*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 - Senior Capstone Project

Software Engineering Focus

Final Deliverable

Project Title: Go Local Staff 2.0

**Team Members:**

Stephenson Petit-Homme

Daniel Gonzalez

**Product Owner(s)**:

Eduardo Garcia

**Mentor(s)**:

Francisco Ortega

**Instructor**: Masoud Sadjadi

The MIT License (MIT)

Copyright (c) *2016 Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

***Abstract***

*This document presents the information necessary to gain a good understanding of the Go Local App 2.0 project. The Go Local App is an iPhone application that connects employers looking to host events with staff willing to work at those events. In the following pages, user stories, diagrams and information will be presented that will describe the project in detail.*

**Table of Contents**

**Introduction** …………………………………………………………………………………………………………..……………….. 5

Current System ……………………………………………………………………………………………………...……………... 5

Purpose of New System ……………………………………………..…………………………………………………………... 5

**User Stories**

Implemented User Stories …………………………………………………………………………………………………….. 7

Pending User Stories ………………………………………………………………………………...……………………..….. 10

**Project Plan**

Hardware and Software Resources …………………………………………………………………………….…….… 12

Sprints Plan …………………………………………………………….....………………………………………………………. 13

*Sprint 1*  …………………………………………………………………………………………………………………………... 13

*Sprint 2*  …………………………………………………………………………………………………………………………... 13

*Sprint 3*  …………………………………………………………………………………………………………………………... 14

*Sprint 4*  …………………………………………………………………………………………………………………………... 15

*Sprint 5*  …………………………………………………………………………………………………………………………... 16

*Sprint 6*  …………………………………………………………………………………………………………………………... 17

*Sprint 7*  …………………………………………………………………………………………………………………………... 18

**System Design** ………………………………………………………………………………………………………………………... 21

Architectural Patterns …………………………………………………………………………………………………….. 21

System and Subsystem Decomposition …………………………………………………………………………………. 22

Deployment Diagram …………………………………………………………….....…………………………………….…... 23

Design Patterns ………………………………………………………………………………...………………………….….... 24

**System Validation**  ………………………………………………………………………………...……………………………….25

**Glossary**  …………………………………………………………………………………………….....……………………………….40

**Appendix**  …………………………………………………………………………………………….………………………………….41

Appendix A - UML Diagrams …………………………………………………………………………………………………. 41

*Static UML Diagrams*  ……………………………………………………………………………………………………….41

*Dynamic UML Diagrams*  …………………………………………………………………………………………………..43

Appendix B - User Interface Design ………………………………………………………………………………….…... 56

Appendix C - Sprint Review Reports ………………………………………………………………………………...…… 73

Appendix D - Sprint Retrospective Reports …………………………………………………………………….……. 78

**References** …………………………………………………………………………………………………………………...………...84

# Introduction

In order to establish the scope and relevance of the project, the current system and the proposed system will be described. This section is meant to provide the reader with a brief overview of the problem that this application is meant to address. The sections following this one will show the users stories worked on, the project planning that done, the system design that was laid out, and the validation tests that were carried out.

## 

## Current System

Finding and managing the staff required to host events is a difficult and resource intensive process. Employers have to go through staffing agencies to find the staff they need, and incur high overhead costs as a result.

Staffing agencies provide access to their collection of available talent. However, this collection is limited, often resulting in the need to visit several of these agencies in order to execute an event. Communication and management are also bottlenecked by having to consult with these middlemen in order to successfully host an event. The employer and staffing agency interaction can only take place during the agency's hours of operation.

The process of connecting talent to employers involves the use of email, phone and fax in order to exchange the necessary information. The staffing agencies compile all of the information and communication into their computer systems and use said systems to manage the entire process.

## Purpose of New System

Go Local Staff is an iPhone application that aims to eliminate the need for staffing agencies by connecting employers directly to their staff. The app allows employer users to search for, employ, and manage staff. The app also allows staff users to search for and apply to jobs that have been posted by employers.

This system removes the bottlenecks of staffing agencies. Users can carry out their desired actions on their own time, inside the app without the need for email, fax, or other software. The process of connecting employers to staff is handled by the app itself. User only need to search for the staff or jobs they are interested in and the app will provide the necessary tools and services to fulfill the user’s needs.

# User Stories

The following section provides the detailed user stories that were implemented in this iteration of the Go Local App. These user stories served as the basis for the implementation of the project’s features. This section also shows the user stories that are to be considered for future development.

## Implemented User Stories

**User Story #803 - Create a job**

## As a user, I need to be able to post a job offering from my employer account so that staff users can see it.

Acceptance Criteria:

1. Allow user to enter job information.
2. Create the job on submission of job information.

**User Story #805 - Staff - View Jobs**

As a user, I need to be able to view the jobs that employers have created from my staff account so that I can see what jobs I can apply to.

