

Allow students and employers show or restore a whiteboard session

Allow students and employers to select an image to upload to the server for sharing purposes

Allow students and employers to draw using the whiteboard

Allow students and employers to change the color of the drawing pencil tool

Allow students and employers to type text into the whiteboard

Allow students and employers to clear the drawings of the whiteboard

Allow students and employers to partially erase drawings from the whiteboard

Allow users to create a new document.

Allow users to invite another user to a shared document.

Allow users to delete a shared document.

Allow users to import a document.

Allow users to export a document.

Allow users to rename a document.

Allow users to save a shared document.

Maintain access boundaries between non-collaborating accounts.

Allow FIU Computer Science Seniors to login using their FIU SCIS credentials.

Allow students and employers to share their screens

Allow student and employers to view each other’s screen.

Allows employers to contact students through SMS

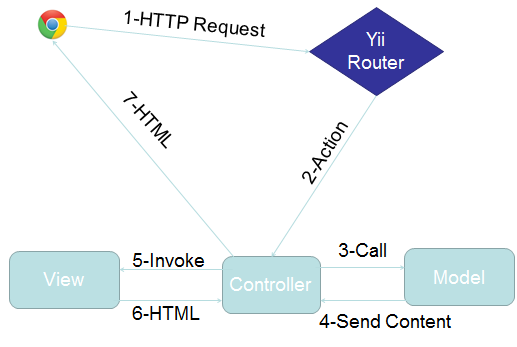
Allows students and employers to receive SMS notifications

Allow student and employers to confirm their phone numbers through SMS validation

## **5.1 Overview**

The previous team implemented Virtual Job Fair using the Model-View-Controller architecture. Our team is adding functionality to the system, meaning that the Model-View-Controller architecture will be maintained, with models, views and controllers added to account for the new functionality. Below is their description of the architecture of the system:

 “The Architectural pattern used to built the system was the popular Model-View-Controller. This is the architecture implemented by the Yii Framework which was used by the team of developers. The following diagram depicts the architecture:”



## 5.2 Subsystem Decomposition

The previous group had a total of 5 subsystems, while we will be adding an extra 4. Below are the subsystems that the previous group used.

**Video Interview Subsystem**

The Video Interview Subsystem allows users to have a virtual interview. This is possible thanks to Web-RTC technology. Web-RTC is an open source project that allows web browsers to communicate directly with each other with the aid of Java Scrip API calls and HTML5. This new technology makes the communication between internet users easier than traditional methods. Thanks to Web-RTC one can share video feed with other users without the need for media servers or plug-ins.

The synchronization of video interviews is handled in a table in the database. This is very important because we need to make sure only users scheduled to have an interview can be in the interview page. This is done by using a session key which is unique for each video interview scheduled. Users that arrive at the interview page would do so via a link which has many parameters such the session key. Once both users arrive at the interview page, the application will check for the session key parameter and match them, and only those users that have matching session keys would be able to connect to each other.

The uses cases related to this subsystem are:

·         VJF-0020 Start Video Interview

·         VJF-0021 Accept Interview

·         VJF-0030 Schedule Video Interview

**Messaging Subsystem**

The messaging subsystem is a very important one because it allows users to keep in contact and communicate with each other right on the system. It is very similar to a traditional inbox, only that it is internal to the system, similar to LinkedIn’s messaging. The messaging subsystem uses database tables to store and retrieve messages sent between users. It allow employers to message students and in doing so open up a line of communication with them; i.e., a student is able to message an employer only after the employer has initiated the communication with that particular student. Storing and retrieving messages efficiently is imperative to a successful messaging system. The messaging subsystem uses AJAX to rapidly access and store data; allowing users to interact faster with the system without having to wait for server calls.

The use cases related to this subsystem are:

·         VJF-0022 Reply to Message

·         VJF-0023 Send Message

**Notification Subsystem**

The notification subsystem is a very important part of the system. This is what allows users to stay up to date with the latest system interactions. The notification subsystem alerts users of any interaction by other users that might implicate them, such as a video interview been schedule for a user, or a new job post that matches a user’s skills.

The notification subsystem relies on the database structure to be able to efficiently keep the users inform. Due to the relationships between tables in the database the notification subsystem can easily detect what notification belongs to what user. Also it is important to sort each notification by category and level of importance. The notification subsystem achieves this by storing different types of notifications in the database and mapping them to their respective categories.

The uses cases related to this subsystem are:

·         VJF-0021 Accept Interview

·         VJF-0025 Post Job

·         VJF-001 Registration

·         VJF-0033 Apply to Job

·         VJF-0034 Read notification

**Profile creation subsystem**

Being able to create a good profile fast and efficiently is very important. The profile creation subsystem takes care of this by allowing students to import profile information from third party websites such as LinkedIn. This ensures integrity of the data in students’ profiles, and makes it very easy for students to create their profiles.

This is possible by using API calls to LinkedIn and retrieving the data from LinkedIn users. As it is to expect, the user must grant permission to do this by providing his/her login credentials which are handled by the LinkedIn API.

The use cases related to this subsystem are:

·         VJF-001 Registration

·         VJF-0019 Integrate LinkedIn

·         VJF-003 Edit Basic Info

·         VJF-004 Verify Email

·         VJF-008 Edit Picture

·         VJF-009 Upload Resume

·         VJF-0011 Add Education

·         VJF-0012 Delete Education

·         VJF-0013 Add Experience

·         VJF-0014 Delete Experience

**Student job match subsystem**

Making the right connection is what this web application is all about. Therefore, an efficient algorithm to match students to job openings is very important. The student job match subsystem takes care of matching students with the required skills to job post, making the job of the recruiters easier, as it shrinks the search to only the most qualified individuals for the job.

The student job match subsystem relies on the relationships between the data in the job table. By matching job skills to students skills listed on their profile the algorithm can effectively narrow down the search to only those individuals who possess those skills.

The uses cases related to this subsystem are:

·         VJF-0016 Add Skill

·         VJF-0017 Delete skill

·         VJF-0018 Change skills Order

·         VJF-0026 Virtual Handshake

Below, are the 4 additional subsystems that we will be using to decompose the system further.

**Collaborative Editor Subsystem**

The collaborative text editor subsystem allows users to edit a document in realtime with one another. The Zoho Docs API will provide the Features, Methods, Calls, and Functionality needed to implement the Collaborative Text Editor Subsystem. Zoho Docs is implemented as a web based application that requires no additional plugins or addons to be installed by the end users. The Zoho Docs API provides a clean, refreshing and intuitive user interface which facilitates usage of the system by end users. Zoho Documents is built upon AJAX technology, this provides the users with the fast, real-time, user experience required of the Subsystem.

The use cases related to this system are:

VJF-048 Create a new Document

VJF-049 Share active Document

VJF-050 Delete shared Document

VJF-051 Import Document

VJF-052 Rename Document

VJF-053 Save Document

VJF-054 Open Document

VJF-055 Export Document

**Screen Share subsystem**

Additionally the video screen Share subsystem allows users to share their screens with one another. This functionality is facilitated by the Screen Leap API. Screen Leap is a service that provides screen sharing functionality without the need for any of the users to install any specific software all that is required to function is the latest version of Java for the broadcaster.  Screen Leap is able to function perfectly with the system by providing different API calls that allow for easy sharing of information. By eliminating the need for the user to download extra software we are able to facilitate and simplify the usage of the Virtual Job Fair website.

VJF-042 Share Screen

VJF-043 View Screen Share

VJF-044 End Screen Sharing

**SMS message subsystem**

One of the most important features of the website is the ability to send messages to students as well as allow the users to receiver reminders about upcoming interviews directly in their SMS enabled phones. The SMS messaging subsystem is designed to allow this kind of functionality which can keep the users connected and active with the system.

All of this is possible by using the Twilio cloud communications service. Twilio is a company that provides services such as SMS sending and receiving, speech and text recognition, conference calling etc.  The most compelling feature is that an extensive and well documented API is provided to developers, this will be necessary in order to create a system that allows for sending text messages to users, as well as validating their identities.

VJF-045 Send SMS to student

VJF-046 Send interview reminder

VJF-047 Confirm phone number

**Whiteboard subsystem**

The whiteboard subsystem will allow users to draw on a whiteboard during a live interview. The whiteboard includes drawing, writing, erasing, clearing and picture-saving functionality, which makes it an ideal tool to share images during a live interview, to jot down notes or simply to brainstorm. The use cases related to the whiteboard are:

VJF-060 Show or Restore Whiteboard

VJF-062 Draw With Pencil

VJF-063 Change Drawing Tool Pencil

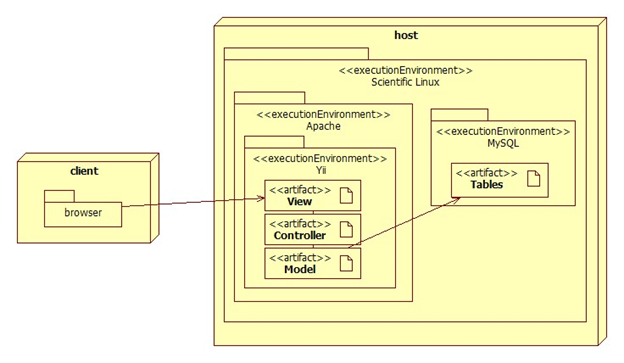
VJF-064 Type Text Into Whiteboard

VJF-065 Clear contents of whiteboard

VJF-066 Erase From Whiteboard

## **5.3 Hardware and Software Mapping**

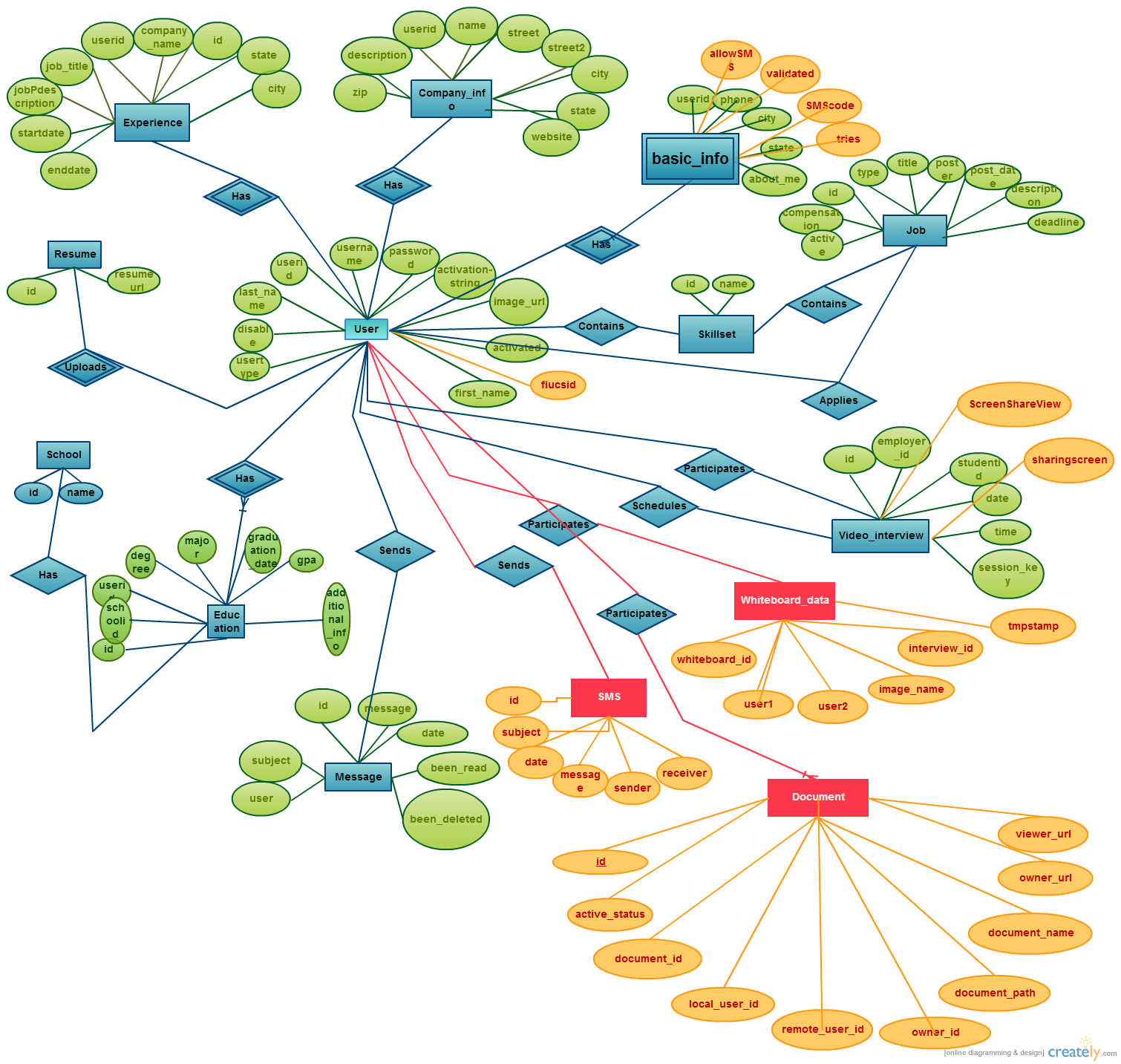
The previous team developed a deployment diagram which indicates the mapping of hardware and software. For our Virtual Job Fair 2.0, we are still using the Model-View-Controller architectural pattern. Moreover, our team requested a virtual machine running on Scientific Linux with an Apache server and the Yii framework on the FIU SCIS network, identical to the set up used by the previous team. For that reason, the same deployment diagram as before will be used. Therefore, the description below was that done by the previous team, and remains valid for Virtual Job Fair 2.0.



The deployment diagram shown above represents the hardware and software mapping in the Virtual Job Fair system.  The main components of the system are the Apache and MySQL environment hosted on a Linux operating system.  The Yii framework environment is using apache to execute, and contains our various artifacts used in development (Model, View, and Controller).  The models are mapped to tables in the MySQL environment set up on the same machine.  The browser on the client’s machine communicates with the server using HTTP.

## 5.4 Persistent Data Management

The previous group completed an ER diagram. Since we are adding new tables and columns into existing tables and database, respectively, below is the updated version of the ER diagram of the original group, which includes our new data, which is identified by a yellow background and dark red text.



Below is the data dictionary for the old group, to which we added several definitions, starting with “Collaborative Editor”:

**application**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **jobid** | int(11) | NO | PRI |  |  |
| **userid** | int(11) | NO | PRI |  |  |
| **application\_date** | varchar(45) | NO |  |  |  |
| **coverletter** | text | YES |  |  |  |

**basic\_info**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **userid** | int(11) | NO | PRI |  |  |
| **phone** | varchar(15) | YES |  |  |  |
| **city** | varchar(45) | YES |  |  |  |
| **state** | varchar(45) | YES |  |  |  |
| **about\_me** | text | YES |  |  |  |
| **hide\_phone** | int(11) | YES |  |  |  |
| **allowSMS** | int(11) | 0 |  |  |  |
| **validated** | int(11) | 0 |  |  |  |
| **smsCode** | int(11) | null |  |  |  |
| **tries** | int(11) | 0 |  |  |  |

**company\_info**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **FK\_userid** | int(11) | NO | PRI |  | auto\_increment |
| **name** | varchar(45) | YES |  |  |  |
| **street** | varchar(45) | YES |  |  |  |
| **street2** | varchar(45) | YES |  |  |  |
| **city** | varchar(45) | YES |  |  |  |
| **state** | varchar(45) | YES |  |  |  |
| **zipcode** | varchar(45) | YES |  |  |  |
| **website** | varchar(45) | YES |  |  |  |
| **description** | text | YES |  |  |  |

**education**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **degree** | varchar(45) | NO |  |  |  |
| **major** | varchar(45) | NO |  |  |  |
| **graduation\_date** | date | NO |  |  |  |
| **FK\_school\_id** | int(11) | YES | MUL |  |  |
| **FK\_user\_id** | int(11) | YES | MUL |  |  |
| **gpa** | float | YES |  |  |  |
| **additional\_info** | text | YES |  |  |  |

**experience**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **FK\_userid** | int(11) | YES | MUL |  |  |
| **company\_name** | varchar(45) | YES |  |  |  |
| **job\_title** | varchar(45) | YES |  |  |  |
| **job\_description** | text | YES |  |  |  |
| **startdate** | datetime | YES |  |  |  |
| **enddate** | datetime | YES |  |  |  |
| **city** | varchar(45) | YES |  |  |  |
| **state** | varchar(45) | YES |  |  |  |

**handshake**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **jobid** | int(11) | YES | MUL |  |  |
| **employerid** | int(11) | NO | MUL |  |  |
| **studentid** | int(11) | NO | MUL |  |  |

**job**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **type** | varchar(45) | NO |  |  |  |
| **title** | varchar(45) | NO |  |  |  |
| **FK\_poster** | int(11) | NO | MUL |  |  |
| **post\_date** | datetime | NO |  |  |  |
| **deadline** | datetime | YES |  |  |  |
| **description** | longtext | NO |  |  |  |
| **compensation** | varchar(45) | YES |  |  |  |
| **other\_requirements** | text | YES |  |  |  |
| **email\_notification** | int(11) | YES |  |  |  |
| **active** | int(11) | YES |  | 1 |  |
| **matches\_found** | int(11) | YES |  |  |  |

**job\_skill\_map**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **jobid** | int(11) | NO | MUL |  |  |
| **skillid** | int(11) | NO | MUL |  |  |
| **level** | varchar(45) | YES |  |  |  |
| **ordering** | int(11) | YES |  |  |  |

**message**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **FK\_receiver** | varchar(45) | NO | MUL |  |  |
| **FK\_sender** | varchar(45) | NO | MUL |  |  |
| **message** | text | YES |  |  |  |
| **date** | datetime | YES |  |  |  |
| **been\_read** | int(11) | YES |  | 0 |  |
| **been\_deleted** | int(11) | NO |  | 0 |  |
| **subject** | varchar(255) | YES |  |  |  |
| **userImage** | varchar(255) | YES |  |  |  |

**notification**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **sender\_id** | int(11) | NO | MUL |  |  |
| **receiver\_id** | int(11) | NO |  |  |  |
| **datetime** | time | NO |  |  |  |
| **been\_read** | int(11) | NO |  | 0 |  |
| **message** | varchar(5000) | YES |  |  |  |
| **link** | varchar(150) | YES |  |  |  |
| **importancy** | int(11) | NO |  | 0 |  |

**resume**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  |  |
| **resume** | varchar(255) | YES |  |  |  |

**school**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **name** | varchar(100) | NO |  |  |  |
| **email\_string** | varchar(45) | YES |  |  |  |

**skillset**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **name** | varchar(45) | NO | UNI |  |  |

**student\_skill\_map**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **userid** | int(11) | YES | MUL |  |  |
| **skillid** | int(11) | YES | MUL |  |  |
| **level** | varchar(45) | YES |  |  |  |
| **ordering** | int(11) | YES |  |  |  |

**user**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **username** | varchar(45) | NO | UNI |  |  |
| **password** | varchar(255) | YES |  |  |  |
| **FK\_usertype** | int(11) | NO | MUL |  |  |
| **email** | varchar(45) | NO | UNI |  |  |
| **registration\_date** | datetime | NO |  |  |  |
| **activation\_string** | varchar(45) | NO |  |  |  |
| **activated** | int(11) | YES |  |  |  |
| **image\_url** | varchar(255) | YES |  |  |  |
| **first\_name** | varchar(45) | NO |  |  |  |
| **last\_name** | varchar(45) | NO |  |  |  |
| **disable** | int(11) | YES |  |  |  |
| **has\_viewed\_profile** | int(11) | YES |  |  |  |
| **linkedinid** | varchar(45) | YES |  |  |  |
| **googleid** | varchar(45) | YES |  |  |  |
| **hide\_email** | int(11) | YES |  |  |  |
| **fiucsid** | varchar(45) | YES |  |  |  |

**usertype**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **type** | varchar(45) | NO |  |  |  |

**video\_interview**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **FK\_employer** | int(11) | NO | MUL |  |  |
| **FK\_student** | int(11) | NO | MUL |  |  |
| **date** | date | NO |  |  |  |
| **time** | time | NO |  |  |  |
| **session\_key** | varchar(45) | NO |  |  |  |
| **notification\_id** | varchar(45) | NO |  |  |  |
| **screenShareview** | varchar(90) | NO |  |  |  |
| **sharingscreen** | int(11) | NO |  |  |  |

**video\_resume**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  |  |
| **video\_path** | varchar(100) | YES | UNI |  |  |

**New System’s Persistent Data**

**Collaborative Editor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **active\_status** | tinyint(1) | NO |  |  |  |
| **document\_id** | varchar(256) | NO |  |  |  |
| **local\_user\_id** | int(11) | NO |  |  |  |
| **remote\_user\_id** | int(11) | NO |  |  |  |
| **owner\_id** | int(11) | NO |  |  |  |
| **document\_path** | varchar(255) | NO | UNI |  |  |
| **document\_name** | varchar(255) | NO | UNI |  |  |
| **owner\_url** | varchar(255) | NO |  |  |  |
| **viewer\_url** | varchar(255) | NO |  |  |  |

**basic\_info**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **userid** | int(11) | NO | PRI |  |  |
| **phone** | varchar(15) | YES |  |  |  |
| **city** | varchar(45) | YES |  |  |  |
| **state** | varchar(45) | YES |  |  |  |
| **about\_me** | text | YES |  |  |  |
| **hide\_phone** | int(11) | YES |  |  |  |
| **allowSMS** | int(11) | YES |  |  |  |
| **SMScode** | int(11) | YES |  |  |  |
| **validated** | int(11) | YES |  |  |  |
| **Tries** | Int(11) | NO |  |  |  |

**video\_interview**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **FK\_employer** | int(11) | NO | MUL |  |  |
| **FK\_student** | int(11) | NO | MUL |  |  |
| **date** | date | NO |  |  |  |
| **time** | time | NO |  |  |  |
| **session\_key** | varchar(45) | NO |  |  |  |
| **notification\_id** | varchar(45) | NO |  |  |  |
| **ScreenShareView** | varchar(45) | YES |  |  |  |
| **Sharing** | int(11) | NO |  |  |  |

**SMS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **id** | int(11) | NO | PRI |  | auto\_increment |
| **receiver** | int(11) | NO |  |  |  |
| **sender** | int(11) | NO |  |  |  |
| **message** | varchar(45) | NO |  |  |  |
| **date** | date | NO |  |  |  |
| **subject** | varchar(45) | YES |  |  |  |

**whiteboard\_sessions**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| **user1** | varchar(20) | NO |  |  |  |
| **user2** | varchar(20) | NO |  |  |  |
| **Whiteboard\_id** | varchar(70) | NO |  |  |  |
| **Interview\_id** | Int | NO | PRI |  |  |
| **Tmpstamp** | TIMESTAMP | NO |  |  |  |

## 5.5 Security/Privacy

The previous team defined security and privacy features for log in, registration, and access to the Yii framework. Moreover,  security and privacy features were added for each of the functionality that was implemented. Both of these are described below:

**Security Features**

·         User password will be hashed in the database.

Upon registration into the system, passwords entered will be hashed right away and will not be saved anywhere on the system.  Upon login, the password entered again will be hashed and the hashed data will be used to query the database.

**·         Administrator will be able to disable users and delete jobs**

An administrative console will be provided to a person to allow basic duties that may be needed in the future.  Due to abuse of the system, it may be necessary to delete jobs or disable users.

· **Yii access control rules**

The Yii framework provides access control with respect to any controller being used.  This access control will reject a subset of users (not logged, students, employers, etc…) from performing certain actions.  For example, users that are not logged in will not have access to profile pages.

**·         Cross-site Scripting Prevention**

The Yii framework takes measures against common web exploitations such as cross-site scripting or MySQL injection.  Using Yii, we can be rest assured that such things should not occur.

**·         Secure registration process**

The registration process is not as simple as most sites, especially for employers.  Administrators will have to verify employers after they register to ensure they are actual employers to ensure the integrity of the system.  Only then will they be able to post jobs and interact with students.

**Privacy**

Students and Employers are distinct user types and therefore have distinct permissions.  It may be necessary to allow employers to do actions that students cannot.  For example, students should not be able to post a job or schedule an interview, which clearly employers should be able to.  Likewise, students will only be able to view an employer’s profile and will not be able to view other students profiles, since it may contain information which should not be shared, such as phone number or email.

Below are these security and privacy features for the new functionality of the system:

**Whiteboard**

For the whiteboard functionality, the URL of the interview page will be hashed using the SHA1 function in PHP. The purpose of this is to make the whiteboard interaction more secure, preventing an intruder from typing in a URL which would give him/her access to the whiteboard session between a student and an employer.

**Collaborative Editor**

Maintain Document Access Boundaries:

Users collaborating on documents will have no access to documents other than the ones that have been shared in a particular session.

Maintain User Limit on Collaboration Sessions:

Users collaborating on documents will only be able to share/collaborate on a document with one other person at a time ( max 2 users collaborating on document )

# Detailed Design

6.1 Overview

**Whiteboard Subsystem**

The whiteboard subsystem handles the interaction of students and employers through the Virtual Whiteboard. Adding to the already implemented video interview functionality Virtual Job Fair, a whiteboard seems like the next logical step. This whiteboard will allow students to make drawings while interviewing, which is ideal for different majors. The whiteboard will function real-time, meaning that there will be a single whiteboard which will be shared by both employers and students.

**Collaborative Editor Subsystem**

The Collaborative Editor Subsystem handles Interactions between users collaborating on a Document. It is meant to be intuitive, responsive and feature rich. The collaborative editor relies on Scientific Linux, ext3 filesystem, PHP and the Yii Framework to provide a robust Document Server and Document Repository Features. It relies on HTML, CSS, Js, and AJAX to provide a light and responsive editor that can be used by the users to collaborate on a document.

**Screen Share subsytem**

The ScreenShare subsystem handles the display and broadcasting of the user’s screens.  When users request a screen Share session, the system send a request to the Screen Leap Api which returns the state of the request as well as other usable information such as a viewer URL. This URL is stored in order to allow the other user to connect and see what is being displayed.  The screenLeap service makes it possible to provide this feature without having any additional services running in the system.

**Sms subsystem**

The sms subsystem allows students and employers alike to receive text message notifications about their upcoming interviews. Additionally, employers will be able to send direct text messages  to students they would like to interview with in a short notice.  The Twilio Service provides a very detailed and rich API that allows for these features to be implemented in a way that allows for reliability and security and most importantly ensuring the privacy of our users.

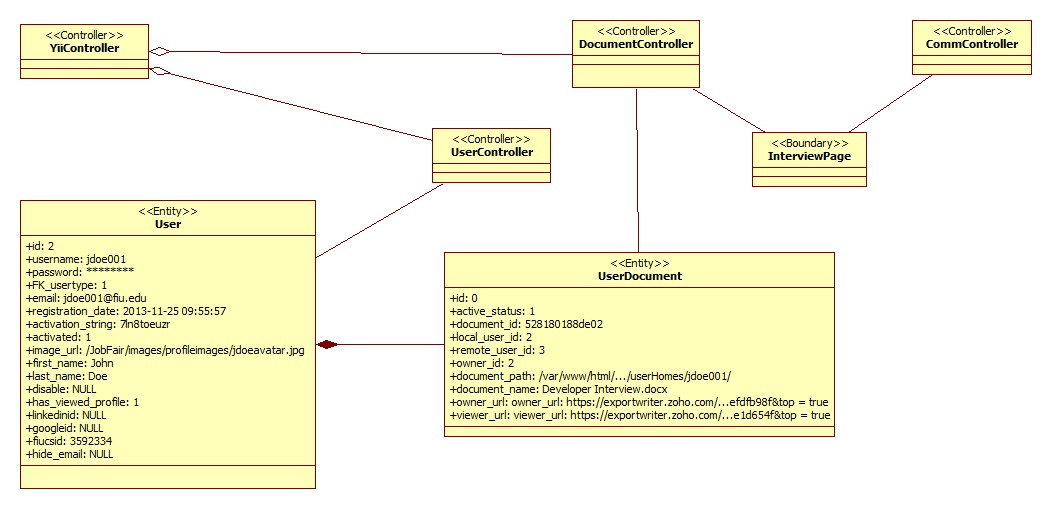
**Design Patterns:**

* **Active Record Pattern:** given that the correct functioning of the project relies on storing and retrieving data from a relational database, the active record pattern makes sense. Moreover, given that the Yii framework is object-oriented, it matches the requirement of the pattern of having a database wrapped in a class.
* **Front Controller Pattern:** the Yii framework is one of the several frameworks which implement the Front Controller Pattern

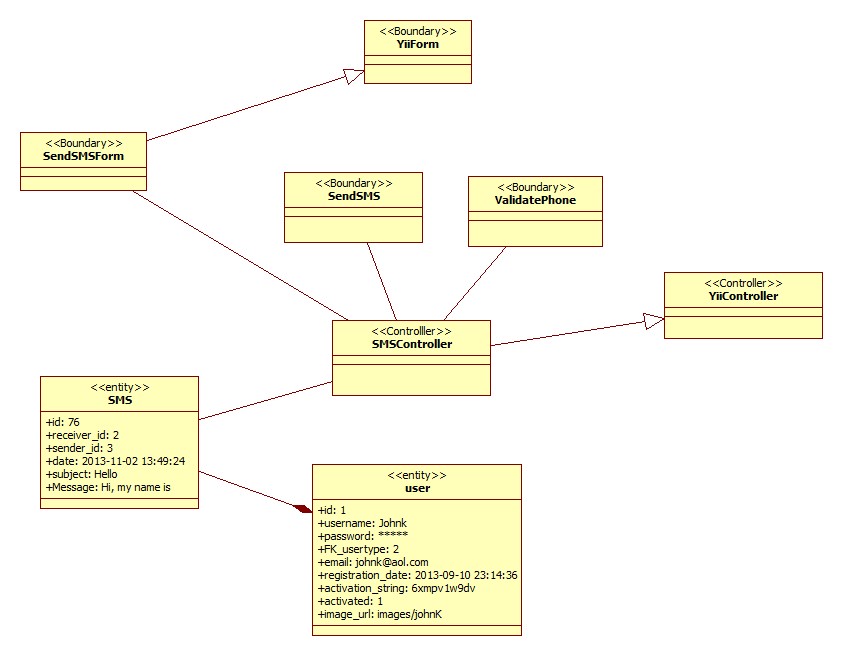
The detailed design chapter introduces the system in terms of subsystems and the relationships among them. Initially, the system is decomposed into subsystem, with each subsystem described in terms of behavior and structure. Then, the static model is introduced in terms of subsystems with descriptions for each. After that, the dynamic model is presented in terms of state machine diagrams with the main control object for each subsystem. Finally, class interfaces and constraints for the main control object in each subsystem are presented.

## 6.2 Static model – detailed description of the structure for each subsystem.

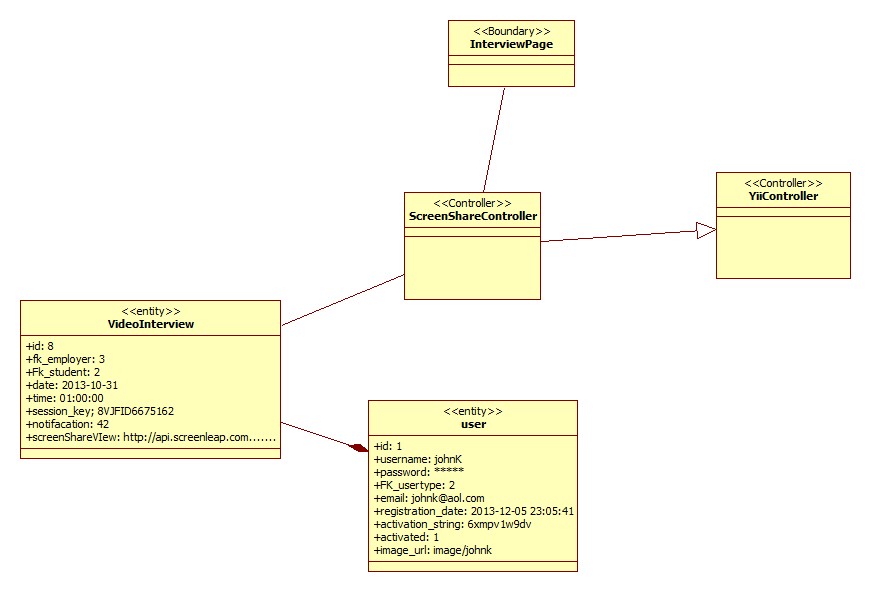
**Collaborative Editor Sub System**



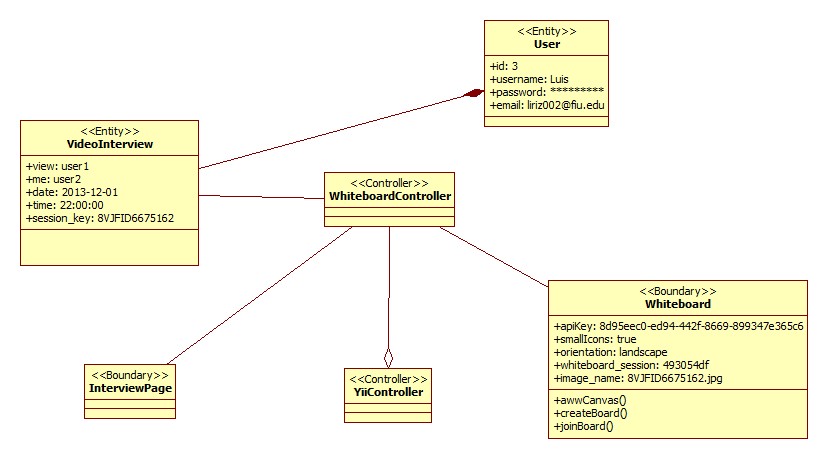
**SMS subsystem**



**Screen Share subsystem**

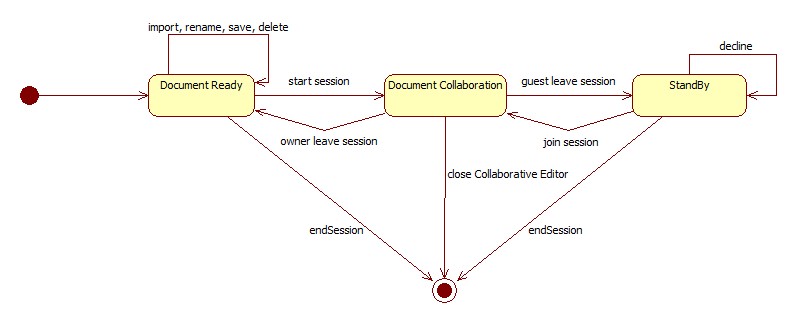


**Whiteboard subsystem**

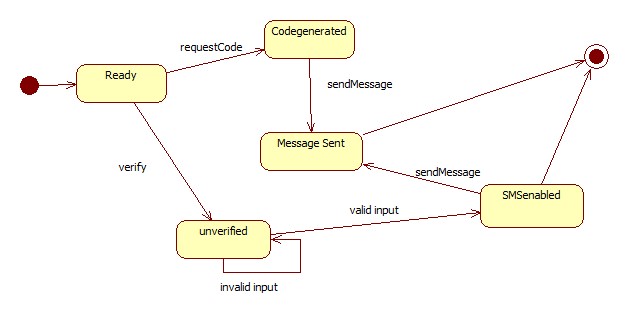


## **6.3   Dynamic model**

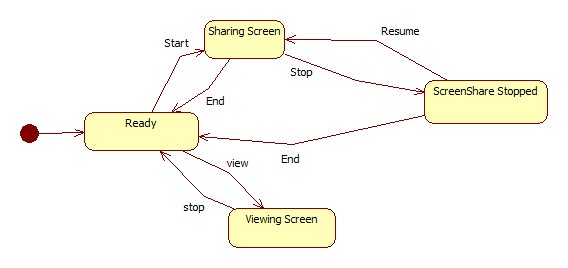
**Collaborative Editor State machine**



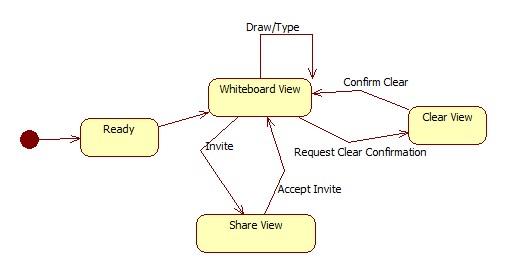
**Sms State machine**



**ScreenShare State machine**



**Whiteboard**



## 6.4 Code Specification

**Whiteboard Subsystem**

Method Signatures

**public function actionCheckWhiteboardExists()**

@precondition: an image has been uploaded server

@postcondition: view receives answer regarding existence of whiteboard

**public function actionStoreWhiteboardSession()**

@precondition: a valid image has been uploaded to the server

@postcondition: the "image\_name" field of the database is updated to reflect the new image

**public function actionCheckDrawingExists()**

@precondition: the user has clickd "View Drawing"

@postcondition: user receives message of whether there is an image for viewing

**public function actionUploadImage()**

@precondition: the usr has clicked on "Submit Drawing"

@postcondition: an image will have been uploaded to the server

**Collaborative Editor Subsystem**

Class:DocumentController

**Method Signatures:**

@dbg

@precondition: user has a message to debug

@postcondition: user has debugged a message

**private function dbg($msg)**

@create new editor

@precondition: interview has been scheduled

@precondition: user is at interview portal

@postcondition: user has a new editor

**private function createNewEditor($uN, $rN, $dI, $mD, $zDI)**

@open existing editor

@precondition: interview has been scheduled

@precondition: user is at interview portal

@postcondition: user has a opened an existing document

**private function openExistingEditor($uN, $rN, $dI, $mD, $zDI)**

@prepare home folder

@precondition: a new document will be created

@precondition: user is at interview portal

@postcondition: user has a folder to save documents

**private function prepareHomeFolder($username, $documentId)**

@save db

@precondition: user has data to save

@precondition: user is at interview portal

@postcondition: data is saved

**private function saveDb($recordArray)**

@mark documents innactive

@precondition: user is at interview portal

@precondition: user has selected a new document

@postcondition: user has a new document

**private function markDocumentsInactive(**

@create document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@postcondition: user has a new document

**public function actionCreateDocument($rU)**

@import document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user has at least one document to import

@postcondition: user has a document imported

**public function actionImportDocument()**

@rename document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user has at least one document to rename

@postcondition: user has at a document with a different name

**public function actionRenameDocument($document, $newName)**

@save document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user has at least one document to save

@postcondition: user has saved a document

**public function actionSaveDocument()**

@delete document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user has at least one document to delete

@postcondition: user no longer has the document

**public function actionDeleteDocument($document)**

@share document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user has at least one document to share

@postcondition: user shared their document

**public function actionShareDocument($documentUrl, $documentName, $remoteUserId)**

@list document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user wants to manage documents

@postcondition: user can view his/her documents

**public function actionListDocument()**

@export document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user has at least a document to export

@postcondition: user has downloaded their document

**public function actionExportDocument()**

@open document

@precondition: interview has been scheduled

@precondition: user is at interview portal

@precondition: user wants to open an existing document

@postcondition: user has his document open

**public function actionOpenDocument($rU, $dI)**

**SMS Subsystem**

Class:SMScontroller

**Method Signatures:**

@send reminder to user and employers via SMS

@precondition: interview has been scheduled

@postcondition: users receive an SMS reminder

**public function actionsendReminder()**

@Employer sends SMS to student

@precondition: user is logged in, user has requested to send an SMS

@postcondition: user receives information about message action

**public function actionsendsms()**

@request phone number validation

@precondition: user has entered a phone number, user clicks on validate

@postcondition: Code is sent to users phone

**public function actionSendCode()**

@verify phone number code

@precondition: user enters validation code

@postcondition: phone is activated or validated accordingly

**public function actionValidation()**

@verify sms contents

@precondition: user has validated phone

@postcondition: SMS contents are valid to be sent

**public function actionverify()**

@specifies access rules

**public function accessRules**

**public function actionGetAutoComplete()**

@return auto-complete values to user

@postcondition: SMS usernames are returned to user

**ScreenShare Subsystem:**

Class:ScreenSharecontroller

**Method Signatures:**

@share screen

@precondition: interview has been scheduled

@postcondition: users shares his screen

**public function actionGetScreenLeap($interview)**

@get the url belonging to the interview

@precondition: user is sharing a screen

@postcondition: user receives screenshare url

**public function actionGetviewerUrl()**

@stop the screen Share

@precondition: user is sharing

@postcondition: Screen share is stopped and sharing field is cleared

**public function actionSetstop()**

# 7 System Validation

The system validation chapter introduces testing to make sure that the project works as was intended. For this specific phase of the project, we performed system testing and subsystem testing, which involves testing using the GUI and then going deep into code to go ahead and test it. Due to the nature of the Yii framework, it was especially troublesome to perform test on controllers. After much research we were able to overcome some of the limitation imposed by the framework in order to test the subsystem. For subsystem testing we used PHPUnit which was invaluable for our subsystem validation. For System testing we used IBM Rational Functional Tester, which allowed us to test the system and automate script thus saving us a lot of time of unnecessary manual testing. In the following sections we have include the drivers and script for both system and subsystem.