Acceptance Criteria:

1. View created jobs.
2. View detailed info on desired jobs.

**User Story #806 - Edit Created Jobs**

As a user, I need to be able to edit the jobs I have posted from my employer account so that I can make changes to the job if I desire.

Acceptance Criteria:

1. Be able to edit jobs from the detailed job view.
2. Save edits and update the database to reflect the edits.

**User Story #807 - Staff & Employer Registration**

As an employer or employee, I want to be able to register so that I can post or apply for jobs.

Acceptance Criteria:

1. Create a registration form for both staff and employers
2. View user information
3. Verify user email address

**User Story #808 - Create A Profile**

As a user, I want to be able to set up a profile so that I can post or apply for jobs.

Acceptance Criteria:

1. Create a profile page for the staff user
2. View Staff information

**User Story #813 - Create Account Management page**

As a user, I want to be able to set up a profile so that I can post or apply for jobs.

Acceptance Criteria:

1. User locations are visible.
2. User credentials (Name,Age,Sex,Nationality) are visible.
3. Users can be deleted.

**User Story #817 - Employer - View Jobs**

As an employer user, I need to be able to view the jobs I have posted from my employer account so that I can see what jobs I have posted.

Acceptance Criteria:

1. View created jobs.
2. View detailed info on desired jobs.

**User Story #828 - Accept A Job**

As a staff user, I need to be able to submit a job application that can then be accepted or rejected by an employer, so that I can be hired for a job.

Acceptance Criteria:

1. Be able to apply for a job as a staff.
2. Be able to receive notifications of job applications as an employer.
3. Be able to accept/reject applications as an employer.
4. Be able to see which staff is assigned to a job as an employer.

**User Story #829 - Post a Job (Website)**

As an employer, I need to be able to post jobs, so that staff can apply for them.

Acceptance Criteria:

1. Allow user to enter job information.
2. Create the job on submission of job information.

**User Story #834 - Notify Users of New Jobs**

As a staff, I need to get notified whenever a new job is posted, so that I can apply for it.

Acceptance Criteria:

1. Notify user when new job is posted.
2. Send email notification.

**User Story #837 - View Assigned Staff**

## As an employer user, I need to be able to view and remove staff that are assigned to my jobs so that I can manage my staff.

Acceptance Criteria:

1. View staff assigned to specific jobs.
2. View detailed information on that staff.
3. Be able to remove specific staff.

**User Story #838 - Post A Resume**

As a staff, I need to upload my resume, so that employers can see it

Acceptance Criteria:

1. Be able to select a resume for uploading.
2. Have the system upload the resume.

**User Story #839 - View Resume**

As a staff, I need to be able to see my resume, so that I can decide if I want to upload a new one.

Acceptance Criteria:

1. View uploaded resume.

## 

## Pending User Stories

**User Story #843 - Filter A Job**

As a staff user, I need to be able to filter the list of available jobs based on a search criteria so that I can better assess which job is best for me.

Acceptance Criteria:

1. Be able to enter a search criteria.
2. Have the list of jobs narrowed down to match that search criteria.

# Project Plan

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plannings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

## Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

**Digital Ocean**

Digital Ocean is the provider used to host the website and server containing the database used in the application. Their service was chosen because of the compelling price and effective management tools.

**Xcode**

Xcode 7 was used as the IDE in which the iPhone application was developed. Xcode was chosen because it is developed and supported by Apple and is the go-to solution for iPhone development. Xcode also bring with it an iPhone simulator that simplified the validation and testing phases.

**OS X El Capitan**

The OS X El Capitan operating system was chosen because it is the only system where Xcode 7 can be used.

**Mingle**

Mingle was used as a planning and management tool for the various agile development processes.

**MySQL**

MySQL was chosen as the relational database because of the open source nature and the group’s familiarity with it.

**PHP**

PHP was used as the server-side scripting language for the web development. PHP was chosen because of it’s flexibility with HTML.

**Phpmyadmin**

Phpmyadmin was used to manage the MySQL databases.

**Github**

Github was used to store and manage the source code.

**Gmail**

Gmail was used for communication.

**Google Drive**

Google Drive was used to store project documents and to transfer data between the group members.

## 

## Sprints Plan

### Sprint 1

(01/16/2016 - 01/29/2016)

**User Story #807 - Staff & Employer Registration**

***Tasks***

* Create the registration forms.
* Save information and create an account.
* Implement email verification.