## 7.1 Subsystem Tests

Due to problems implementing the original real-time whiteboard, and due to having been forced to select an alternate, simpler approach to the whiteboard with tight time constraints, the whiteboard subsystem was not tested. It was, however, tested with system testing.

|  |  |
| --- | --- |
| 1. Identifier | VJF\_SubSystem\_SMSconfirm01 |
| Purpose | This test verifies that the system stores a digit between 1000 and 999 as code |
| Setup | Set user as hello5  Create a fixture with basic info information |
| Input |  |
| Expected Output | Basic->info->code = 0; |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSautocomplete01 |
| Purpose | This test verifies that the system correctly autocompletes the given input |
| Setup | Create a user with username “test” |
| Input | $\_GET ['term'] = 'te'; |
| Expected Output | “test” |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSautocomplete02 |
| Purpose | This test verifies that the system correctly autocompletes the given input |
| Setup | Create a user with username “test” |
| Input | $\_GET ['term'] = 'xxxxxxxx'; |
| Expected Output | “” |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSverify01 |
| Purpose | This test verifies that the system correctly checks for valid screenName |
| Setup | No user in database with username “John” |
| Input | $\_POST ['username'] = 'john'; $\_POST ['SMS'] ['Message'] = "hello"; |
| Expected Output | "User does not exist. <br />" |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSaverify02 |
| Purpose | This test verifies that the system correctly validates message values |
| Setup | Create a user with username “Peter”  Set user as not validated |
| Input | $\_POST ['username'] = 'Peter' ;$\_POST ['SMS'] ['Message'] = ""; |
| Expected Output | "Please enter a message to be sent. No blank messages allowed. <br />User has not set up the SMS functionality. <br />" |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSverify03 |
| Purpose | This test verifies that the system does not send SMS longer than allowed |
| Setup | Create user in database with username “Mike”  Set user as not validated |
| Input | $\_POST ['username'] = 'mike';  $\_POST ['SMS'] ['Message'] = " In a relational database, a Weak Entity is an entity that cannot be uniquely identified by its attributes alone; therefore, it must use a foreign key in conjunction with its attributes to create a primary key.,"; |
| Expected Output | ""The message is too big. Please create a message with less than 160 characters <br />User does not have a phone number in record. <br />User has not set up the SMS functionality. <br />" |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSvalidate01 |
| Purpose | This test verifies that the system correctly handles validation |
| Setup | Create user in database with username “Hello5”  Set Validation Code as 5543 |
| Input | $\_POST ['BasicInfo'] ['smsCode'] = null; |
| Expected Output | "Please enter a code" |
|  | Validated = 0; |
|  |  |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSvalidate02 |
| Purpose | This test verifies that the system correctly handles validation |
| Setup | Create user in database with username “Hello5”  Set Validation Code as 5543  Set tries = 0; |
| Input | Create user in database with username Create $\_POST ['BasicInfo'] ['smsCode'] = '23';  $\_POST ['BasicInfo'] ['smsCode'] = '23';  $\_POST ['BasicInfo'] ['smsCode'] = '23';  $\_POST ['BasicInfo'] ['smsCode'] = '23'; |
| Expected Output | "Invalid codeInvalid codeInvalid codeInvalid code"  Tries = 4;  Validated = 0; |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSvalidate03 |
| Purpose | This test verifies that the system correctly handles validation |
| Setup | Create user in database with username “Hello5”  Set Validation Code as 5543  Set tries = 0; |
| Input | $\_POST ['BasicInfo'] ['smsCode'] = 2; |
| Expected Output | $this->expectOutputString ( "Too many wrong inputs, please contact Hola@aol.com\n" );  Tries = 4;  Validated = 0; |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSvalidate04 |
| Purpose | This test verifies that the system correctly handles validation |
| Setup | Create user in database with username “Hello5”  Set Validation Code as 9555  Set tries = 0; |
| Input | $\_POST ['BasicInfo'] ['smsCode'] = 9555; |
| Expected Output | Tries = 0;  Validated = 1; |

|  |  |
| --- | --- |
| Identifier | VJF\_SubSystem\_SMSchangePref\_01 |
| Purpose | This test verifies that the system correctly changes the sms preferences |
| Setup | Create user in database with username “Hello5”  Set allowSMS = 1 |
| Input | $\_POST ['BasicInfo'] ['allowSMS'] = 0; |
| Expected Output | allowSms = 0; |

|  |  |
| --- | --- |
| Identifier | VJF-055-ss1 test login |
| Purpose | This test verifies that users can successfully login using valid FIU SCIS credentials |
| Setup | set valid fiu SCIS user  Create a fixture with basic info information |
| Input |  |
| Expected Output | User->fiucsid > 1000 |

|  |  |
| --- | --- |
| Identifier | VJF-055-ss2 test user is not duplicate |
| Purpose | This test verifies that FIU SCIS users trying to log in do not have a previous account in the system |
| Setup | set valid fiu SCIS user with an already linked non-FIU SCIS account |
| Input | $\_POST ['panthermail'] = 'ncapo006';  $\_POST ['pantherid'] = "999999"; |
| Expected Output | User email is already linked with another account. |

|  |  |
| --- | --- |
| Identifier | VJF-055-ss3 test user is activated |
| Purpose | This test verifies that FIU SCIS users have their accounts activated immediately upon account creation |
| Setup | set valid fiu SCIS user with no previous account |
| Input |  |
| Expected Output | User->activated >= 1 |

* 1. System Tests

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Send\_SMS\_TC01 |
| Purpose | This test verifies that the system does not allow empty messages to be sent |
| Setup | User is logged in as an employer  User has validated phone number |
| Input | Student username: johnm  Message: “” |
| Expected Output | "Please enter a message to be sent. No blank messages allowed” |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Send\_SMS\_TC02 |
| Purpose | This test verifies that a username does not exist and notifies the user |
| Setup | User is logged in as an employer  User has validated phone number  Student username: Markus25” is not in database |
| Input | Student username: “Markus25”  Message: “Hello sir, how are you? |
| Expected Output | “User does not exist” |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Send\_SMS\_TC03 |
| Purpose | This test verifies that usernames with special characters are properly handled |
| Setup | User is logged in as an employer  User has validated phone number  Student username: @#$#@%###” is not in database |
| Input | Student username: “Markus25”  Message: “Hello sir, how are you? |
| Expected Output | “User does not exist” |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Confirm\_Phone\_number\_TC01 |
| Purpose | This test verifies that correct validation codes are accepted. |
| Setup | User is logged in as an employer  User has validated phone number  User has code 5252 associated with his account |
| Input | Code: 5252 |
| Expected Output | User is redirected to SMS page |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Confirm\_Phone\_number\_TC02 |
| Purpose | This test verifies that incorrect validation codes are not accepted. |
| Setup | User is logged in as an employer  User has validated phone number  User has code 5252 associated with his account |
| Input | Code: 4444 |
| Expected Output | "Invalid code" |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Confirm\_Phone\_number\_TC03 |
| Purpose | This test verifies that null codes are not accepted and user is notified |
| Setup | User is logged in as an employer  User has validated phone number  User has code 5252 associated with his account |
| Input | Code: “” |
| Expected Output | "Please enter a code" |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Share\_Screen\_TC01 |
| Purpose | This test verifies that users are able to share their screens |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page |
| Input | User clicks on Screen Share  User clicks on share Screen |
| Expected Output | “Your screen is being Shared”  Java applet loads |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Share\_Screen\_TC02 |
| Purpose | This test verifies that users are able to share their screens after cancelling the java applet |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page |
| Input | User clicks on Screen Share  User clicks on share Screen  User cancels java applet popup  User clicks on Share Screen  User clicks ok on java applet |
| Expected Output | “Your screen is being Shared”  Java applet loads |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Share\_Screen\_TC03 |
| Purpose | This test verifies that users are not able to share more than two screens at a time |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page |
| Input | User clicks on Screen Share  User clicks on share Screen  User clicks ok on java applet  Screen Sharing app starts  User clicks on Share Screen |
| Expected Output | "There is an active screenShare at the moment, please wait until it has ended before attempting to share a screen" |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_Share\_Screen\_TC04 |
| Purpose | This test verifies that users are not able to share more than two screens at a time |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page |
| Input | User clicks on Screen Share  User clicks on share Screen  User clicks ok on java applet  Screen Sharing app starts  Other user involved in interview clicks on share |
| Expected Output | "There is an active screenShare at the moment, please wait until it has ended before attempting to share a screen" |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_View\_Screen\_Share\_TC01 |
| Purpose | This test verifies that users can correctly see each other’s screen |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page  Student is sharing Screen |
| Input | User clicks on screen Share  User clicks on view screen |
| Expected Output | Other user’s screen is displayed. |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_View\_Screen\_Share\_TC02 |
| Purpose | This test verifies that users can correctly hide the screen Share |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page  Student is sharing Screen |
| Input | User clicks on screen Share  User clicks on view screen  User clicks on hide screen |
| Expected Output | Screen is hidden. |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_View\_Screen\_Share\_TC03 |
| Purpose | This test verifies that the system correctly displays the stock screen |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page  No one is sharing screen |
| Input | User clicks on screen Share  User clicks on view screen |
| Expected Output | Stock screen is displayed |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_End\_Screen\_Share\_TC01 |
| Purpose | This test verifies that the system correctly terminates the screen Share |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page  User is Sharing screen |
| Input | User clicks on end Screen |
| Expected Output | Screen Share is terminated |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_End\_Screen\_Share\_TC02 |
| Purpose | This test verifies that the system correctly terminates the screen Share |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page  User is Sharing screen |
| Input | User clicks on stop share  User clicks on end Screen |
| Expected Output | Screen Share is terminated |

|  |  |
| --- | --- |
| Identifier | VJF\_SYSTEM\_End\_Screen\_Share\_TC03 |
| Purpose | This test verifies that the system correctly terminates the screen Share |
| Setup | User is logged in as an employer  User has set up interview with student  User is in interview page  User is Sharing screen |
| Input | User presses refresh on browser window |
| Expected Output | Screen Share is terminated |

|  |  |
| --- | --- |
| Identifier | VJF-048 Create Document |
| Purpose | This test verifies that users can successfully create a new document |
| Setup | 1. click on Collaborative Editor 2. click on create new document button |
| Input |  |
| Expected Output | "New Document is created” |

|  |  |
| --- | --- |
| Identifier | VJF-048 Create Document |
| Purpose | This test verifies that only loggeid in users can successfully create a new document |
| Setup | 1. click on Collaborative Editor 2. click on create new document button |
| Input | NONE |
| Expected Output | "Unable to create new document, user has been logged out due to innactivity” |

|  |  |
| --- | --- |
| Identifier | VJF-049 Share Active Document |
| Purpose | This test verifies that users can successfully share a new document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents button 3. select active document 4. click on share document 5. Invitation is sent to other user 6. Other user receives notification 7. Other user joins the shared document session |
| Input | NONE |
| Expected Output | "New Document is shared and collaboration on document is initiated” |

|  |  |
| --- | --- |
| Identifier | VJF-049 Share Active Document |
| Purpose | This test verifies that users can not share a new document when a remote user has not logged into the interview |
| Setup | 1. click on Collaborative Editor 2. click on manage documents button 3. select active document 4. click on share document 5. Invitation is sent to other user |
| Input | NONE |
| Expected Output | "User is notified that remote user has not logged in” |

|  |  |
| --- | --- |
| Identifier | VJF-049 Share Active Document |
| Purpose | This test verifies that users can’t share a document unless they has selected one from his list of documents or an active document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents button 3. click on share document |
| Input | NONE |
| Expected Output | "User is notified that no active document has been selected to share” |

|  |  |
| --- | --- |
| Identifier | VJF-050 Delete Document |
| Purpose | This test verifies that users can successfully delete an existing document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. select document to delete 4. click delete document |
| Input | NONE |
| Expected Output | "Document is deleted” |

|  |  |
| --- | --- |
| Identifier | VJF-050 Delete Document |
| Purpose | This test verifies that users can’t delete a document unless they have selected one from his list of documents or an active document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click delete document |
| Input | NONE |
| Expected Output | " User is notified that no active document has been selected to delete” |

|  |  |
| --- | --- |
| Identifier | VJF-051 Import Document |
| Purpose | This test verifies that users can successfully import a document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click on document import 4. drag document over import area 5. document is imported |
| Input | NONE |
| Expected Output | “Document is imported” |

|  |  |
| --- | --- |
| Identifier | VJF-051 Import Document |
| Purpose | This test verifies that users only import a document by dragging it into the specified import area |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click on document import 4. drag document outside of import area |
| Input | NONE |
| Expected Output | “Document import fails” |

|  |  |
| --- | --- |
| Identifier | VJF-051 Import Document |
| Purpose | This test verifies that users can only upload a document that of a valid document format |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click on document import 4. drag document inside import area |
| Input | NONE |
| Expected Output | “Document import fails, document is invalid” |

|  |  |
| --- | --- |
| Identifier | VJF-052 Rename Document |
| Purpose | This test verifies that users can successfully rename a document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click on rename document 4. enter new name in input box 5. document is renamed |
| Input | NONE |
| Expected Output | “Document is renamed” |

|  |  |
| --- | --- |
| Identifier | VJF-052 Rename Document |
| Purpose | This test verifies that users can only rename a document following the utf8-en encoding character set |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click on rename document 4. enter new name in input box 5. document is renamed |
| Input | NONE |
| Expected Output | “Document is not renamed, user entered an invalid character” |

|  |  |
| --- | --- |
| Identifier | VJF-053 Save Document |
| Purpose | This test verifies that users can successfully save a document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click on save document 4. document is saved |
| Input | NONE |
| Expected Output | “Document is saved” |

|  |  |
| --- | --- |
| Identifier | VJF-053 Save Document |
| Purpose | This test verifies that users are warned of document data loss if user chooses to navigate away from active document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. user navigates away from interview portal without saving |
| Input | NONE |
| Expected Output | “User is warned that if they navigates away from editor, document data will be lost/not saved” |

|  |  |
| --- | --- |
| Identifier | VJF-054 Open Document |
| Purpose | This test verifies that users can successfully open a document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. select document to open 4. click open document 5. document is opened |
| Input | NONE |
| Expected Output | “Document is opened in the editor” |

|  |  |
| --- | --- |
| Identifier | VJF-054 Open Document |
| Purpose | This test verifies that users can’t open a document unless they have selected one from his list of documents or an active document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click open document |
| Input | NONE |
| Expected Output | “User is notified that no active document has been selected to open” |

|  |  |
| --- | --- |
| Identifier | VJF-055 Export Document |
| Purpose | This test verifies that users can successfully export a document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. select document to export 4. click export document 5. document is exported |
| Input | NONE |
| Expected Output | “Document is downloaded on the user’s browser” |

|  |  |
| --- | --- |
| Identifier | VJF-055 Export Document |
| Purpose | This test verifies that users can successfully export a document |
| Setup | 1. click on Collaborative Editor 2. click on manage documents 3. click export document |
| Input | NONE |
| Expected Output | “User is notified that no active document has been selected to export” |

|  |  |
| --- | --- |
| Identifier | VJF-055 Login using FIU SCIS Credentials |
| Purpose | This test verifies that users can successfully login using FIU SCIS credentials |
| Setup | 1. click on FIU SCIS login 2. enter username 3. enter password 4. click login/submit |
| Input | FIU SCIS username  Panther id as password |
| Expected Output | “user successfully logs in” |

|  |  |
| --- | --- |
| Identifier | VJF-055 Login using FIU SCIS Credentials |
| Purpose | This test verifies that users can successfully login only using valid FIU SCIS credentials |
| Setup | 1. click on FIU SCIS login 2. enter non-valid username 3. enter non-valid password 4. click login/submit |
| Input | FIU SCIS username  Panther id as password |
| Expected Output | “login form resets itself, user can’t login user has to go through process again” |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Upload\_Image\_TC01** |
| **Purpose** | This test verifies that the system does not non-image files to be uploaded |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started  File “1. CGS 3095 - Course Overview.pdf” was the last uploaded file |
| **Input** | File name: “1. CGS 3095 - Course Overview.pdf” |
| **Expected Output** | This file is not an image. Please try another file. |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Upload\_Image\_TC02** |
| **Purpose** | This test verifies that the system allows .png images to be uploaded |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started  User uploads the file “2a.png” |
| **Input** | File name: “2a.png” |
| **Expected Output** | “File upload was successful!” |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Upload\_Image\_TC03** |
| **Purpose** | This test verifies that the system allows images with consecutive dots (.) to be uploaded |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started  User uploads the file “2a..png” |
| **Input** | File name: “2a..png” |
| **Expected Output** | “File upload was successful!” |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_View\_Uploaded\_Image\_TC01** |
| **Purpose** | This test verifies that the system does not display a non-image file, if it has been uploaded |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started  The “8VJFID6675162.pdf” was the last uploaded file |
| **Input** | File name: “8VJFID6675162.pdf” |
| **Expected Output** | “There is no image to view in the system” |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_View\_Uploaded\_Image\_TC02** |
| **Purpose** | This test verifies that the system that the system does not allow to view an empty upload |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started  The “8VJFID6675162.pdf” was the last uploaded file |
| **Input** | File name: “” |
| **Expected Output** | “There is no image to view in the system” |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_View\_Uploaded\_Image\_TC03** |
| **Purpose** | This test verifies that “.jpg” images are able to be viewed after they are uploaded |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started  The “test.jpg” was the last uploaded file |
| **Input** | File name: “test.jpg” |
| **Expected Output** | Display of image “test.jpg” |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Show\_or\_Restore\_Whiteboard\_TC1** |
| **Purpose** | This test verifies that the whiteboard can be shown after a document has been shared |
| **Setup** | User is logged in as an employer or student  A document has been shared with the document-sharing feature |
| **Input** | User clicks on the “Whiteboard” blue button on the left-hand side of the interview page |
| **Expected Output** | Buttons of the document sharing feature fade from view and the whiteboard buttons and whiteboard screen appears |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Show\_or\_Restore\_Whiteboard\_TC2** |
| **Purpose** | This test verifies that only one instance of the whiteboard is displayed after “Whiteboard” is clicked more than once in a row |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started |
| **Input** | User clicks on the “Whiteboard” blue button on the left-hand side of the interview page |
| **Expected Output** | Buttons of the document sharing feature fade from view and the whiteboard buttons and whiteboard screen appears |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Show\_or\_Restore\_Whiteboard\_TC3** |
| **Purpose** | This test verifies that only one instance of the whiteboard is displayed after “Whiteboard” is clicked more than once in a row |
| **Setup** | User is logged in as an employer or student  Whiteboard session has been started |
| **Input** | User clicks on the “Whiteboard” blue button on the left-hand side of the interview page |
| **Expected Output** | Buttons of the document sharing feature fade from view and the whiteboard buttons and whiteboard screen appears |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Select\_Upload\_Image\_TC01** |
| **Purpose** | This test case verifies that the system supports selecting an image whose file name contains extended ASCII code characters |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User right clicks on “Choose File”  User selects file “ŒšÆß.jpg” |
| **Expected Output** | The file name is displayed in the upload text box successfully |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Select\_Upload\_Image\_TC02** |
| **Purpose** | This test case verifies that the system supports selecting an image whose file name contains extended ASCII code characters |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User right clicks on “Choose File”  User selects file “ŒšÆß.jpg” |
| **Expected Output** | The file name is displayed in the upload text box successfully |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Draw\_With\_Pencil\_Tool\_TC1** |
| **Purpose** | This test case verifies that the right click also draws on the whiteboard |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User right clicks anywhere in the whiteboard screen to draw |
| **Expected Output** | The drawing made by the user appears on the whiteboard screen |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Draw\_With\_Pencil\_Tool\_TC2** |
| **Purpose** | This test case verifies that a drawing persists after a whiteboard screen goes in, out and into view again |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User draws on the whiteboard and clicks on another feature  User clicks on “Whiteboard” again |
| **Expected Output** | The drawing made by the user appears on the whiteboard screen after the whiteboard is in view |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Change\_Pencil\_Tool\_Color\_TC1** |
| **Purpose** | This test case  verifies that the selected pencil tool color is maintained when the whiteboard is substituted by another feature and then displayed again |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User changes the color of the pencil tool by clicking the “Color” button  User clicks on another feature  User clicks on “Whiteboard” again |
| **Expected Output** | The initial color selected should be maintained  as selected in the past whiteboard session |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Change\_Pencil\_Tool\_Color\_TC2** |
| **Purpose** | This test case  verifies that the selected pencil tool color is maintained after an image is uploaded and the whiteboard session restored |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User uploads a new image through the image-sharing tool  User clicks on the “Whiteboard” button again |
| **Expected Output** | The initial color of the pencil tool  is still the same after uploading an image |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Type\_Text\_Into\_Whiteboard\_TC01** |
| **Purpose** | This test case  verifies that the whiteboard container supports word wrapping with the text feature of the whiteboard |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User clicks on right border of the whiteboard container  Text box input: “123456789123456789123456789123456789123456789123456789” |
| **Expected Output** | The overflowing text is sent to the next line instead of disappearing in the non-visible part of the whiteboard |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Type\_Text\_Into\_Whiteboard\_TC02** |
| **Purpose** | This test case  verifies that the text font color is maintained after text is typed with the whiteboard |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started |
| **Input** | User selects the green color from the color palette in the “Color” tab of the whiteboard  Text box input: “Drawing on the text box”  User selects color palette to see which color is selected |
| **Expected Output** | The color selected is the same as the one which was selected before |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Clear\_Whiteboard\_Contents\_TC01** |
| **Purpose** | This test case  verifies that system tells the user that a whiteboard is already empty when clearing an empty whiteboard |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started  Whiteboard has not been written on or has been cleared |
| **Input** | User selects the “Clear” option from the whiteboard menu |
| **Expected Output** | The system lets the user know that the whiteboard is already empty |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Clear\_Whiteboard\_Contents\_TC02** |
| **Purpose** | This test case  verifies that a selected color is maintained after a whiteboard has been cleared |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started  User selects a color from the color palette |
| **Input** | User selects the “Clear” option from the whiteboard menu |
| **Expected Output** | The previous color selected by the user should be still selected when the color palette is displayed |

|  |  |
| --- | --- |
| **Identifier** | **VJF\_Erase\_From\_Whiteboard\_TC01** |
| **Purpose** | This test case verifies that a selected color is maintained after a user has erased from the whiteboard |
| **Setup** | User is logged in as an employer or student Whiteboard session has been started  User selects a color from the color palette  User draws on the whiteboard |
| **Input** | User uses the erase feature by clicking on the “Erase” button on the whiteboard menu and dragging it on top of the drawing |
| **Expected Output** | The previous color selected by the user should be still selected when the color palette is displayed |

## 7.3 Evaluation of Tests

**Subsystem**

|  |  |
| --- | --- |
| Identification | Result |
| VJF\_SubSystem\_SMSconfirm01 | PASS |
| VJF\_SubSystem\_SMSautocomplete01 | PASS |
| VJF\_SubSystem\_SMSautocomplete02 | PASS |
| VJF\_SubSystem\_SMSverify01 | PASS |
| VJF\_SubSystem\_SMSaverify02 | PASS |
| VJF\_SubSystem\_SMSverify03 | PASS |
| VJF\_SubSystem\_SMSvalidate01 | PASS |
| VJF\_SubSystem\_SMSvalidate02 | PASS |
| VJF\_SubSystem\_SMSvalidate03 | PASS |
| VJF\_SubSystem\_SMSvalidate04 | PASS |
| VJF\_SubSystem\_SMSchangePref\_01 | PASS |
| VJF-055-ss1 test login | PASS |
| VJF-055-ss2 test user is not duplicate | PASS |
| VJF-055-ss3 test user is activated | PASS |

**System**

|  |  |
| --- | --- |
| Identification | Result |
| VJF\_SYSTEM\_Send\_SMS\_TC01 | PASS |
| VJF\_SYSTEM\_Send\_SMS\_TC02 | PASS |
| VJF\_SYSTEM\_Send\_SMS\_TC03 | PASS |
| VJF\_SYSTEM\_Confirm\_Phone\_number\_TC01 | PASS |
| VJF\_SYSTEM\_Confirm\_Phone\_number\_TC02 | PASS |
| VJF\_SYSTEM\_Confirm\_Phone\_number\_TC03 | PASS |
| VJF\_SYSTEM\_Share\_Screen\_TC01 | PASS |
| VJF\_SYSTEM\_Share\_Screen\_TC02 | PASS |
| VJF\_SYSTEM\_Share\_Screen\_TC03 | PASS |
| VJF\_SYSTEM\_Share\_Screen\_TC04 | PASS |
| VJF\_SYSTEM\_View\_Screen\_Share\_TC01 | PASS |
| VJF\_SYSTEM\_View\_Screen\_Share\_TC02 | PASS |
| VJF\_SYSTEM\_View\_Screen\_Share\_TC03 | PASS |
| VJF\_SYSTEM\_End\_Screen\_Share\_TC01 | PASS |
| VJF\_SYSTEM\_End\_Screen\_Share\_TC02 | PASS |
| VJF\_SYSTEM\_End\_Screen\_Share\_TC02 | PASS |
| VJF\_SYSTEM\_End\_Screen\_Share\_TC03 | PASS |
| VJF\_Upload\_Image\_TC01 | PASS |
| VJF\_Upload\_Image\_TC02 | PASS |
| VJF\_Upload\_Image\_TC03 | PASS |
| VJF\_View\_Uploaded\_Image\_TC01 | FAIL |
| VJF\_View\_Uploaded\_Image\_TC02 | FAIL |
| VJF\_View\_Uploaded\_Image\_TC03 | PASS |
| VJF\_Show\_or\_Restore\_Whiteboard\_TC01 | PASS |
| VJF\_Show\_or\_Restore\_Whiteboard\_TC02 | PASS |
| VJF\_Show\_or\_Restore\_Whiteboard\_TC03 | PASS |
| VJF\_Select\_Upload\_Image\_TC01 | PASS |
| VJF\_Select\_Upload\_Image\_TC02 | PASS |
| VJF\_Draw\_With\_Pencil\_Tool\_TC01 | PASS |
| VJF-Create Document\_TC01 | PASS |
| VJF-Create Document\_TC01 | PASS |
| VJF-Share Active Document \_TC01 | PASS |
| VJF-Share Active Document \_TC01 | PASS |
| VJF-Share Active Document\_TC01 | PASS |
| VJF-Delete Document\_TC01 | PASS |
| VJF-Delete Document\_TC00 | PASS |
| VJF-Import Document\_TC01 | PASS |
| VJF-Import Document\_TC02 | PASS |
| VJF-Import Document\_TC03 | PASS |
| VJF-Rename Document\_TC01 | PASS |
| VJF- Rename Document\_TC02 | PASS |
| VJF-Save Document\_TC01 | PASS |
| VJF- Save Document\_TC02 | PASS |
| VJF-Save Document\_TC03 | PASS |
| VJF-Open Document\_TC01 | PASS |
| VJF-Open Document\_TC02 | PASS |
| VJF-Export Document\_TC01 | PASS |
| VJF-Export Document\_TC02 | PASS |
| VJF-Login using FIU SCIS Credentials \_TC01 | PASS |
| VJF-Login using FIU SCIS Credentials\_TC02 | PASS |

## **8. Glossary**

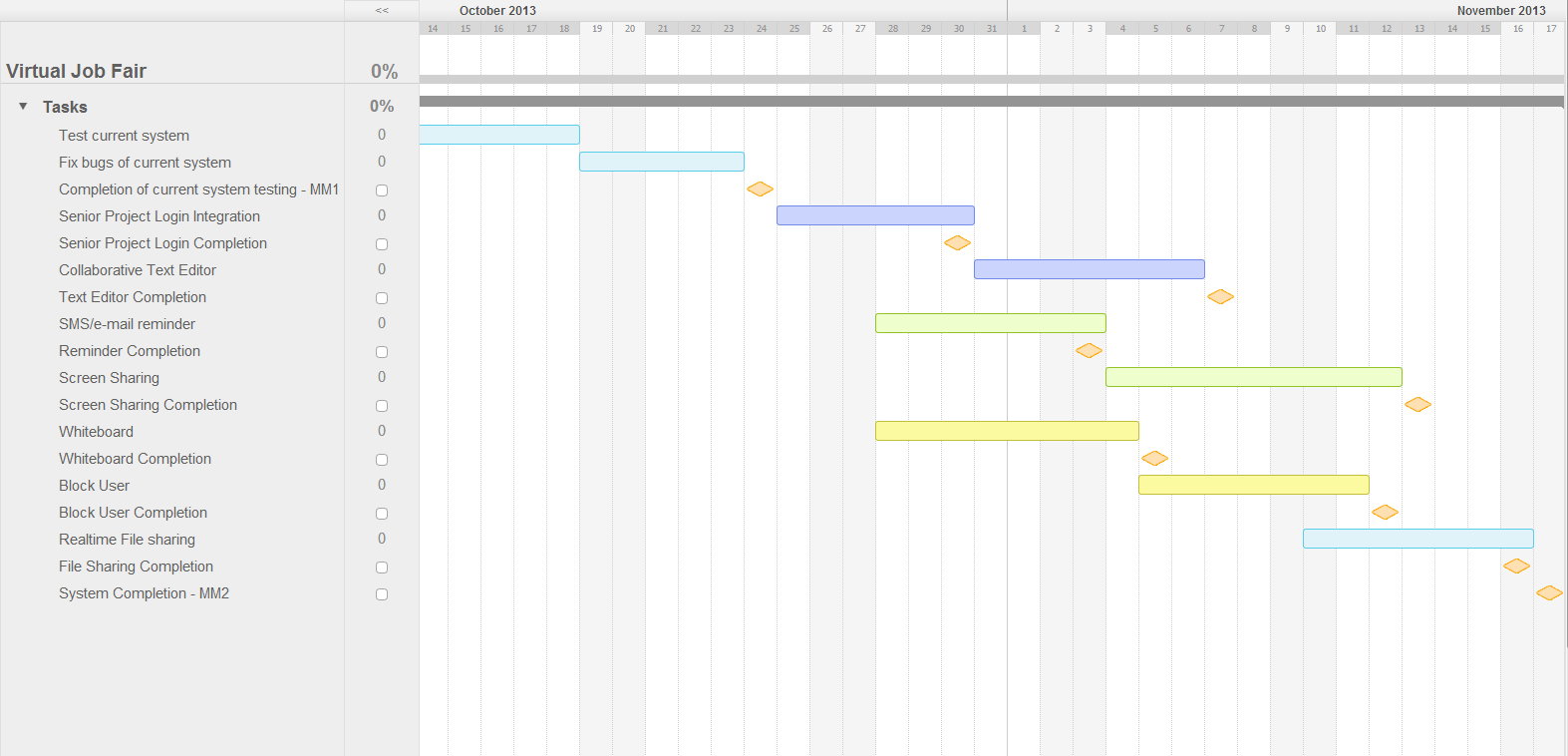
**- Salary:** a periodic payment made to an employee in exchange for services provided. Salaries are provided in yearly terms.

- **Résumé:** a document which describes a student’s qualifications, skills and education

- **Cover Letter:** a document which is used by students to introduce themselves to the companies that they are applying to. It usually goes together with a résumé

# 9. Appendix

* 1. Appendix A - Project schedule (Gantt chart or PERT chart).



## 9.2 Appendix B

**Current System’s Functional Requirements**

The system shall…

· **Allow students and employers to register**

- **Usability**: The register form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 2 seconds.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to view respective profiles**

- **Usability**: Data displayed in profiles is easy to follow. Students are only able to see their own profile and the employer ones. Employers can see all student profiles.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to edit their basic profile information**

- **Usability**: The edit form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to take part in a video interview**

- **Usability**: Starting a video interview is simple and understandable.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 3 seconds when connecting.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to upload a resume and video resume**

- **Usability**: The upload form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1-5 seconds, depending on the file size.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students and employers to upload an image for their profile**

- **Usability**: The upload form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to associate skills to their profile**

- **Usability**: The ability to add skills to a profile is simple and understandable. It can be done by using LinkedIn connect or adding them manually.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within one 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to add and delete education information**

- **Usability**: The corresponding form is easy to complete and follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to add and delete experience information**

- **Usability**: The corresponding form is easy to complete and follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to integrate with their LinkedIn account to provide education and experience information (security)**

- **Usability**: The connection with LinkedIn should be easy to follow. Users will enter their LinkedIn credentials and get appropriate data that the user allowed.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to apply to open job postings and provide a cover letter**

- **Usability**: Students are presented with a user-friendly interface that is easy to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to reply to an employer’s message**

- **Usability**: Students are presented with a clear and simple interface to send messages.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow students to search for jobs based on skills**

- **Usability**: The search form is easy to follow and complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to post jobs**

- **Usability**: The post job form is easy to understand and complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to close a job posting**

- **Usability**: The closing of a post is easy to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to associate skills to a job posting**

- **Usability**: the addition of skills to a post is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second when adding each skill.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to search for students based on skills**

- **Usability**: The search form is simple to submit.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to view student profiles**

- **Usability**: The view of a student profile is easy to understand.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within one 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to send messages to students**

- **Usability**: Employers are presented with a clear and simple interface to send messages.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow employers to give students a “virtual handshake” to show interest in the student**

- **Usability**: The virtual handshake form is easy complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow an administrator to disable an account (security)**

- **Usability**: Disabling a user is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow an administrator to close a job posting**

- **Usability**: Closing a job post is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Allow an administrator to validate an employer registration (security)**

- **Usability**: The validation of an employer is done by one click and is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Require a username and password to log into the system (security)**

- **Usability**: This is required for a user to log in. Form is simple and easy to follow.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Encrypt the user password before storing into the database (security)**

- **Usability**: Storing user password in a secure way without user intervention.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

· **Require login before viewing user profiles (security)**

- **Usability**: Security measure for system. Interface is simple to complete.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should respond within 1 second.

- **Supportability**: The system should be easy to maintain and make appropriate changes.

**New System’s Functional Requirements**

**· Allow users to create a new shared document.**

- **Usability:** The document creation interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to open a document.**

- **Usability:** The document open interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to share a document.**

- **Usability:** The invitation should be transparent to the users upon document creation. The invitation interface on new documents should be easy to use.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to delete a document.**

- **Usability:** The document deletion interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 second.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to import a document.**

- **Usability:** The import document interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to export a document.**

- **Usability:** The export document interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to rename document.**

- **Usability:** The document rename interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow users to save a shared document.**

- **Usability:** The document saving interface should be intuitive, easy to understand and navigate.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 3 seconds.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Maintain access boundaries between non-collaborating temporary accounts (security).**

- **Usability:** Temporary accounts not related to a collaborating session shall have no access to that session’s documents.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 second.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

**· Allow FIU Computer Science Seniors to login using their FIU SCIS credentials**

- **Usability:** The system should provide an easy and integrated login process for FIU SCIS Seniors using the school UNIX account.

-  **Reliability:** The system should perform correctly 99% of the time.

- **Performance:** The system should respond in less than 1 second.

- **Supportability:** The system should be supported on Google Chrome version 29+ The system should be easily maintainable, allowing for improvements, corrections and adaptation in environment, requirement and/or specification changes.

· **Allow users to share their screens**

- **Usability**: User should be able to share their screens solely with mouse clicks

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should start streaming content in under 3 seconds

- **Supportability**: Screen sharing page should be supported by Google Chrome versions 29 and up.

· **Allow users to view shared screens**

- **Usability**: User should be able to watch a shared screen with single mouse click

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should start displaying the screen in under 2 seconds

- **Supportability**: Screen sharing page should be supported by Google Chrome versions 29 and up.

· **Allow users to stop sharing their screens**

- **Usability**: User should be able to stop sharing their screens with a single mouse click

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The system should stop the stream in under 1 second.

- **Supportability**: Screen sharing page should be supported by Google Chrome versions 29 and up.

· **Allow employers to send SMS to students**

- **Usability**: Sending an SMS should not take more than 15 seconds for a novice user.

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: The SMS should be sent in under 2 seconds after user presses send

- **Supportability**: Sending SMS page should be supported by IE, Firefox, Chrome and Safari.

· **Allow users to receive automatic email and SMS reminders**

- **Usability**: Not applicable

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: Users should receive a reminder 30 minutes before scheduled interview

- **Supportability**: Not applicable

· **Allow users to confirm their phone numbers**

- **Usability**: Confirming a phone number should take less than 30 seconds for inexperienced users

- **Reliability**: The system should perform correctly 99% of the time.

- **Performance**: Sending authentication code and validating it should take less than 2 seconds respectively.