***Acceptance Criteria***

1. Create a registration form for both staff and employers
2. View user information
3. Verify user email address

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 2

(01/30/2016 - 02/12/2016)

**User Story #803 - Create A Job**

***Tasks***

* Add button with navigation for creating a job in the employer home screen.
* Create a form for the user to enter job information.
* Create and upload the job on form submission.

***Acceptance Criteria***

1. Allow user to enter job information.
2. Create the job on submission of job information.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #808 - Create A Profile**

***Tasks***

* Create the profile page for staff users.
* Populate the page with the staff user’s information.

***Acceptance Criteria***

1. Create a profile page for the staff user
2. View Staff information

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 3

(02/13/2016 - 02/26/2016)

**User Story #817 - Employer - View Jobs**

***Tasks***

* Add button with navigation for viewing created jobs in the employer home screen.
* Implement a list of jobs created by the employer.
* Implement detailed job information.

***Acceptance Criteria***

1. View created jobs.
2. View detailed info on desired jobs.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 4

(02/27/2016 - 03/11/2016)

**User Story #805 - Staff - View Jobs**

***Tasks***

* Add button with navigation for viewing available jobs in the staff home screen.
* Implement a list of jobs available to the staff.
* Implement detailed job information.

***Acceptance Criteria***

1. View created jobs.
2. View detailed info on desired jobs.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #813 - Create Account Management page**

***Tasks***

* Create management page.
* Show user information.
* Implement deletion of users.

***Acceptance Criteria***

1. User locations are visible.
2. User credentials(Name,Age,Sex,Nationality) are visible.
3. Users can be deleted.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #811 - Search for Jobs**

***Tasks***

* Create a page to search for jobs.
* Implement search based on different criteria.

***Acceptance Criteria***

1. Search by locations
2. Search by radius
3. Search by date

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #806 - Employer - Edit Created Jobs**

***Tasks***

* Create edit button in the detailed job view.
* Create the edit form.
* Implement saving of edits on form submission.

***Acceptance Criteria***

1. Be able to edit jobs from the detailed job view.
2. Save edits and update the database to reflect the edits.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 5

(03/19/2016 - 04/01/2016)

**User Story #829 - Post a Job (Website)**

***Tasks***

* Create page for job creation.
* Create job form.
* Implement creation of a job on form submission.

***Acceptance Criteria***

1. Allow user to enter job information.
2. Create the job on submission of job information.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #843 - Filter Available Jobs**

***Tasks***

* Create a search bar above the jobs list.
* Implement searching functionality.

***Acceptance Criteria***

1. Be able to enter a search criteria.
2. Have the list of jobs narrowed down to match that search criteria.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 6

(04/02/2016 - 04/15/2016)

**User Story #828 - Staff - Accepting A Job**

***Tasks***

* Add an application button to staff’s detailed job view.
* Implement notification functionality on employer end.
* Implement sending of application notification when staff applies for a job.
* Implement acceptance of job application on employer end.
* Assign staff to a job if their application is accepted.

***Acceptance Criteria***

1. Be able to apply for a job as a staff.
2. Be able to receive notifications of job applications as an employer.
3. Be able to accept/reject applications as an employer.
4. Be able to see which staff is assigned to a job as an employer.
5. Be able to view and accept notifications as an employer

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #834 - Notify Users of New Jobs**

***Tasks***

* Implement new job notification.
* Implement email sending functionality.

***Acceptance Criteria***

1. Notify user when new job is posted.
2. Send email notification.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 7

(04/16/2016 - 04/29/2016)

**User Story #837 - View Assigned Staff**

***Tasks***

* Create a button on the detailed job information to show staff assigned to the job.
* Implement table showing which staff is assigned to the selected job.

***Acceptance Criteria***

1. View staff assigned to specific jobs.
2. View detailed information on that staff.
3. Be able to remove specific staff.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #838 - Post A Resume**

***Tasks***

* Create a page to upload a resume.
* Implement upload functionality.

***Acceptance Criteria***

1. Be able to select a resume for uploading.
2. Have the system upload the resume.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #839 - View Resume**

***Tasks***

* Implement resume viewing functionality.

***Acceptance Criteria***

1. View uploaded resume.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

**User Story #852 - Reset Password (Website)**

***Tasks***

* Create password reset page.
* Implement password reset functionality.

***Acceptance Criteria***

1. Be able to reset a password.
2. Be able to enter a new desired password.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# System Design

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

## Architectural Patterns

The main architectural pattern used for this project is Model View Controller (MVC). MVC was chosen because it was the pattern that best fit iPhone application development. The IDE Xcode, and iPhone apps in general, are set up to use the MVC pattern and using said pattern allows us to use Xcode to it’s fullest potential. In addition to MVC, the system also uses a client-server architecture wherein the clients are the individual applications running on the users’ phones and the server is the Ubuntu LAMP server, where all of the client’s requests are made, that is hosted on Digital Ocean.