- **Supportability**: Confirm phone number page should be supported by IE, Firefox, Chrome and Safari

**· Allow students and employers to start using the whiteboard functionality**

**- Usability:** Starting the whiteboard is a matter of clicking a button while interviewing, making it a fairly easy process to do

**- Reliability:** The whiteboard tool should be available 90% of the time, since not every major/employer will use it

**- Performance:** The whiteboard should be started within 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to upload an image to share during an interview**

**- Usability:** if an image has been selected for uploading, actually uploading it to the system is as simple as click the “Submit Drawing” button, making it simple to use

**- Reliability:** Uploading images should work 99% of the time, since it is a critical function of the image-sharing subsystem

**- Performance:** Depending on its size, the maximum time it should take to upload an image should be 10 seconds

**- Supportability:** The system be supported by Chrome version 29 or above

· **Allow students to view images uploaded by the other party in an interview**

**- Usability:** viewing an image which has been uploaded already relies simply on clicking a “View Drawing” button, so it is fairly simple to use

**- Reliability:** as uploading an image, viewing an image should be working 99% of the time, since it is a critical function of the image-sharing subsystem

**- Performance:** depending on the size of the image, the maximum time it should take to view an uploaded image should be 10 seconds

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers show or restore a whiteboard session**

**- Usability:** starting a whiteboard session involves only clicking the “Whiteboard” button below the video chat section of the interview page, making it a simple process

**- Reliability:** the whiteboard should be available 99% of the time, since it is a main feature of the system

**- Performance:** displaying the whiteboard should take 1 second, since a default whiteboard is not preloaded with any information

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to select an image to upload to the server for sharing purposes**

**- Usability:** selecting an image requires basic experience with a browsing dialog, which should be fairly common knowledge, making it a simple functionality

**- Reliability:** the ability to select an image is critical to the image-sharing feature, so it should be available 99% of the time

**- Performance:** since selecting an image does not depend on image size, this functionality should take less than 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to draw using the whiteboard**

**- Usability:** drawing on the whiteboard simply requires to drag the mouse through the whiteboard’s area, which makes it a simple process

**- Reliability:** without being able to draw in the whiteboard, the whiteboard serves no purpose, so drawing in the whiteboard should be available 99.99% of the time

**- Performance:** since the whiteboard is self-contained and does not need any API calls, the drawing should take less than half a second

**- Supportability:** The system be supported by Chrome version 29 or above

**Allow students and employers to change the color of the drawing pencil tool**

**- Usability:** changing the color is a matter of clicking on the new color desired, so it is fairly simple to do

**- Reliability:** changing the color of the drawing tool is not of the utmost importance, so it is expected to work 90% of the time

**- Performance:** since the whiteboard is self-contained and does not need any API calls, changing the color should take less than half a second

**- Supportability:** The system be supported by Chrome version 29 or above

**Allow students and employers to type text into the whiteboard**

**- Usability:** typing text into the whiteboard requires clicking a button and typing text, so it is fairly simple to use

**- Reliability:** since typing text is not the main whiteboard feature, it is expected to work 90% of the time

**- Performance:** the text should appear in the whiteboard in less than three seconds

**- Supportability:** The system be supported by Chrome version 29 or above

**· Allow students and employers to clear the drawings of the whiteboard**

**- Usability:** Clearing the contents is a matter of clicking a single button, making it a fairly easy process to conduct

**- Reliability:** It should be available 99% of the time that the whiteboard is being used, given that new drawings might be required

**- Performance:** Whiteboard screen should be cleared out within 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**Allow students and employers to partially erase drawings from the whiteboard**

**- Usability:** Partially erasing from the drawing is a matter of clicking a single button, drawing the cursor and releasing the click, making it a fairly easy process to conduct

**- Reliability:** It should be available 99% of the time that the whiteboard is being used, given that a user might make a mistake while drawing or that new drawings might be required

**- Performance:** selected sections should be erased within 1 second

**- Supportability:** The system be supported by Chrome version 29 or above

**Current System’s Functional Requirements Descriptions**

The first 41 use cases were implemented by the previous group, while the remaining ones were added for Virtual Job Fair 2.0.

|  |  |
| --- | --- |
| Use Case ID | VJF-001 Registration |
| Description | Registration process for a student type |
| Actor | Student |
| Pre-conditions | ·         User has navigated to Virtual Job Fair |
| Steps | 1. User clicks on register link 2. User selects registration 3. User selects student or employer 4. User fills in required details 5. User clicks submit |
| Post-conditions | 1. User is redirected to page where he/she is asked to check email for verification link 2. Verification email is sent |
| Exceptions | 1. User fails to fill out one of the details in the form 2. User fails to provide an FIU email address |

|  |  |
| --- | --- |
| Use Case ID | VJF-002 View Profile |
| Description | Viewing profiles for student and employers |
| Actor | All Actors |
| Pre-conditions | 1. User is logged in |
| Steps | 1. User clicks on username hyperlink 2. User is redirected to the user profile |
| Post-conditions | 1. User is on profile page |
| Exceptions | 1. If a student is viewing another student’s profile, certain information is withheld |

|  |  |
| --- | --- |
| Use Case ID | VJF-003 Edit Basic Info |
| Description | Allow a user to edit their profile |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in |
| Steps | 1. User clicks on “My Profile” 2. User clicks on “Edit Basic Info” 3. User changes necessary data 4. User clicks “Save” |
| Post-conditions | 1. User is redirected to “My Profile” page |
| Exceptions | 1. User inputs invalid/empty data |

|  |  |
| --- | --- |
| Use Case ID | VJF-004 Verify Email |
| Description | Allow a user to verify his account |
| Actor | Student, Employer, Faculty |
| Pre-conditions | 1. User has registered for an account and email has been sent |
| Steps | 1. User navigates to verification email sent by system 2. User clicks on verification link 3. User is redirected to Virtual Job Fair page to verify Email |
| Post-conditions | 1. Access is granted to user with correct verification link |
| Exceptions | 1. Verification link does not match system’s expectectation |

|  |  |
| --- | --- |
| Use Case ID | VJF-005 Login |
| Description | Allow a user to login to his account |
| Actor | All user types |
| Pre-conditions | 1. User has navigated to Virtual Job Fair website |
| Steps | 1. User enters username and password 2. User clicks “Login” 3. User is redirected to his home page |
| Post-conditions | 1. User is on homepage |
| Exceptions | 1. User entered incorrect username/password combination |

|  |  |
| --- | --- |
| Use Case ID | VJF-006 Logout |
| Description | Allow a user to logout from his account |
| Actor | All user types |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks Logout 2. User is redirected to main page |
| Post-conditions | 1. User is on main page |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-007 Home Page |
| Description | Allow a user to visit Home Page |
| Actor | All user types |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks Home 2. User is redirected to his home page |
| Post-conditions | user is his home page |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-008 Upload Picture |
| Description | Allow user to change his profile picture |
| Actor | all user types |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | User clicks My Profile                 User is redirected to his profile page                 User clicks on the edit picture buttom                 User pick his picture and then click save. |
| Post-conditions | user post his picture |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-009 Upload Resume |
| Description | Allow user to Upload his resume |
| Actor | Student |
| Pre-conditions | User is on Virtual Job Fair                 User is logged in |
| Steps | User clicks My Profile                 User is redirected to his profile page                 User clicks on the edit resume button                 User pick his resume file and then click save. |
| Post-conditions | user post his resume |
| Exceptions | User file invalid/empty data |

|  |  |
| --- | --- |
| Use Case ID | VJF-0011 Add Education |
| Description | Allow user to add education to his profile |
| Actor | Student |
| Pre-conditions | User is on Virtual Job Fair                 User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on the add education 4. User add his education info then click save |
| Post-conditions | user add education to his profile |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0012 Delete Education |
| Description | allow user to delete education from his profile |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on the delete education button |
| Post-conditions | user delete the education that has been chosen to be deleted |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0013 Add Experience |
| Description | user can add experience to his profile |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on the add experience 4. User add his experience info then click save |
| Post-conditions | user add experience to his profile |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0014 Delete Experience |
| Description | user can delete experience from his profile |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on the delete experience button |
| Post-conditions | user delete the experience that has been chosen to be deleted |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0015 Change password |
| Description | user can change his password |
| Actor | All Actors |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on the change password button 4. User is redirect to a new page for changing his password 5. User is typing his old, new password and retype the new password 6. User click submit and redirect to the login page |
| Post-conditions | user change his password |
| Exceptions | - Old Password was incorrect.  - Passwords do not match |
| Use Case ID | VJF-0016 Add Skill |
| Description | user can add skill to his profile |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User type a new skill 4. User clicks add skill |
| Post-conditions | user add a new skill |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0017 Delete skill |
| Description | user can delete skill from his profile |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User is clicking on the skill delete button |
| Post-conditions | Skill is deleted |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0018 Change skills Order |
| Description | user can change the order of each skill |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User drags the skill to any position 4. User clicks save skills |
| Post-conditions | User change the order of the skill |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0019 Integrate LinkedIn |
| Description | Get user information from LinkedIn |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on LinkedIn link 4. User is redirected to page where prompted for username and password for LinkedIn 5. User enters username and password for LinkedIn and clicks continue. 6. User is redirected to My Profile with complete information from LinkedIn |
| Post-conditions | User profile is built |
| Exceptions | User cancels the action |
| Use Case ID | VJF-0020 Start Video Interview |
| Description | Start Video Interview |
| Actor | Employer, Student |
| Pre-conditions | 1. Actors are logged in 2. A video interview has been previously scheduled 3. A notification for the video interview was sent to both parties and is displaying in the homepage 4. Actors are in homepage |
| Steps | 1. Actor clicks on link to video interview in the notifications window in homepage    1. Notification for employer: ([You schedule interview with Diego on 2013-03-13 at 3:00pm](http://srprog-spr13-01.aul.fiu.edu:8080/demos/videointerview.html?view=Diego&notificationRead=842&usertype=2) )    2. Notification for Student: ([Company IBM wants to have a video interview with you](http://srprog-spr13-01.aul.fiu.edu:8080/demos/videointerview.html?view=IBM) [2013-03-13 at 3:00pm](http://srprog-spr13-01.aul.fiu.edu:8080/demos/videointerview.html?view=Diego&notificationRead=842&usertype=2) .[Good Luck!](http://srprog-spr13-01.aul.fiu.edu:8080/demos/videointerview.html?view=IBM) ) 2. Actor is redirected to the video interview page where he/she sees to the left the video connection tools and the video window, and to the right: 3. For the employer:    1. the profile of the student participating in the interview. 4. For the student:    1. the profile of the employer conducting the interview. 5. Once the employer is ready to start the interview he/she clicks in the connect button which allows the student participating in the interview to connect right after 6. Once both parties are connect the video interview is started 7. Once the interview is finish, the employer and student clicks the finish button to be disconnected |
| Post-conditions | Both parties participated in a video interview |
| Exceptions | Actor ends the interview before it is finished |

|  |  |
| --- | --- |
| Use Case ID | VJF-0021 Accept Interview |
| Description | Student user accepts a video interview |
| Actor | Student |
| Pre-conditions | Actor is logged in and is at the home page |
| Steps | 1. Actor clicks on the video interview notification section. 2. Actor is notified that a video interview has been schedule for him 3. Actor is asked to confirm the video interview 4. Actor clicks the accept button |
| Post-conditions | The video interview is confirmed and a notification is sent back to the user who originated the interview (employer user) |
| Exceptions | Actor does not accept the interview and closes the notification |

|  |  |
| --- | --- |
| Use Case ID | VJF-0022 View Student Profile |
| Description | Allow Employer to view student profile |
| Actor | Employer |
| Pre-conditions | Employer is logged in and is at the home page |
| Steps | 1. Employer is typing student name on the search input. 2. Employer is clicking on the student that he want to view |
| Post-conditions | The employer is on the student profile view |
| Exceptions | The employer is typing a wrong student name that doesn't exist |
| Use Case ID | VJF-0023 Send Message |
| Description | Send a message to a user |
| Actor | A user |
| Pre-conditions | -User is in the compose message page |
| Steps | 1-User populates the ‘To’ field  2-User populates the ‘Subject’ field  3-User types in the message in the text area  4-User clicks ‘Send’ |
| Post-conditions | The system sends the message. The message appears in the inbox of the target user |
| Exceptions | -Inexistent username selected as the receiver of the message  -Wrong username format typed in the ‘To’ field |

|  |  |
| --- | --- |
| Use Case ID | VJF-0024 Reply to Message |
| Description | Reply to a message from some user |
| Actor | A user |
| Pre-conditions | -User has selected a message to be seen |
| Steps | 1-User clicks on the Reply button that appears when reading a message |
| Post-conditions | 2- The System redirects the user to the compose a message page, and the original message the user had selected appears in the text area in the format:  On <Date> <User> wrote:  <message> |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0025 Get Inbox |
| Description | User requests to see all the received messages |
| Actor | A User |
| Pre-conditions | -User is logged in |
| Steps | 1- User navigates to the messages page |
| Post-conditions | User is shown with a list of all the received messages in the format: <Sender>  <Subject> |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0026 Get Sent Messages |
| Description | User requests to see all the messages he/she has sent |
| Actor | A User |
| Pre-conditions | -User is logged in |
| Steps | 1- User selects the ‘Sent” messages from the messages page |
| Post-conditions | User is shown with a list of all the sent messages in the format: <Receiver>  <Subject> |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0027 Get Trashed Messages |
| Description | User requests to see all the trashed messages |
| Actor | A User |
| Pre-conditions | -User is logged in |
| Steps | 1- User selects the ‘Trash” messages from the messages page |
| Post-conditions | User is shown with a list of all the trashed messages in the format: <Sender/Receiver>  <Subject> |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0028 Delete Messages |
| Description | User checks all the messages he/she wants to send to the trash |
| Actor | A User |
| Pre-conditions | -User is logged in |
| Steps | 1-User checks the messages to be sent to the trash  2-User clicks on the trash icon |
| Post-conditions | The selected messages are sent to the trash |
| Exceptions | User does not select any messages before clicking on the trash icon. The System invokes an alert message |

|  |  |
| --- | --- |
| Use Case ID | VJF-0029 Post Job |
| Description | Employer posts a job for students to apply to |
| Actor | Employer |
| Pre-conditions | 1. Employer is logged in 2. Employer is on Home Page |
| Steps | 1. Employer clicks on Post Job Menu Item 2. Employer fills in job details (type, description, compensation, expire date) 3. Employer adds skills to posting if necessary 4. Employer clicks post job |
| Post-conditions | 1. Employer is taken to student match page to view students whose skillset is aligned with the job skillset |
| Exceptions | 1. Employer fills in job details incorrectly, is given an error |
| Use Case ID | VJF-0030 Virtual Handshake |
| Description | Employer gives student a virtual handshake to show interest |
| Actor | Employer, Student |
| Pre-conditions | 1. Employer has posted a job |
| Steps | 1. After employer posts job, he is taken to a student match page 2. Employer can review the list of students who matches with the job he posted 3. Employer clicks on “virtual handshake” for any student |
| Post-conditions | 1. Employer remains on student match page 2. Student receives a notification that the employer has shown interest in him for the position |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0031 Edit Company Info |
| Description | Employer changes his company Information |
| Actor | Employer |
| Pre-conditions | 1. Employer is logged in 2. Employer is viewing his profile |
| Steps | 1. Employer clicks on edit image in company info section 2. Employer changes text in any of the text boxes 3. Employer clicks on checkmark |
| Post-conditions | 1. New company info is saved 2. Employer is redirected back to profile page |
| Exceptions | Employer fills in invalid values for the fields |
| Use Case ID | VJF-0032 Search Jobs |
| Description | Student searches for jobs by skill |
| Actor | Student |
| Pre-conditions | 1. Student is logged in 2. Student is on home page |
| Steps | 1. Student clicks on job search text box 2. Student begins typing a skill 3. Student either selects skill from auto complete or fully types out skill 4. Student clicks submit |
| Post-conditions | 1. Student is redirected to search result page with relevant jobs, and option to search more jobs |
| Exceptions | None |
| Use Case ID | VJF-0033 Search Students |
| Description | Student searches for students by skill |
| Actor | Employer |
| Pre-conditions | 1. Employer is loggedin 2. Employer is on home page |
| Steps | 1. Employer clicks on job search text box 2. Employer begins typing a skill 3. Employer either selects skill from auto complete or fully types out skill 4. Employer clicks submit |
| Post-conditions | Employer is redirected to search result page with relevant students, and option to search more students |
| Exceptions | None |
| Use Case ID | VJF-0034 Close Job |
| Description | Close a job from further applications |
| Actor | Employer |
| Pre-conditions | 1. Employer is logged in |
| Steps | 1. Employer views his own profile 2. Employer selects a job from one of his own postings 3. Employer clicks on “Close Job” |
| Post-conditions | 1. User is redirected back to the job page 2. The job is closed |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0035 Administrator Close Job |
| Description | Close a job from further applications |
| Actor | Admin |
| Pre-conditions | 1. Admin is logged in |
| Steps | 1. Admin goes to home page 2. Admin enters text included in a job title 3. Admin is redirected to search results with a list of relevant jobs 4. Admin clicks on “delete” for a job of his choosing |
| Post-conditions | 1. The respective job is closed 2. Admin is redirected to search page |
| Exceptions | Search may not return any results |

|  |  |
| --- | --- |
| Use Case ID | VJF-0036 Disable User |
| Description | Disable a user from the website |
| Actor | Admin |
| Pre-conditions | 1. Admin is logged in 2. Admin is on home page |
| Steps | 1. Admin enters a search for a username 2. Admin is taken to results page with list of users 3. Admin can disable users by clicking on “delete” |
| Post-conditions | 1. User is disabled 2. Admin is taken back to search page for more users |
| Exceptions | Search may not return any results |

|  |  |
| --- | --- |
| Use Case ID | VJF-0037 Apply to Job |
| Description | Student applies to an employers job posting |
| Actor | Student |
| Pre-conditions | Student is viewing a job |
| Steps | 1. Student clicks on apply 2. System displays a popup box 3. Student fills in a cover letter 4. Student clicks submit |
| Post-conditions | 1. System notifies employer of new application 2. User is redirected back to job page |
| Exceptions | Student has already applied for the job |
| Use Case ID | VJF-0038 Read notification |
| Description | User read notification from his/home page |
| Actor | All user types |
| Pre-conditions | 1. User is logged in 2. User is on home page |
| Steps | 1. User is clicking on the notification section that he or she will like to read from. 2. User gets list of notifications |
| Post-conditions | User read his notifications |
| Exceptions | None |

|  |  |
| --- | --- |
| Use Case ID | VJF-0039 Schedule interview |
| Description | Employer is schedule interview with the student |
| Actor | Employer |
| Pre-conditions | 1. Employer is logged in 2. Employer is on student profile view. |
| Steps | 1. Employer is clicking on the button video interview. 2. Employer is choosing the date and time. 3. Employer clicks submit |
| Post-conditions | 1. System is notifies the employer for a new schedule interview that he posted 2. System is notifies the student for a new schedule interview that he has been invited for. |
| Exceptions | employer type wrong input for time and date |
| Use Case ID | VJF-0040 Validate an Employer Register |
| Description | Admin validate a new employer that register |
| Actor | Admin |
| Pre-conditions | 1. Admin is logged in. 2. Admin is on his home page. |
| Steps | Admin is clicking on the notification like that validate the new employer. |
| Post-conditions | The new employer got validate. |
| Exceptions | noon |

|  |  |
| --- | --- |
| Use Case ID | VJF-041 Upload Video Resume |
| Description | Allow user to Upload his resume |
| Actor | Student |
| Pre-conditions | 1. User is on Virtual Job Fair 2. User is logged in |
| Steps | 1. User clicks My Profile 2. User is redirected to his profile page 3. User clicks on the edit video resume button 4. User pick his resume file and then click save. |
| Post-conditions | user post his video resume |
| Exceptions | User file invalid/empty data |

|  |  |
| --- | --- |
| Use Case ID | VJF-042 Share Screen |
| Description | Allow a user to share his screen |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in. 2. User is in the homepage 3. Interview has been scheduled. 4. Notification for the interview is displayed for both involved parties 5. Other user involved in interview is not sharing screen |
| Steps | 1. User clicks on scheduled interview notification 2. User is redirected to the interview page. 3. User clicks on share screen |
| Post-conditions | User is able to share his screen and database is update with required information. |
| Exceptions | User tries to share screen while the other party is sharing. |
| Use Case ID | VJF-043 View Screen Share |
| Description | Allow a user to see a shared screen |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in. 2. User is in the homepage 3. Interview has been scheduled. 4. Notification for the interview is displayed for both involved parties 5. Other user involved in interview is sharing a screen |
| Steps | 1.      User clicks on scheduled interview notification  2.      User is redirected to the interview page.  3.      User clicks on view screen share |
| Post-conditions | User is able to view screen shared by other party |
| Exceptions | The other user is not sharing a screen |

|  |  |
| --- | --- |
| Use Case ID | VJF-044 End Screen Sharing |
| Description | Allow a user to end current screen sharing |
| Actor | Student, Employer |
| Pre-conditions | 1.       User is logged in.   1. User is in the interview portal   3.      User is sharing screen |
| Steps | 1.      User clicks on end screen sharing |
| Post-conditions | 1.   User is not sharing screen  2.   System information is updated to allow other party to share screen  3.   Other party involved will not continue to see live feed from user screen |
| Use Case ID | VJF-045 Send SMS to student |
| Description | Allows Employer to send a text message to student |
| Actor | Employer |
| Pre-conditions | 1. User is logged in. 2. User is in the homepage 3. Student has a phone number associated with his account 4. Student has allowed employers to contact him through SMS |
| Steps | 1.      User clicks on send SMS  2.      User is redirected to SMS page  3.   User enters user name of student to contact  4.   User enters message  5.   User presses send |
| Post-conditions | Selected Student receives text message on his phone. |
| Exceptions | The student has not allowed contact by sms  Student has not entered a phone number |