## System and Subsystem Decomposition

The diagram below, Figure 1, shows the subsystem decomposition of our system. The system is divided into four parts. The first is the view which is essentially the user interface that the user interacts with. In Xcode the user interface’s design is dictated by storyboards. These storyboards pertain to the different views that the user sees and interacts with.

The next part of our system is the controller subsystem. This subsystem is the link between the views the user sees and the data that is in the models. In Xcode this subsystem is implemented through the use of view controllers, which are classes that interact with the storyboard elements.

The third subsystem is the model. The model are the classes needed to interact with the database. The controller subsystem communicates with these classes in order to access or update data that is on the database.

The last part of the system is the database itself. This database is on the Ubuntu server that is hosted on Digital Ocean’s servers. This database is what the model connects to in order to access data.

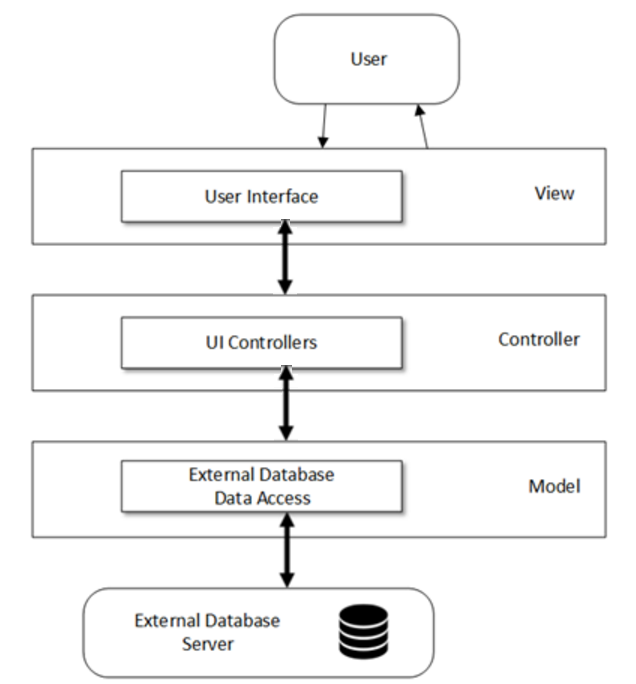


Figure 1 - Subsystem Decomposition

## Deployment Diagram

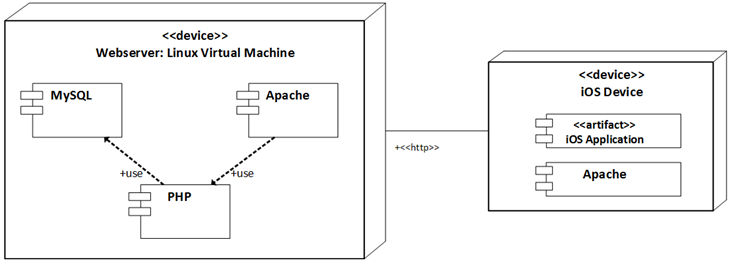


Figure 2 - Deployment Diagram

## Design Patterns

The design pattern used in the system is the singleton pattern. The singleton pattern is used in the class that handles the accessing of the database. Applying the singleton pattern there ensures that only one instance of this class is running, helping to maintain the integrity of the database by preventing other components from modifying the database.

# System Validation

In this section, all of the test cases that were down to validate our system are outlined. All of the testing was done manually, following the input requirements of the test cases.

**User Story #803 - Create a job**

System Tests

## Test Case ID - CreateJob\_001(Sunny Day)

Purpose:

* To test that a job is properly created by the system upon the submission of the requested data.

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has clicked on the create job button

Input:

* The user fills out all required fields.
* The user clicks the done button.

Expected Output:

* The system will create the job.

## Test Case ID - CreateJob\_002(Rainy Day)

Purpose:

* To test that the system rejects the job creation if required/correct information is not entered during the inputting of job information.

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has clicked the create job button.

Input:

* The user does not fill out any of the required fields.
* The user clicks the done button.

Expected Output:

* The system will not create the job, tell the user what fields are incorrect/missing, and prompt them to enter the fields again.

## Test Case ID - CreateJob\_003(Rainy Day)

Purpose:

* To test that job creation is cancelled correctly if the internet connection is lost during the job information entry phase.

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has has clicked the create job button

Input:

* The user fills out all required fields.
* The user disconnects their internet connection.
* The user clicks the done button.

Expected Output:

* The system will cancel job creation and inform the user that the internet connection was lost and that the job was not created.

**User Story #806 - Edit A Job**

## Test Case ID - EditJob\_001(Sunny Day)

Purpose:

* To test that the job can be edited.

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has clicked on the view jobs button
* The user has selected a job to edit

Input:

* The user clicks on the edit job button
* The user updates the desired information
* The user clicks on the save button

Expected Output:

* The system will update the job and return to the user’s job view.

## Test Case ID - ViewJob\_002(Sunny Day)

Purpose:

* To test that job edits are saved correctly

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has edited a job

Input:

* The user clicks on the job that was edited to view job information

Expected Output:

* The system updated the job information and the user is able to see his edited job information

**User Story #804 - View Job**

## Test Case ID - ViewJob\_001(Sunny Day)

Purpose:

* To test that the list of created jobs is populated

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has clicked on the view jobs button

Input:

* None

Expected Output:

* The system will show a list of created jobs.

## Test Case ID - ViewJob\_002(Sunny Day)

Purpose:

* To test that jobs can be selected and viewed from the list of created jobs

Precondition:

* The user has an active internet connection
* The user has a employer account
* The user has clicked the view jobs button.

Input:

* The user clicks on the desired job.

Expected Output:

* The system will provide detailed information on the job.

**User Story #805 - View a Job (Staff)**

## Test Case ID - ViewJobStaff\_001(Sunny Day)

Purpose:

* To test that the staff user can see created jobs

Precondition:

* The user has an active internet connection
* The user has a staff account
* The user is at the staff home screen

Input:

* The user clicks on the view jobs button

Expected Output:

* The system will show a list of created jobs.

## Test Case ID - ViewJob\_002(Sunny Day)

Purpose:

* To test that jobs can be selected and that detailed information can be viewed from the list of created jobs

Precondition:

* The user has an active internet connection
* The user has a staff account
* The user has clicked the view jobs button.

Input:

* The user clicks on the desired job.

Expected Output:

* The system will provide detailed information on the job.

**User Story #837 - View Assigned Staff**

## Test Case ID - ViewStaff\_001(Sunny Day)

Purpose:

* To test that staff assigned to the job are shown correctly.

Precondition:

* The user has an active internet connection.
* The user has a employer account.
* The user is at the detailed information page for a selected job.

Input:

* The user clicks on the view staff button.

Expected Output:

* The system will show a list of assigned staff.

## Test Case ID - ViewStaff\_002(Rainy Day)

Purpose:

* To test that the list is empty if no staff are assigned.

Precondition:

* The user has an active internet connection.
* The user has a employer account.
* The user is at the detailed information page for a selected job.

Input:

* The user clicks on the view staff button.

Expected Output:

* The system will show an empty list.

## Test Case ID - ViewStaff\_003(Sunny Day)

Purpose:

* To test that the user can remove the selected staff.

Precondition:

* The user has an active internet connection.
* The user has a employer account.
* The user has has clicked the create job button.

Input:

* The user clicks on the staff they wish to remove.
* The user clicks on the remove button.
* The user clicks on the confirm button on the confirmation popup.

Expected Output:

* The system will remove the selected staff from the job.

**User Story #828 - Accept A Job**

## Test Case ID - AcceptJob\_001(Sunny Day)

Purpose:

* To test that the job application can be sent properly.

Precondition:

* The users have an active internet connection.
* The staff user has a staff account.
* The employer user has an employer account.
* The staff user has selected the desired job.

Input:

* The staff user clicks the apply button.

Expected Output:

* The system will send a notification to the employer user with the staff user’s application.
* The system will let the staff user know that they have sent the application.

## Test Case ID - AcceptJob\_002(Sunny Day)

Purpose:

* To test that the employer user can view and accept job applications.

Precondition:

* The employer user has an active internet connection
* A staff user has submitted an application.
* The employer user has an employer account
* The employer user has clicked on the notifications button in the home page.

Input:

* The employer user clicks on the notification containing the staff user’s application.
* The employer user clicks on the accept button.

Expected Output:

* The system shall let the employer user know that they have accepted the application.
* The system will add the staff user to the job.
* The system will remove the job application from the employer’s notifications page.

## Test Case ID - AcceptJob\_003(Sunny Day)

Purpose:

* To test that the employer user can view and reject job applications.

Precondition:

* The employer user has an active internet connection
* A staff user has submitted an application.
* The employer user has an employer account
* The employer user has clicked on the notifications button in the home page.

Input:

* The employer user clicks on the notification containing the staff user’s application.
* The employer user clicks on the reject button.

Expected Output:

* The system shall let the employer user know that they have rejected the application.
* The system will remove the job application from the employer’s notifications page.