|  |  |
| --- | --- |
| Use Case ID | VJF-046 Send interview reminder |
| Description | An interview reminder is sent to involved parties |
| Actor | Time |
| Pre-conditions | 1. An employer has set up an interview appointment 2. Interview starts in the next 30 minutes. |
| Steps | 1.      Database is continuously checked for interviews starting in the next 30 minutes  2.      After finding jobs that meet this criteria an email message is sent to involved parties about the event.  3.      If any of the accounts is set up to receive SMS then a text message will be sent as well. |
| Post-conditions | Parties involved in the interview will receive an email reminder and SMS according to set up permissions |
| Use Case ID | VJF-047 Confirm phone number |
| Description | Allows user to confirm and validate a phone number |
| Actor | Employee, Student |
| Pre-conditions | 1.      User entered a phone number in the system  2.      User is logged in.  3.      User is in the homepage  4. User phone has not been validated |
| Steps | 1.      User clicks on SMS page  2.      User is redirected to validate number page  3.      User clicks on validate phone  4.      An SMS message is sent to user’ phone  5.      User enters received code  6.      User presses validate button |
| Post-conditions | The system is updated to reflect phone validation |
| Exceptions | User enters a wrong authentication code |

|  |  |
| --- | --- |
| Use Case ID | VJF-048 Create new document |
| Description | Allow user to create a new document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on create new document button 3. The New document editing session is started |
| Post-conditions | 1. User is on the interview portal 2. A new document is displayed |
| Exceptions | 1. Connection Error |

|  |  |
| --- | --- |
| Use Case ID | VJF-049 Share active document |
| Description | Allow users to invite another user to a shared document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one document to share |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on share document button 3. Invitation is sent to other user 4. Other user receives notification 5. Other user joins the shared document session |
| Post-conditions | 1. User is on the interview portal 2. User is on the shared document session |
| Exceptions | 1. Connection Error 2. Other user not available |

|  |  |
| --- | --- |
| Use Case ID | VJF-050 Delete shared document |
| Description | Allow user to delete a document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one document to delete |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on the manage documents button 3. User selects a document from documents list 4. User clicks on the delete document button 5. User is presented with a confirmation dialog 6. User confirms deletion of file 7. Document is deleted |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error 2. User did not select a document to delete |

|  |  |
| --- | --- |
| Use Case ID | VJF-051 Import document |
| Description | Allow user to import a document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one document to import |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on import document button 3. User chooses file to import and drags it over the import document area 4. The document is imported into the system |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error 2. File is not a valid document |

|  |  |
| --- | --- |
| Use Case ID | VJF-052 Rename document |
| Description | Allow user to rename a document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one document to rename |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on the manage documents button 3. User clicks on document to rename 4. User clicks on the rename document button 5. Rename dialog appears 6. User chooses new document name 7. The document name is changed |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error |

|  |  |
| --- | --- |
| Use Case ID | VJF-053 Save document |
| Description | Allow user to save a document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one active document to save |
| Steps | 1. User clicks on save document link 2. The document is saved 3. User is notified |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error 2. There are no active documents to save |

|  |  |
| --- | --- |
| Use Case ID | VJF-054 Open document |
| Description | Allow user to open a document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one active document to open |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on manage documents 3. User clicks on document to open 4. User clicks on the open document button 5. Document is loaded into the Editor |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error 2. There are no active documents to open |

|  |  |
| --- | --- |
| Use Case ID | VJF-055 Export document |
| Description | Allow user to export a document |
| Actor | Student, Employer |
| Pre-conditions | 1. User is logged in 2. User is on the interview portal 3. User has at least one active document to export |
| Steps | 1. User clicks on Collaborative Editor 2. User clicks on the manage documents button 3. User clicks on document to export 4. User clicks on export document 5. User receives the document as a file |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error 2. There are no active documents to export |

|  |  |
| --- | --- |
| Use Case ID | VJF-056 Log in using FIU SCIS Credentials |
| Description | Allow certain users, FIU SCIS, to login to the system using their FIU SCIS Credentials / UNIX account, Provided by the Senior Project API |
| Actor | Student |
| Pre-conditions | 1. User is at the Login page |
| Steps | 1. User enters his / her username 2. User enters his / her password 3. User is logged in |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. User fails to fill the login form 2. Connection Error |

|  |  |
| --- | --- |
| Use Case ID | VJF-057 Maintain document access boundaries |
| Description | Users accessing the system using their personal accounts will have no access to shared documents in the interview session |
| Actor | System |
| Pre-conditions | 3. User is logged in using their personal account |
| Steps | 1. User clicks on create new document link 2. User is informed of the restriction 3. User is required to click on the create temporary document editing session |
| Post-conditions | 1. User is on the interview portal |
| Exceptions | 1. Connection Error |

|  |  |
| --- | --- |
| Use Case ID | VJF-057  Start whiteboard functionality |
| Description | Allows user to start the whiteboard functionality |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in and is in the interview portal page |
| Steps | 1. Actor clicks on “Start Whiteboard” 2. Use case ends when whiteboard becomes visible to the actor |
| Post-conditions | The user will have either an embedded whiteboard or a whiteboard on a new window |
| Exceptions | 1.   There is a connection error connecting with the whiteboard |

|  |  |
| --- | --- |
| Use Case ID | VJF-058 Draw on whiteboard |
| Description | Allows actor to draw on the whiteboard |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in and currently has the whiteboard functionality open |
| Steps | 1. On the whiteboard screen, the actor clicks on the drawing tool 2. On the blank space in the whiteboard screen, actor clicks and holds the left click of the mouse 3. Actor drags the mouse around the whiteboard screen 4. Use case ends when actor releases the left click |
| Post-conditions | There will be a drawing on the whiteboard |
| Exceptions | None |
| Use Case ID | VJF-059 Change color of drawing tool |
| Description | Allows actor to change color of drawing tool |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in and currently has the whiteboard functionality open |
| Steps | 1. Use case begins when, on the whiteboard screen, the actor clicks on the color palette tool 2. From the options that pop up, the actor selects the color he or she wants to use 3. Use case ends when the new color shows up \_\_\_\_\_\_ |
| Post-conditions | There will be a drawing on the whiteboard |
| Exceptions | 1.   User clicks outside of the color palette, and the color is not changed |

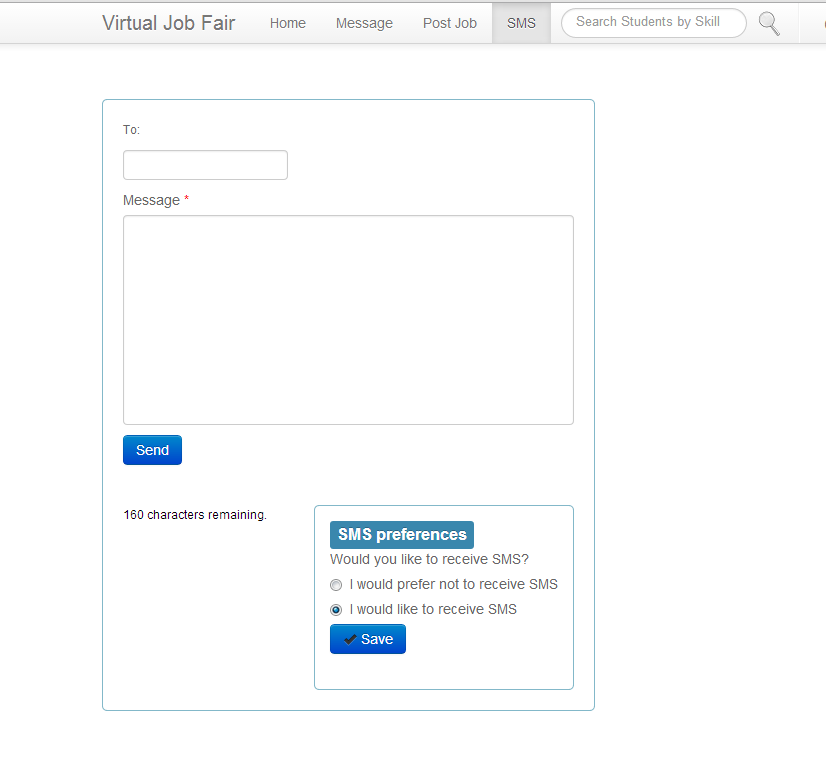
|  |  |
| --- | --- |
| Use Case ID | VJF-060 Type text into whiteboard |
| Description | Allows to type text into the whiteboard |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in, is interviewing and  has the whiteboard functionality open |
| Steps | 1. Use case begins when, on the whiteboard screen, the actor clicks on the “Pencil” option on the left 2. From the sub-menu that pops up, the user shall click on “Text” 3. The user shall click anywhere on the whiteboard screen where he/she wants the text to show 4. The system shall display a window with a textbox, and the user shall type the text that will appear 5. The user shall press “OK” after entering the text |
| Post-conditions | 1. The text the user typed will appear on the screen where the user initially clicked before typing the text |
| Exceptions | 1.   In step 2, if the user clicks on the whiteboard when the sub-menu is showing, the current whiteboard feature selected will be used  2.      In step 4, if the user clicks “Cancel” instead of “OK”, the window will disappear, and not text will be shown |
| Use Case ID | VJF-061 Clear contents of whiteboad |
| Description | Allows actor to clear the current contents of the whiteboard |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in, is interviewing and  has the whiteboard functionality open |
| Steps | 1.      Use case begins when, on the whiteboard screen, the actor clicks on the “Menu” option on the left  2.      From the sub-menu that pops up, the user shall click on “Clear”  3.      The system shall display a pop-up message confirming if the user wants to continue   1. Use case ends when user clicks “OK” on the pop-up |
| Post-conditions | 1.         The whiteboard screen is cleared |
| Exceptions | 1.      In step 4, if the user clicks on “Cancel”, the whiteboard contents will not be cleared, and the whiteboard screen will be shown again |

|  |  |
| --- | --- |
| Use Case ID | VJF-062    Block an user |
| Description | Allows a student to block an employer (and vice versa) |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in and is viewing the profile of the user he/she wants to block |
| Steps | 1. Use case begins when the actor clicks on “Block User” 2. The system shall present a pop-up asking the actor whether he/she really wants to block the other user 3. The actor shall click “Yes” to confirm the blocking 4. Use case ends when system gives confirmation of blocking with a pop-up message |
| Post-conditions | 1. The blocking user will not be found when searched by the user who was blocked  2. Actor will be sent to profile of the actor who was blocked |
| Exceptions | In step 3, if the actor clicks on “No”, then no blocking will be done, and the actor will remain on the same website |

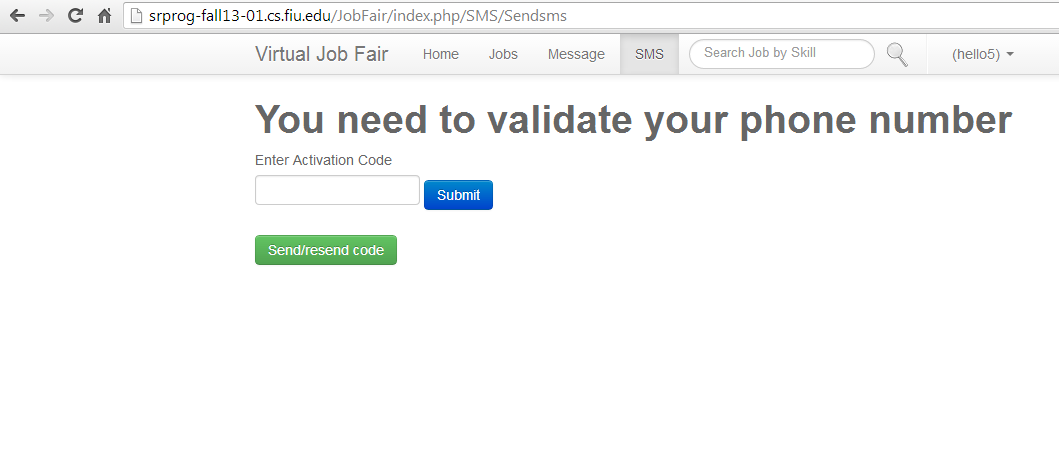
|  |  |
| --- | --- |
| Use Case ID | VJF-063    Unblock an user |
| Description | Allows a student to unblock an employer that was previously blocked (and vice versa) |
| Actor | Employer or student |
| Pre-conditions | Actor is logged in and is viewing the profile of the user he/she wants to unblock |
| Steps | 1.      Use case begins when the actor clicks on “Unblock User”  2.      The system shall present a pop-up asking the actor whether he/she really wants to unblock the other user   1. The actor shall click “Yes” to confirm the blocking 2. Use case ends when system gives confirmation of unblocking with a pop-up message |
| Post-conditions | 1. The blocking user will be able to be found when searched by the user who was blocked  2. Actor will be sent to the profile of the user who has unblocked |
| Exceptions | In step 1, if the user that is being unblocked had not been blocked before, then the actor will not see the “Unblock user” option |

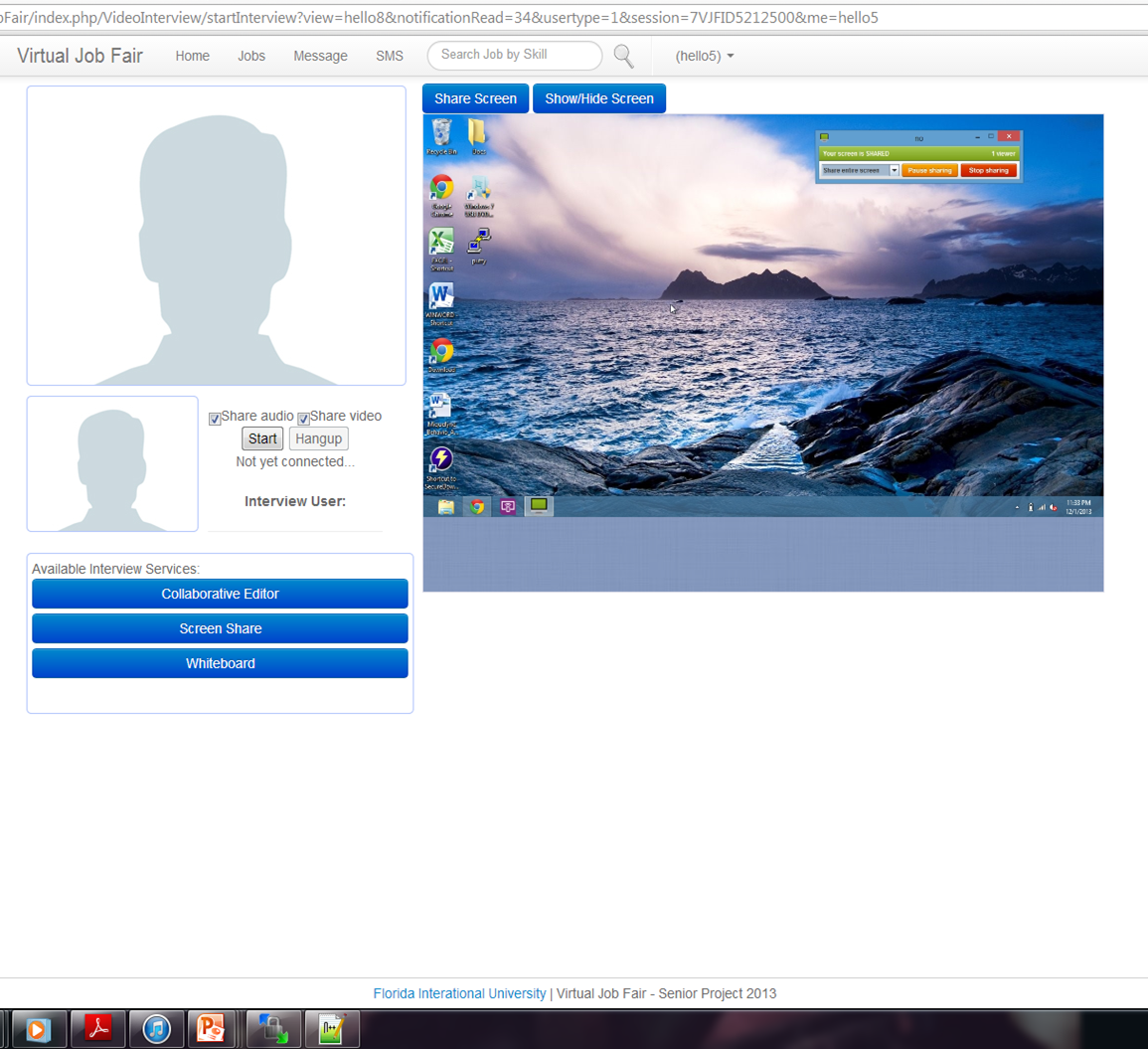
9.3 Appendix C

**Send SMS**

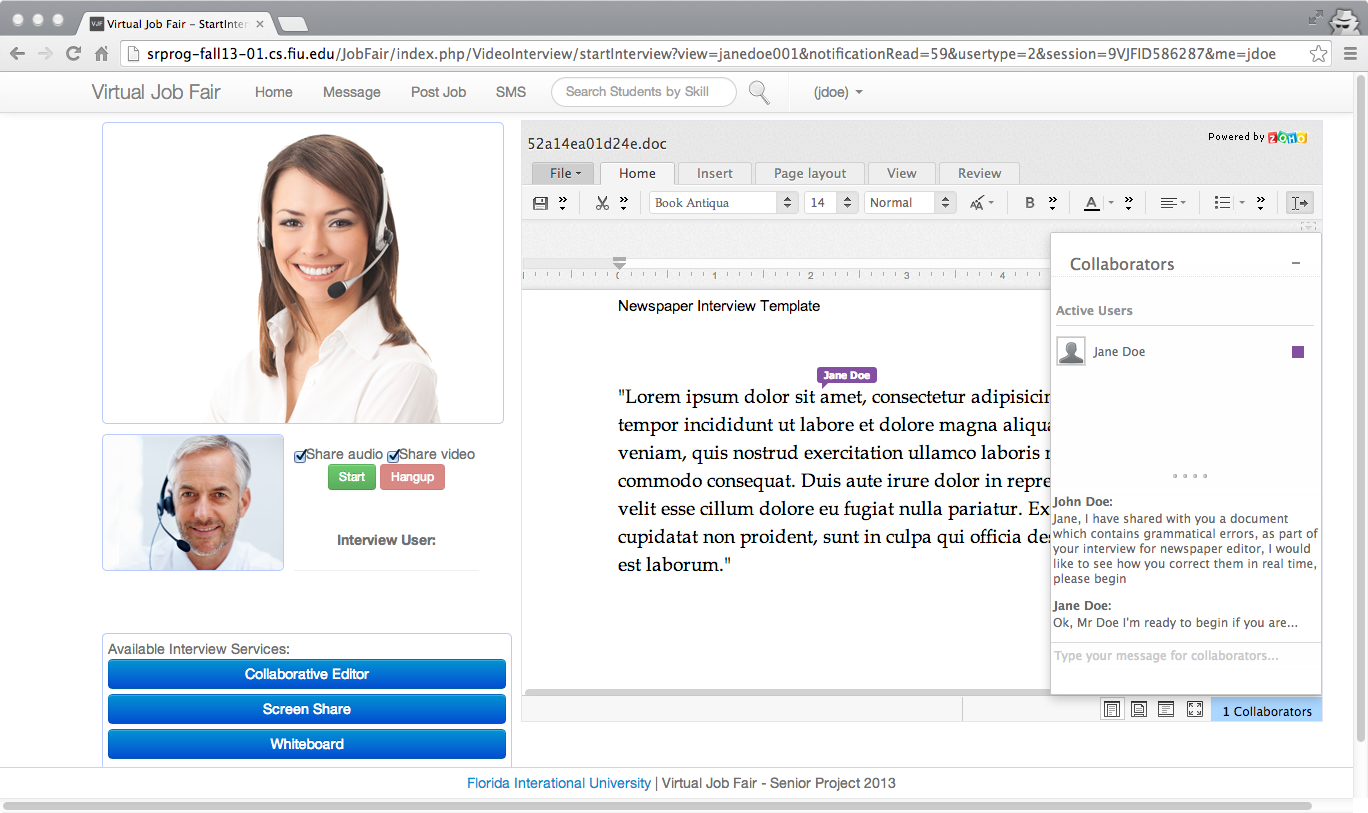
****

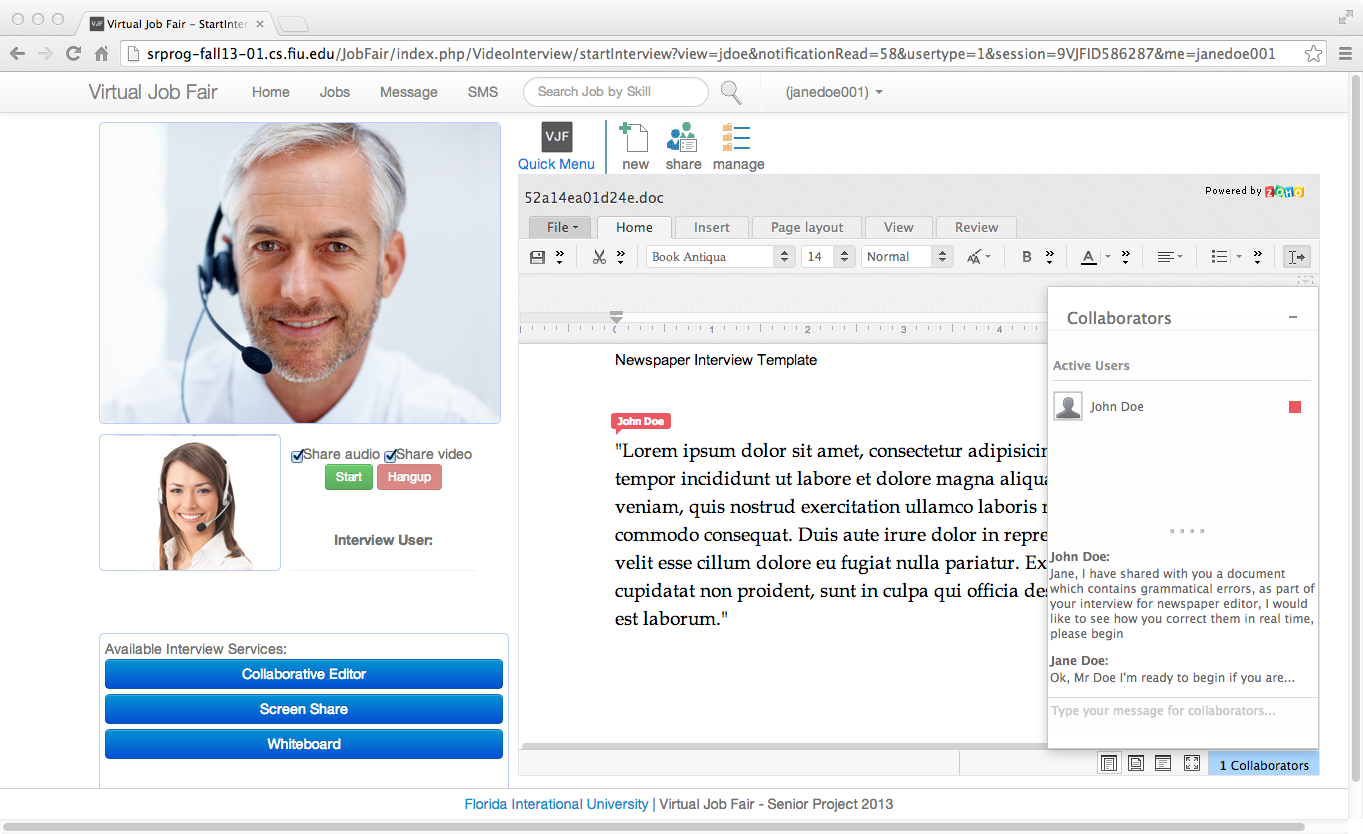
**Validate Phone Number**

****

**Screen Share**

**Collaborative Editor**

****



## Log In Using FIU SCIS Credentials

## F:\Luis B Final Revision UML\login screenshot.jpeg

## Whiteboard\\buffalo.cs.fiu.edu\homes\Desktop\CIS 4911\Deliverable #2 User Interface\Screenshot 1.png

## Image Sharing

## \\buffalo.cs.fiu.edu\homes\Desktop\CIS 4911\Deliverable #2 User Interface\Screenshot 2.png

9.4 Appendix D

**Object Diagrams Analysis**

**Whiteboard**

An object from the Whiteboard class is instantiated when a user first requests to use the whiteboard. The instantiated Whiteboard object has 4 properties. autoJoin is a feature that deals with automatically joining a whiteboard. “toolbar” determines whether the toolbar will be shown on the whiteboard. “smallIcons” decides whether small icons will be forced on the whiteboard. Finally, “orientation” determines the view of the whiteboard (acceptable values are “portrait”, “landscape” and “auto”.

**Document**

An object which represents the Document class gets instantiated upon initiation of the Collaborative Text Editor feature, more specifically, upon the request of any of the features that involve document manipulation. The Document Object describes 10 properties:

- id: unique identifier.

- active\_status: represents if the current document is in an active state.

- document\_id: represents a unique document, it's an internal unique identifier that represents a file.

- local\_user\_id: contains the unique user id of the user who creates a local document.

- remote\_user\_id: contains the unique user id of the user who collaborates with the document remotely.

- owner\_id: which represents which of the users collaborating in the document owns it.

- document\_path: the document path on the filesystem.

- document\_name: represents the document identifier from the point of the user, not the system.

- owner\_url: the url for the Zoho API that accesses and displays a specific document.

-viewer\_url: the url for the Zoho API that permits another user to join a specific document.

**SMS**

An object from the SMS class is instantiated when a SMS message will be sent to a particular user. The object has 6 properties which are useful for any interaction involving a text message. The id property specifies the unique identifier of this text message, the receiver and sender properties specify the identification numbers of the parties involved, the message field contains the content of the SMS, then there is the date field which stores the current date and finally subject which holds the topic of the message.

The other two objects, BasicInfo and VideoInterview, were already explained by the old group.

**Class Diagram Analysis**

The Class Diagram in Appendix C depicts the additional classes that will be added to the original system. It portrays the proposed changes to the scheme, where we build on the implementation of the previous team. The new proposed classes are colored green in order to emphasize the difference between the new and old implementations. In addition, the Diagram follows UML notation in order to facilitate the developers as well as adhere to industry standards.

In our sequence diagrams, actors can be either students, employers, or any. In all of our use cases, the actors communicate with an object from the view component. Also, some sequence diagrams require communication with our database.

* 1. Appendix E

**Whiteboard Subsystem**

The whiteboard subsystem handles the interaction of students and employers through the Virtual Whiteboard. Adding to the already implemented video interview functionality Virtual Job Fair, a whiteboard seems like the next logical step. This whiteboard will allow students to make drawings while interviewing, which is ideal for different majors. The whiteboard will function real-time, meaning that there will be a single whiteboard which will be shared by both employers and students.

**Collaborative Editor Subsystem**

The Collaborative Editor Subsystem handles Interactions between users collaborating on a Document. It is meant to be intuitive, responsive and feature rich. The collaborative editor relies on Scientific Linux, ext3 filesystem, PHP and the Yii Framework to provide a robust Document Server and Document Repository Features. It relies on HTML, CSS, Js, and AJAX to provide a light and responsive editor that can be used by the users to collaborate on a document.

**Screen Share subsytem**

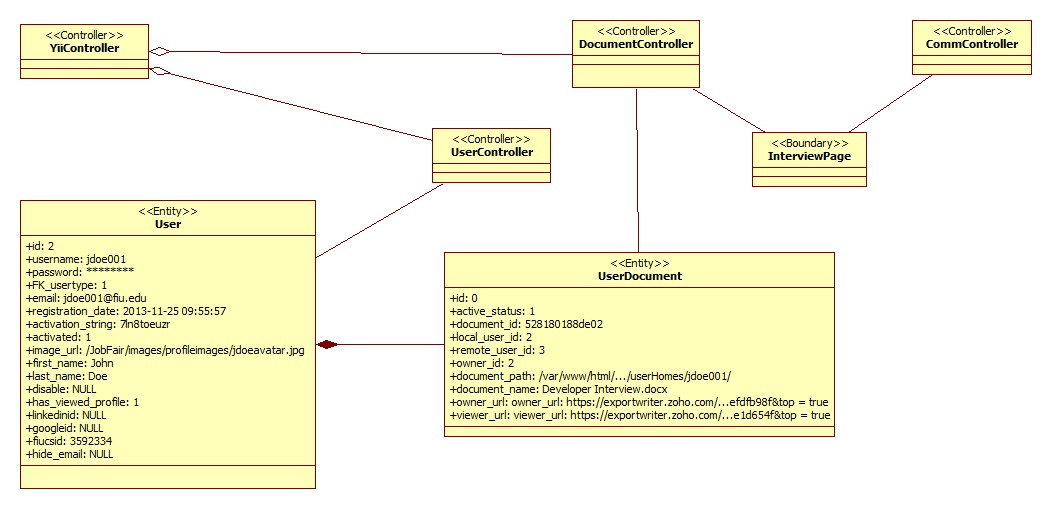
The ScreenShare subsystem handles the display and broadcasting of the user’s screens.  When users request a screen Share session, the system send a request to the Screen Leap Api which returns the state of the request as well as other usable information such as a viewer URL. This URL is stored in order to allow the other user to connect and see what is being displayed.  The screenLeap service makes it possible to provide this feature without having any additional services running in the system.  The ScreenShare .

**Sms subsystem**

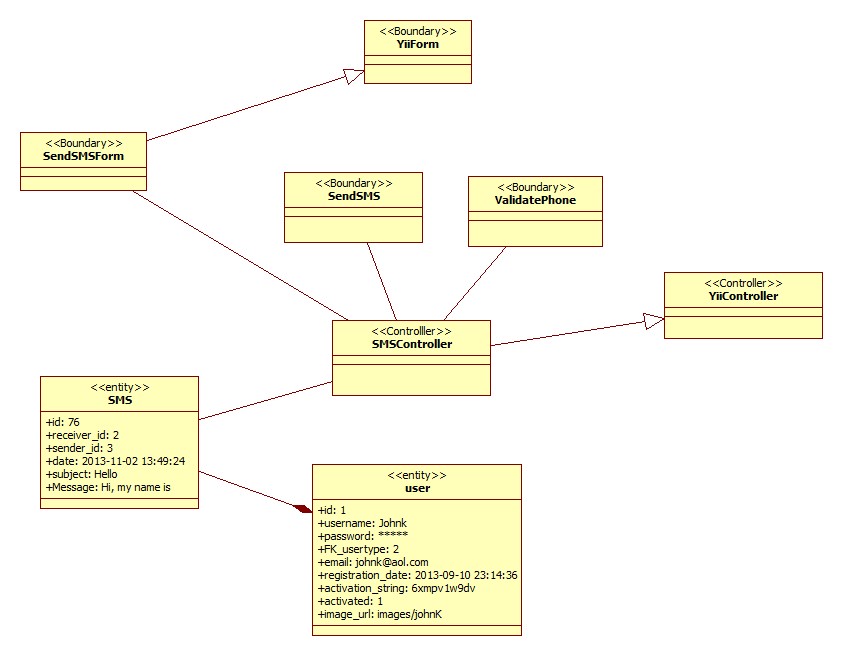
The sms subsystem allows students and employers alike to receive text message notifications about their upcoming interviews. Additionally, employers will be able to send direct text messages  to students they would like to interview with in a short notice.  The Twilio Service provides a very detailed and rich API that allows for these features to be implemented in a way that allows for reliability and security and most importantly ensuring the privacy of our users.

9.5.1 Static model

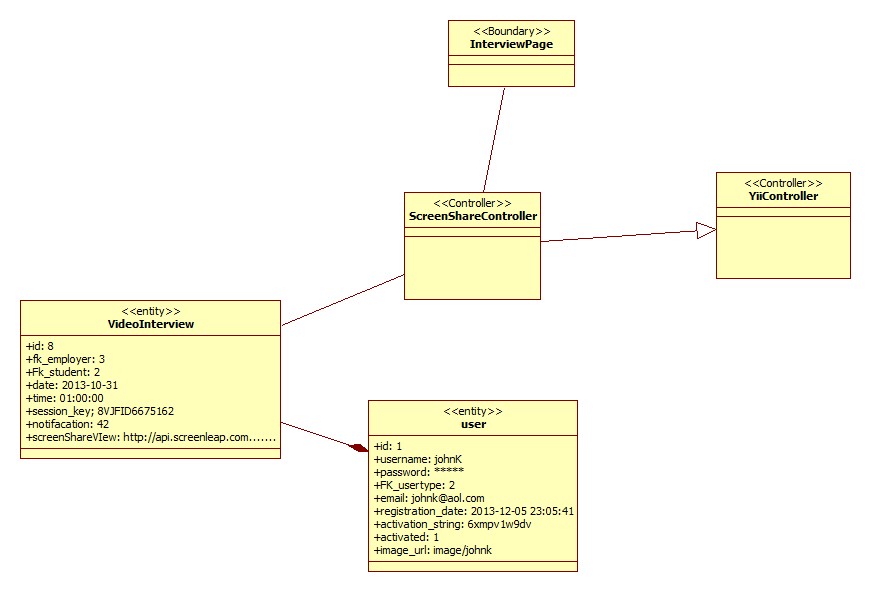
**Collaborative Editor Sub System**



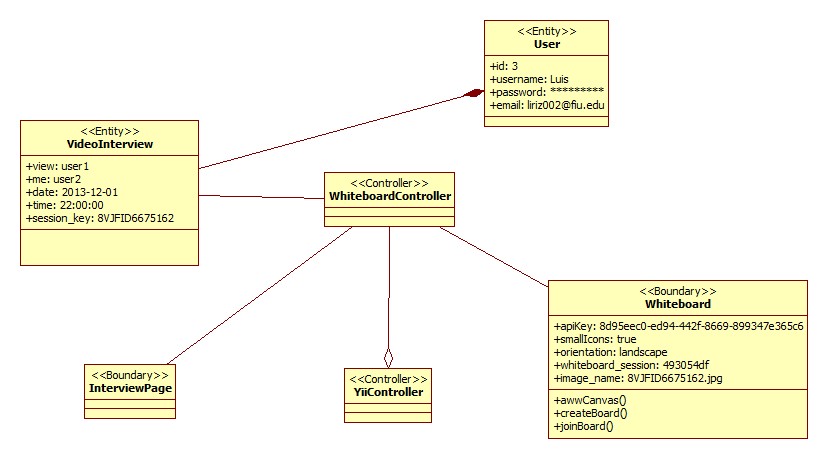
**SMS subsystem**



**Screen Share subsystem**

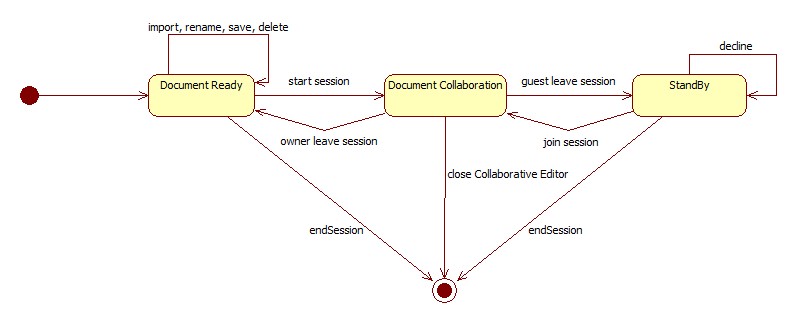


**Whiteboard subsystem**

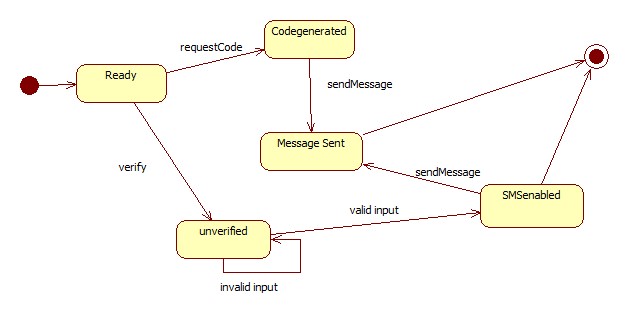


9.5.2 Dynamic model

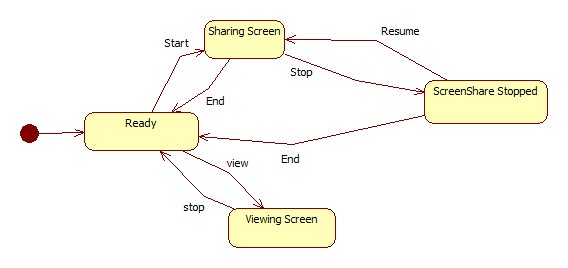
**Collaborative Editor State machine**



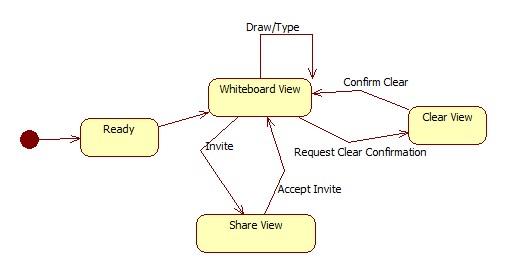
**Sms State machine**



**ScreenShare State machine**



**Whiteboard**



## 9.6 Appendix F.

**Whiteboard**

public function startWhiteboard( );

public function startSharingImage();

public function clearWhiteboard();

public function endWhiteboard();

**Collaborative Editor Subsystem**

private function dbg($msg)

private function createNewEditor($uN, $rN, $dI, $mD, $zDI)

private function openExistingEditor($uN, $rN, $dI, $mD, $zDI)

private function prepareHomeFolder($username, $documentId)

private function saveDb($recordArray)

private function markDocumentsInactive(

public function actionCreateDocument($rU)

public function actionImportDocument()

public function actionRenameDocument($document, $newName)

public function actionSaveDocument()

public function actionDeleteDocument($document)

public function actionShareDocument($documentUrl, $documentName, $remoteUserId)

public function actionListDocument()

public function actionExportDocument()

public function actionOpenDocument($rU, $dI)

* 1. Appendix G

**System Testing**

// DO NOT EDIT: This file is automatically generated each time the script is modified.

// To modify this file either use 'Insert Java Code Snippet'or 'Insert Java Method'

// option from simplified script.

import resources.wrongcodeHelper;

import com.rational.test.ft.\*;

import com.rational.test.ft.object.interfaces.\*;

import com.rational.test.ft.object.interfaces.SAP.\*;

import com.rational.test.ft.object.interfaces.WPF.\*;

import com.rational.test.ft.object.interfaces.dojo.\*;

import com.rational.test.ft.object.interfaces.siebel.\*;

import com.rational.test.ft.object.interfaces.flex.\*;

import com.rational.test.ft.object.interfaces.generichtmlsubdomain.\*;

import com.rational.test.ft.script.\*;

import com.rational.test.ft.value.\*;

import com.rational.test.ft.vp.\*;

import com.ibm.rational.test.ft.object.interfaces.sapwebportal.\*;

// BEGIN custom imports

//TODO: Add custom imports here.