**User Story #843 - Filter A Job**

## Test Case ID - FilterJob\_001(Sunny Day)

Purpose:

* To test that the job list can be filtered properly.

Precondition:

* The user has an active internet connection
* The user has a staff account
* The user has clicked on the filter criteria search bar.

Input:

* The user enters the desired filter criteria.

Expected Output:

* The system will only show jobs matching the filter criteria.

## Test Case ID - FilterJob\_002(Rainy Day)

Purpose:

* To test that the system correctly shows no jobs available if a search criteria with no matches is entered.

Precondition:

* The user has an active internet connection
* The user has a staff account
* The user has clicked on the filter criteria bar.

Input:

* The user enters a nonsensical query to ensure no jobs match it.

Expected Output:

* The system shall show an empty list because no matching jobs exist.

**User Story #829 - Post a Job (Website)**

## Test Case ID - PostJob\_001(Sunny Day)

Purpose:

* To test that the job is posted.

Precondition:

* The user has an employer account
* The user clicks on the ***Post Job*** button

Input:

* The user fills out the job form by entering all information related to the job or event.

Expected Output:

* The job title is added to the ***Job List*** section.

Test Case ID - ViewJob\_002(unny Day)

Purpose:

* To test that a job is posted.

Precondition:

* The user has a employer account
* The user clicks on the ***Post Job*** button

Input:

* The user submits a blank job

Expected Output:

* The system doesn’t allow the user to submit a blank job.

**User Story #834 - Notify Users of New Jobs**

Purpose: To test the functionality of the above use case in isolation.

Test Case ID - Notify\_001 (Sunny day):

Purpose:

* Test to see if users are notified whenever a new job is posted.

Precondition:

* Users have an active account.
* Users have a registered email address.

Input

* None

Expected output:

1. Users receive an email address.
2. A numeric badged is displayed in the *Staff* page

Purpose: To test the functionality of the above use case.

Test Case ID - Notify\_002 (Rainy day):

Purpose:

* Test to see if the numeric badge corresponds to the number of available jobs in the system.

Precondition:

* At least a job has been posted.
* The job has been saved in the database.

Input:

* Details of the Job.

Expected Output:

* Wrong numeric badge( It always output the number 10) .

**User Story #813 - Create Account Management page**

Purpose: To test the functionality of the above use case.

Test Case ID Management\_001 (Sunny day):

Purpose:

* Test management login page with authorized credentials

Precondition:

* The user is at the right URL
* The user has entered his username and password.
* The user has clicked the login button

Input

* Username: eddiegarcia
* Password: 123456

Expected output: The system displays the list of all registered users.

Purpose: To test the functionality of the above use case.

Test Case ID Management\_002 (Rainy day):

Purpose:

* Test user entries when creating an account with invalid input.

Precondition:

* The user is at the right URL
* The user has entered his username and password.
* The user clicks the login button.

Input:

* Username: hello
* Password: errrsdd

Expected Output:

* Username or password is incorrect

**User Story #807 - Staff & Employer Registration**

Purpose: To test the functionality of the above use case.

Test Case ID Register\_001 (Sunny day):

Purpose:

* Test user entries when creating an account with valid input

Precondition:

* The user is at the right URL
* The user has filled out the form.
* The user has clicked the sign up button

Input

* Username: speti008
* Password: 1804
* Email: [speti008@fiu.edu](mailto:speti008@fiu.edu)

Expected output: The user is presented with a link that allow him to go to the login page.

Purpose: To test the functionality of the above use case.

Test Case ID Register\_002 (Rainy day):

Purpose:

* Test user entries when creating an account with invalid input.

Precondition:

* The user is at the right URL
* The user has filled out the form.

Input:

* Username:
* Password: 1804
* Email: speti008&fiu.xml

Expected Output:

* Username is required
* Invalid email format

**User Story #839 - View Resume**

Purpose: To test the functionality of the above use case in isolation.

Test Case ID View\_001 (Sunny day):

Purpose:

* Test to see if staff are able to view their resume.

Precondition:

* Users have an active account.
* Users have a registered email address.

Input

* None

Expected output:

1. The resume is displayed

Purpose: To test the functionality of the above use case.

Test Case ID View\_001 (Rainy day):

Purpose:

* To warn staffs that are trying to view their resume when they did not even post one yet.

Precondition:

* The staff is logged in

Input:

* None

Expected Output:

* “*We’re sorry, you haven’t post a resume yet!*”

**User Story #808 - Create A Profile**

Purpose: To test the functionality of the above use case.

Test Case ID Profile\_001 (Sunny day):

Purpose:

* Test user entries when setting up a profile

Precondition:

* The user is at the right URL.
* The user has signed up or is logged in.
* The user has clicked the *submit* button.