// END custom imports

/\*\*

\* Description : Functional Test Script

\* @author jorge

\*/

public class wrongcode extends wrongcodeHelper

{

/\*\*

\* Script Name : <b>wrongcode</b>

\* Generated : <b>Dec 8, 2013 2:34:11 PM</b>

\* Description : Functional Test Script

\* Original Host : WinNT Version 6.1 Build 7601 (S)

\*

\* @since 2013/12/08

\* @author jorge

\*/

public void testMain(Object[] args)

{

// Group: Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

setSimplifiedScriptLine(1); //Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

timerStart("VirtualJobFairSendsmsSMS\_1");

setSimplifiedScriptLine(2); //Click SMS

link\_sms().click();

setSimplifiedScriptLine(3); //Click BasicInfo[smsCode]

text\_basicInfoSmsCode().click(atPoint(105,18));

setSimplifiedScriptLine(4); //Type Value 4444

browser\_htmlBrowser(document\_virtualJobFairSendsms(),DEFAULT\_FLAGS).inputKeys("{Num4}{Num4}{Num4}{Num4}");

setSimplifiedScriptLine(5); //Click Submit

button\_submit().click();

setSimplifiedScriptLine(6); //Verify Visible Text of errors

html\_errors().performTest(errors\_textVP());

timerStop("VirtualJobFairSendsmsSMS\_1");

}

}

public void testMain(Object[] args)

{

// Group: Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

setSimplifiedScriptLine(1); //Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

timerStart("VirtualJobFairSendsmsSMS\_1");

setSimplifiedScriptLine(2); //Click SMS

link\_sms().click();

setSimplifiedScriptLine(3); //Click username

text\_username().click(atPoint(66,17));

setSimplifiedScriptLine(4); //Type Value markus25

browser\_htmlBrowser(document\_virtualJobFairSendsms(),DEFAULT\_FLAGS).inputKeys("markus{Num2}{Num5}");

setSimplifiedScriptLine(5); //Click SMS[Message]

text\_smsMessage().click(atPoint(23,64));

setSimplifiedScriptLine(6); //Type Value Hello how are you?{TAB}

browser\_htmlBrowser(document\_virtualJobFairSendsms(),DEFAULT\_FLAGS).inputKeys("Hello how are you?{TAB}");

setSimplifiedScriptLine(7); //Click Sendsubmit

button\_sendsubmit().click();

setSimplifiedScriptLine(8); //Verify Visible Text of content

html\_content().performTest(content\_textVP());

timerStop("VirtualJobFairSendsmsSMS\_1");

}

public void testMain(Object[] args)

{

// Group: Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

setSimplifiedScriptLine(1); //Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

timerStart("VirtualJobFairSendsmsSMS\_1");

setSimplifiedScriptLine(2); //Click SMS

link\_sms().click();

setSimplifiedScriptLine(3); //Click username

text\_username().click(atPoint(88,5));

setSimplifiedScriptLine(4); //Type Value from Datapool Column username

browser\_htmlBrowser(document\_virtualJobFairSendsms(),DEFAULT\_FLAGS).inputKeys(dpString("username"));

setSimplifiedScriptLine(5); //Click ui-id-9

link\_uiId9().click();

setSimplifiedScriptLine(6); //Click SMS[Message]

text\_smsMessage().click(atPoint(41,23));

setSimplifiedScriptLine(7); //Type Value from Datapool Column SMSMessage

browser\_htmlBrowser(document\_virtualJobFairSendsms(),DEFAULT\_FLAGS).inputChars(dpString("SMSMessage"));

setSimplifiedScriptLine(8); //Click Sendsubmit

button\_sendsubmit().click();

setSimplifiedScriptLine(9); //Verify Visible Text of errors

html\_errors().performTest(errors\_textVP());

timerStop("VirtualJobFairSendsmsSMS\_1");

}

public void testMain(Object[] args)

{

// Group: Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

setSimplifiedScriptLine(1); //Virtual Job Fair - Sendsms SMS: http://srprog-fall13-01.cs.fiu.edu/JobFair/index.php/SMS/Sendsms

timerStart("VirtualJobFairSendsmsSMS\_1");

setSimplifiedScriptLine(2); //Click SMS

link\_sms().click();

setSimplifiedScriptLine(3); //Click BasicInfo[smsCode]

text\_basicInfoSmsCode().click(atPoint(102,15));

setSimplifiedScriptLine(4); //Type Value 2000{TAB}

browser\_htmlBrowser(document\_virtualJobFairSendsms(),DEFAULT\_FLAGS).inputKeys("{Num2}{Num0}{Num0}{Num0}{TAB}");

setSimplifiedScriptLine(5); //Click Virtual Job Fair - Sendsms SMS

document\_virtualJobFairSendsms().click(atPoint(603,247));

setSimplifiedScriptLine(6); //Click Submit

button\_submit().click();

timerStop("VirtualJobFairSendsmsSMS\_1");

// Group: HtmlDialog

setSimplifiedScriptLine(7); //HtmlDialog

timerStart("HtmlDialog\_7");

setSimplifiedScriptLine(8); //Verify Properties of HtmlDialog

html\_htmlDialog().performTest(HtmlDialog\_standardVP());

setSimplifiedScriptLine(9); //Click Html.DialogButtonOK

button\_htmlDialogButtonOK().click();

setSimplifiedScriptLine(10); //Verify Image RootTestObject.getScreenTestObject

RootTestObject.getScreenTestObject().performTest(Screen\_imageVP());

timerStop("HtmlDialog\_7");

}

**Subsystem Testing**

*<?php*

*class SmsTest extends CDbTestCase {*

*public static function setUpBeforeClass() {*

*$pass = dirname ( \_\_FILE\_\_ ) . '/../../../PasswordHash.php';*

*require\_once ($pass);*

*}*

*protected function setUp() {*

*$this->getFixtureManager ()->basePath = Yii::getPathOfAlias ( 'application.tests.fixtures.\*' );*

*parent::setUp ();*

*// This will override the regenerateID method. remove headers error*

*$mockSession = $this->getMock ( 'CHttpSession', array (*

*'regenerateID'*

*) );*

*Yii::app ()->setComponent ( 'session', $mockSession );*

*// needed by controller*

*$identity = new UserIdentity ( 'ncapo006', '999999' );*

*$identity->authenticate ();*

*Yii::app ()->user->login ( $identity );*

*Yii::import ( 'application.controllers.\*' );*

*require\_once Yii::app()->basePath."/fiucsauth/FiuCsAuth.php";*

*global $csCon = new ProfileController("31416");*

*}*

*// test login using FIU SCIS Credentials*

*public function testLogin() {*

*global $csCon;*

*$csCon->actionSendcode ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$this->assertGreaterThanOrEqual ( 1000, $info->fiucsid );*

*}*

*// test user is not linked with another account*

*public function testNoDuplicateUser() {*

*global $csCon;*

*$\_POST ['panthermail'] = 'ncapo006';*

*$\_POST ['pantherid'] = "999999";*

*$this->expectOutputString ( "User email is already linked with another account. <br />" );*

*$this->actionFiuCsSeniorAuth();*

*}*

*// test*

*public function testUserActivated() {*

*global $csCon;*

*$csCon->actionSendcode ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$this->assertGreaterThanOrEqual ( 1, $info->activated );*

*}*

*}*

**SMS**

*<?php*

*class SmsTest extends CDbTestCase {*

*public static function setUpBeforeClass() {*

*$pass = dirname ( \_\_FILE\_\_ ) . '/../../../PasswordHash.php';*

*require\_once ($pass);*

*}*

*protected function setUp() {*

*$this->getFixtureManager ()->basePath = Yii::getPathOfAlias ( 'application.tests.fixtures.\*' );*

*parent::setUp ();*

*// This will override the regenerateID method. remove headers error*

*$mockSession = $this->getMock ( 'CHttpSession', array (*

*'regenerateID'*

*) );*

*Yii::app ()->setComponent ( 'session', $mockSession );*

*// needed by controller*

*$identity = new UserIdentity ( 'test', 'hello5' );*

*$identity->authenticate ();*

*Yii::app ()->user->login ( $identity );*

*Yii::import ( 'application.controllers.\*' );*

*global $smscon;*

*$smscon = new SMSController ( "1" );*

*// ///////////////////////////////////////////////////////////////////////////////////*

*}*

*// test sendcode action*

*public function testConfirm() {*

*global $smscon;*

*$smscon->actionSendcode ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$this->assertGreaterThanOrEqual ( 1000, $info->smsCode );*

*$this->assertlessThanOrEqual ( 9999, $info->smsCode );*

*}*

*public function testAutoComplete() {*

*global $smscon;*

*$\_GET ['term'] = 'te';*

*$this->expectOutputString ( '["test"]' );*

*$smscon->actionGetAutoComplete ();*

*}*

*public function testAutoCompleteBlank() {*

*global $smscon;*

*$\_GET ['term'] = 'xxxxxxxxxx';*

*$this->expectOutputString ( null );*

*$smscon->actionGetAutoComplete ();*

*}*

*public function testverifyBlankName() {*

*global $smscon;*

*$\_POST ['username'] = 'john';*

*$\_POST ['SMS'] ['Message'] = "hello";*

*$this->expectOutputString ( "User does not exist. <br />" );*

*$smscon->actionVerify ();*

*}*

*public function testverifyNoSms() {*

*global $smscon;*

*$\_POST ['username'] = 'Peter';*

*$\_POST ['SMS'] ['Message'] = "";*

*$this->expectOutputString ( "Please enter a message to be sent. No blank messages allowed. <br />User has not set up the SMS functionality. <br />" );*

*$smscon->actionVerify ();*

*}*

*public function testverifyLongSMS() {*

*global $smscon;*

*$\_POST ['username'] = 'mike';*

*$\_POST ['SMS'] ['Message'] = "A wall of text is something that is frowned upon in most, actually virtually all Internet societies, including forums, chat boards, and Uncyclopedia. You should not make walls of text because it can get you banned anywhere unless it is a place that encourages walls of text. I highly doubt any place does support something so irritating and annoying, but anything can exist, but not really because unless you are in heaven then that can happen. But no one actually knows that was just a hypothesis, a lame one that is. Actually not really lame. Y, but you would be hated if you do that,";$this->expectOutputString ( "The message is too big. Please create a message with less than 160 characters <br />User does not have a phone number in record. <br />User has not set up the SMS functionality. <br />" );*

*$smscon->actionVerify ();*

*}*

*//No input is entered*

*public function testValidationNoInput() {*

*global $smscon;*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*//Post variable is set for controller*

*$\_POST ['BasicInfo'] ['smsCode'] = null;*

*//Expected Output that should be sent to the user*

*$this->expectOutputString ( "Please enter a code" );*

*$smscon->actionValidation ();*

*// tries not incremented*

*$this->assertTrue ( $info->tries == 0 );*

*}*

*public function testValidationWrongInput() {*

*global $smscon;*

*$\_POST ['BasicInfo'] ['smsCode'] = '23';*

*$smscon->actionValidation ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$this->assertEquals ( 1, $info->tries );*

*// second*

*$smscon->actionValidation ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$this->assertEquals ( 2, $info->tries );*

*// third*

*$smscon->actionValidation ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$this->assertEquals ( 3, $info->tries );*

*$this->expectOutputString ( "Invalid codeInvalid codeInvalid code" );*

*$smscon->actionValidation ();*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$this->assertEquals ( 4, $info->tries );*

*$this->expectOutputString ( "Invalid codeInvalid codeInvalid codeInvalid code" );*

*}*

*public function testValtooMany() {*

*global $smscon;*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$\_POST ['BasicInfo'] ['smsCode'] = 2;*

*$this->expectOutputString ( "Too many wrong inputs, please contact Hola@aol.com\n" );*

*$smscon->actionValidation ();*

*$this->assertTrue ( $info->tries == 4 ); // tries not incremented*

*}*

*public function testvalidationreset() {*

*Yii::app ()->user->logout (); // logout and reloging*

*$mockSession = $this->getMock ( 'CHttpSession', array (*

*'regenerateID'*

*) );*

*Yii::app ()->setComponent ( 'session', $mockSession );*

*// needed by controller*

*$identity = new UserIdentity ( 'Peter', 'hello5' );*

*$identity->authenticate ();*

*Yii::app ()->user->login ( $identity );*

*Yii::import ( 'application.controllers.\*' );*

*global $smscon;*

*$smscon = new SMSController ( "1" );*

*// ///////////////////////////////////////////////////////////////////////////////////*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$\_POST ['BasicInfo'] ['smsCode'] = 9555;*

*$this->assertTrue ( $info->tries == 3 ); // before*

*$smscon->actionValidation ();*

*// after*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$this->assertTrue ( $info->tries == 0 );*

*// tries reset*

*$this->assertTrue ( $info->validated == 1 );*

*}*

*public function testchangepref() {*

*echo dirname ( Yii::app ()->getRequest ()->getScriptFile () ) . DIRECTORY\_SEPARATOR;*

*global $smscon;*

*$\_POST ['BasicInfo'] ['allowSMS'] = 0;*

*$username = Yii::app ()->user->name;*

*$model = User::model ()->find ( "username=:username", array (*

*':username' => $username*

*) );*

*$info = BasicInfo::model ()->find ( "userid=:userid", array (*

*'userid' => $model->id*

*) );*

*$smscon->actionChangeSMSpref ();*

*// $this->assertTrue ( $info->allowSMS == 1);*

*}*

*}*

## 9.7 Appendix H

**Project:** Virtual Job Fair

**Date:** Friday, September 6th, 2013

**Start Time:** 7:00PM

**End Time:** 10:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:** Start creating an overview of the project and call our mentor, Mr. Caraballo, to discuss specific issues about the project

Assigned Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Set up Yii framework to review the spring 2013 project | Create a list with information and to-do items based on Mr. Caraballo’s questions; review the documentation of the old project | Work on the definitions, acronyms, and abbreviations section of the Deliverable | Called Mr. Caraballo and asked project-specific questions |

**Project:** Virtual Job Fair

**Date:** Saturday, September 7th, 2013

**Start Time:** 9:00AM

**End Time:** 2:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:** Based on brainstorming and information gathered from Mr. Caraballo, clearly define the purpose of the new system when compared to the current system. Also, keep working on the first deliverable, which is due on Monday, September 9th, 2013.

Assigned Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Watch tutorials on the Yii framework; statically analyze the old group’s project to better understand that group’s implementation | Put the agreed-upon problem definition into words and complete the high-level requirements | Work on the diary of meetings and research free tools to create GANTT charts | Keep working on Deliverable #1 as a group. Also, define specific roles for each member of the group |

**Project:** Virtual Job Fair

**Date:** Sunday, September 8th, 2013

**Start Time:** 10:30AM

**End Time:** 7:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:** Finish setting up the framework in which to run the old group’s project, complete the Feasibility Study and Plan, and do the PowerPoint presentation for tomorrow

Assigned Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Check the old project’s code thoroughly and change absolute paths to match that of our framework; work on the cost matrix; create GANTT chart with project schedule information | Determine a rating scale and assign values to alternatives for the feasibility matrix; list the tasks, milestones, and deliverables to be used as reference for the GANTT chart | Introduce each chapter briefly; format the document for it to have a consistent look and feel; detail hardware and software requirements | Finish the first deliverable and brainstorm for the presentations that are due on Monday, September 10th, 2013 |

**Project:** Virtual Job Fair

**Date:** Sunday, September 10th, 2013

**Start Time:** 8:00PM

**End Time:** 10:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:** Set up the virtual machine that was provided by SCIS and install everything all the software that will be needed

Assigned Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Research different software packages and APIs for integration into the project | Set up the LAMP environment that will be used for development and testing in the virtual machine | Create a document that contains the proposed functionality of the system to discuss with our mentor and our instructor | Set up the virtual machine as soon as possible in order to start developing and adding features |

**Project:** Virtual Job Fair

**Date:** Wednesday, September 11th, 2013

**Start Time:** 7:00PM

**End Time:** 11:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Review the existing code in order to get the video interview system working | Replace old paths from existing code to point our current virtual machine for interview process | Review the existing code in order to get the video interview system working | Set up the video interview system from the old’s project so it works on our virtual machine |

**Project:** Virtual Job Fair

**Date:** Saturday, September 14th, 2013

**Start Time:** 2:00PM

**End Time:** 4:30PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Plan GANTT chart for the whole project based on new schedule | Lead the selection of features of the system that will be implemented | Modify current system section of Deliverable #1 to reflect the project done in spring 2013 | Advance on Deliverable #1 |

**Project:** Virtual Job Fair

**Date:** Sunday, September 15th, 2013

**Start Time:** 2:00PM

**End Time:** 6:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Redo individual GANTT charts for the presentation, according to a new schedule | Formalize the features of the new system | Edit the features of the new system and add them to the document, each with a description | Finish Deliverable #1 and individual presentations for resubmission |

**Project:** Virtual Job Fair

**Date:** Tuesday, September 17th, 2013

**Start Time:** 7:30PM

**End Time:** 10:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Perform research on EasyRTC’s API and start integrating it into project | Perform research on how to be able to allow users to have a collaborative text editor available | Email 8 companies regarding the use of their whiteboard applications and correspondent API; do more research on how to implement the whiteboard functionality | Finish Deliverable #1 and individual presentations for resubmission |

**Project:** Virtual Job Fair

**Date:** Friday, September 20th, 2013

**Start Time:** 6:30PM

**End Time:** 10:30PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Develop screen-sharing use cases | Development collaborative text editor use cases and do research on feasibility of using Google API for the text editor | Develop whiteboard use cases based on the requirements analysis | Work on use cases; start working on sequence diagrams for these use cases |

**Project:** Virtual Job Fair

**Date:** Sunday, September 22nd, 2013

**Start Time:** 12:00PM

**End Time:** 9:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Develop class diagram by modifying last class diagram done by the old project’s members | Work on the use cases assuming the Google API will be used for the collaborative text editor | Work on functional/non-functional requirements for the system | Finish sequence diagrams for all of the use cases |

**Project:** Virtual Job Fair

**Date:** Monday, September 23rd, 2013

**Start Time:** 12:00PM

**End Time:** 9:00PM

**In Attendance:** Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:** None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Write summaries of use case and static diagrams; develop the use case diagram in UML | Organize functional requirements appropriately and check overall flow of document before turning in | Edit document extensively and for revision after merging everyone’s parts | Finish Deliverable #2 to turn it in; complete presentations in order to possibly present on Monday, September 23rd, 2013 |

**Project:**Virtual Job Fair

**Date:**Thursday, October 3rd, 2013

**Start Time:** 7:00PM

**End Time:**9:00PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Verified persistent data store relationships, system integration, outlined general feature functionality | Worked on use case diagrams and state machine diagrams, revised ER diagram for correctness | Edit document extensively and for revision after merging everyone’s parts | Worked on diagrams and designed general functionality of features implemented |

**Project:**Virtual Job Fair

**Date:**Friday, October 6th, 2013

**Start Time:** 7:30PM

**End Time:**10:00PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Refined persistent data store relationships, specific subsystem description | Refined diagrams, refined persistent data storage tables worked on specific subsystem description | Edit document extensively and for revision after merging everyone’s parts specific subsystem description | Work on description of subsystems |

**Project:**Virtual Job Fair

**Date:**Sunday, October 13, 2013

**Start Time:** 8:30PM

**End Time:**10:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Begin work on deliverable 4 | Begin work on deliverable 4 | Begin work on deliverable 4 | Begin work on deliverable 4 |

**Project:**Virtual Job Fair

**Date:**Sunday, October 27, 2013

**Start Time:** 5:00PM

**End Time:**8:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Fixed some bugs on SMS subsystem | Working on document user interface | Debugging Whiteboard application | Continue work on deliverable 4 |

**Project:**Virtual Job Fair

**Date:**Thursday, November 7, 2013

**Start Time:** 11:00AM

**End Time:**1:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Working on SMS front end functionality | Debugging | Contact with Whiteboard creator | Test case design |

**Project:**Virtual Job Fair

**Date:**Wednesday, November 20, 2013

**Start Time:** 5:00pM

**End Time:**7:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Edit screenShare implementation | Test tool research | Selenium research | Test case design |

**Project:**Virtual Job Fair

**Date:**Wednesday, November 20, 2013

**Start Time:** 5:00pM

**End Time:**7:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Edit screenShare implementation | Test tool research | Selenium research | Test case design |

**Project:**Virtual Job Fair

**Date:**Wednesday, 4 December , 2013

**Start Time:** 5:00pM

**End Time:**7:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Edit sections in deliverable 4 | Edit sequence Diagrams | New whiteboard tool research | Deliverable 4 implementation |

**Project:**Virtual Job Fair

**Date:**Friday, 6 December , 2013

**Start Time:** 7:00pM

**End Time:**10:30PM

**In Attendance:**Jorge Fernandez, Luis Benjumea, Luis Irizarry

**Late:**None

**Agenda:**

Assigned Tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Jorge F.** | **Luis B.** | **Luis I.** | **Group** |
| Work on deliverable 4 | Work on deliverable 4 | Work on deliverable 4 | Work on deliverable 4 |

## 10 References