Input

* Username: speti008
* Password: 1804
* Email: [speti008@fiu.edu](mailto:speti008@fiu.edu)

Expected output: The system takes the user to the ***Browse Jobs*** page.

Purpose: To test the functionality of the above use case.

Test Case ID Profile\_001 (Rainy day):

Purpose:

* Test user entries when creating an account with invalid input.

Precondition:

* The user is at the right URL
* The user has filled out the form.

Input:

* Username:
* Password: 1804
* Email: speti008&fiu.xml

Expected Output:

* Username is required
* Invalid email format

# 

# 

# 

# 

# Glossary

**Storyboard**: In Xcode, storyboards are the canvases where UI elements are laid out. Storyboards items can refer directly to the code and to other items in the storyboard.

**LAMP**: LAMP is an acronym for the web service stack containing a Linux based OS, an Apache web server, a MySQL database system, and PHP code.

**View Controllers**: In Xcode, view controllers are the classes that interact with the elements on a storyboard.

# Appendix

## Appendix A - UML Diagrams

### Static UML Diagrams

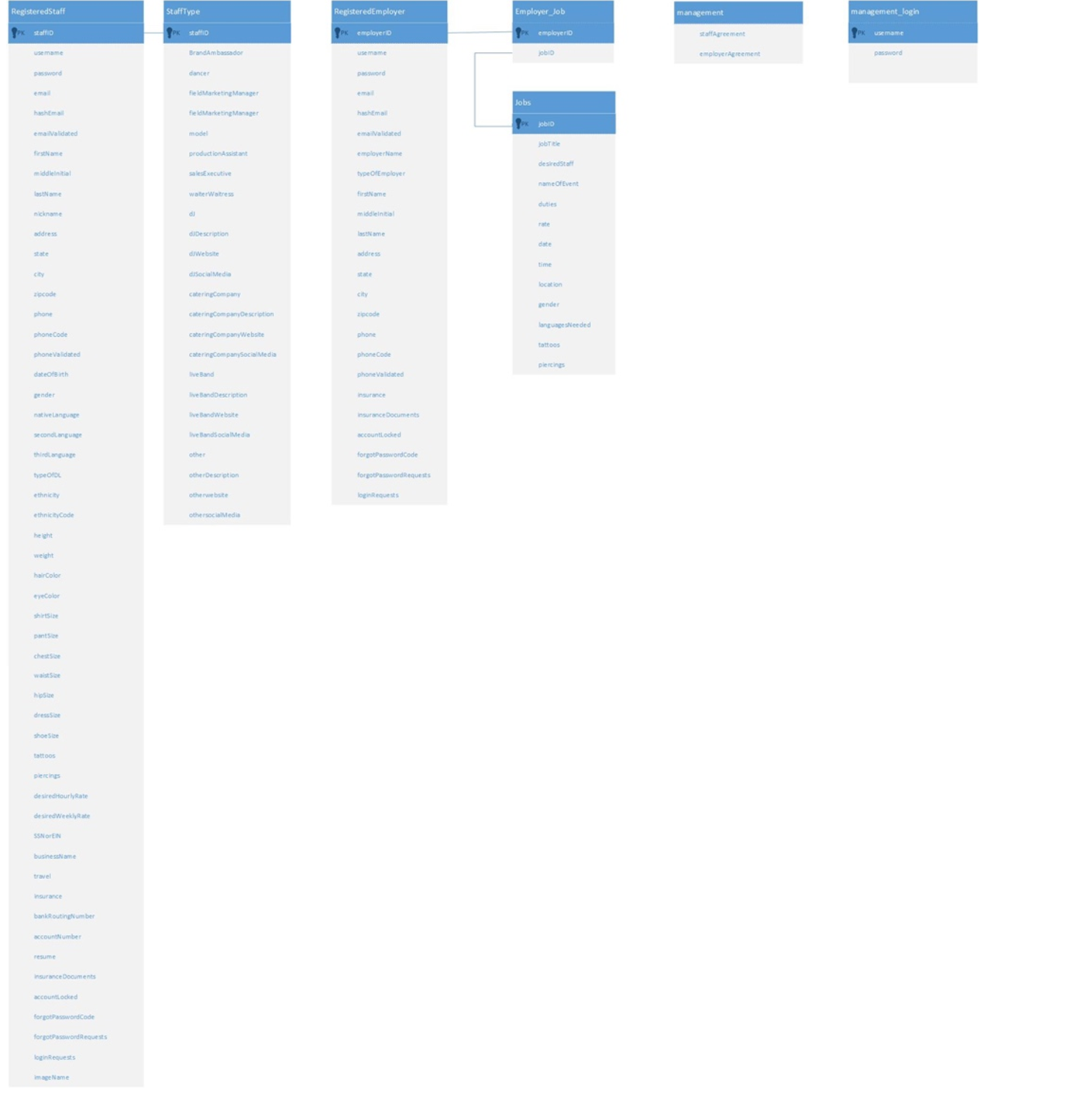


Figure A0.1 - Database Schema

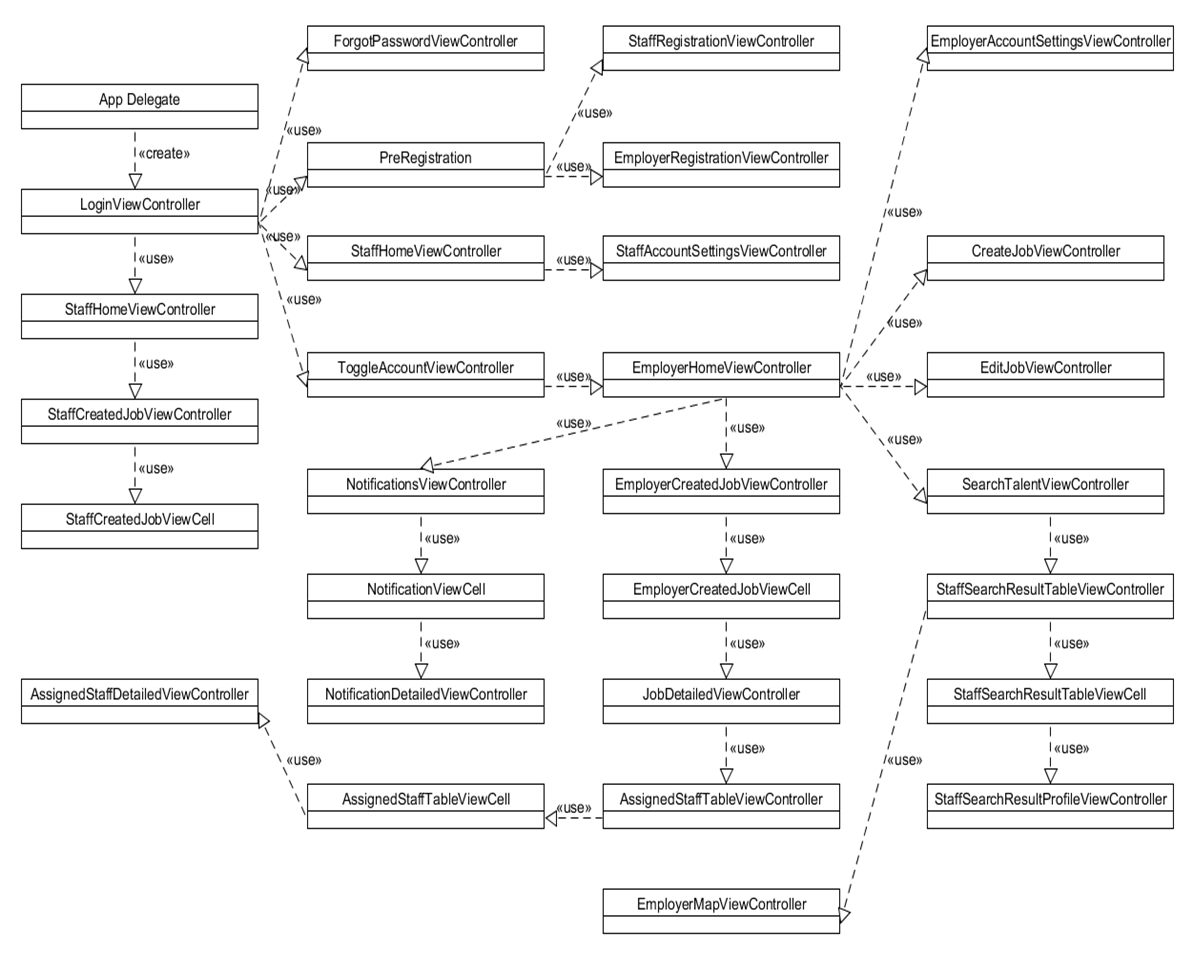


Figure A0.2 - Class Diagram

### 

### 

### Dynamic UML Diagrams

## 

Figure A1 - User Story #803 - Create A Job Sequence Diagram

## Figure A2 - User Story #843 - Filter A Job Sequence Diagram

Figure A3 - User Story #828 - Accept A Job Sequence Diagram 1

## 

Figure A4 - User Story #828 - Accept A Job Sequence Diagram 2

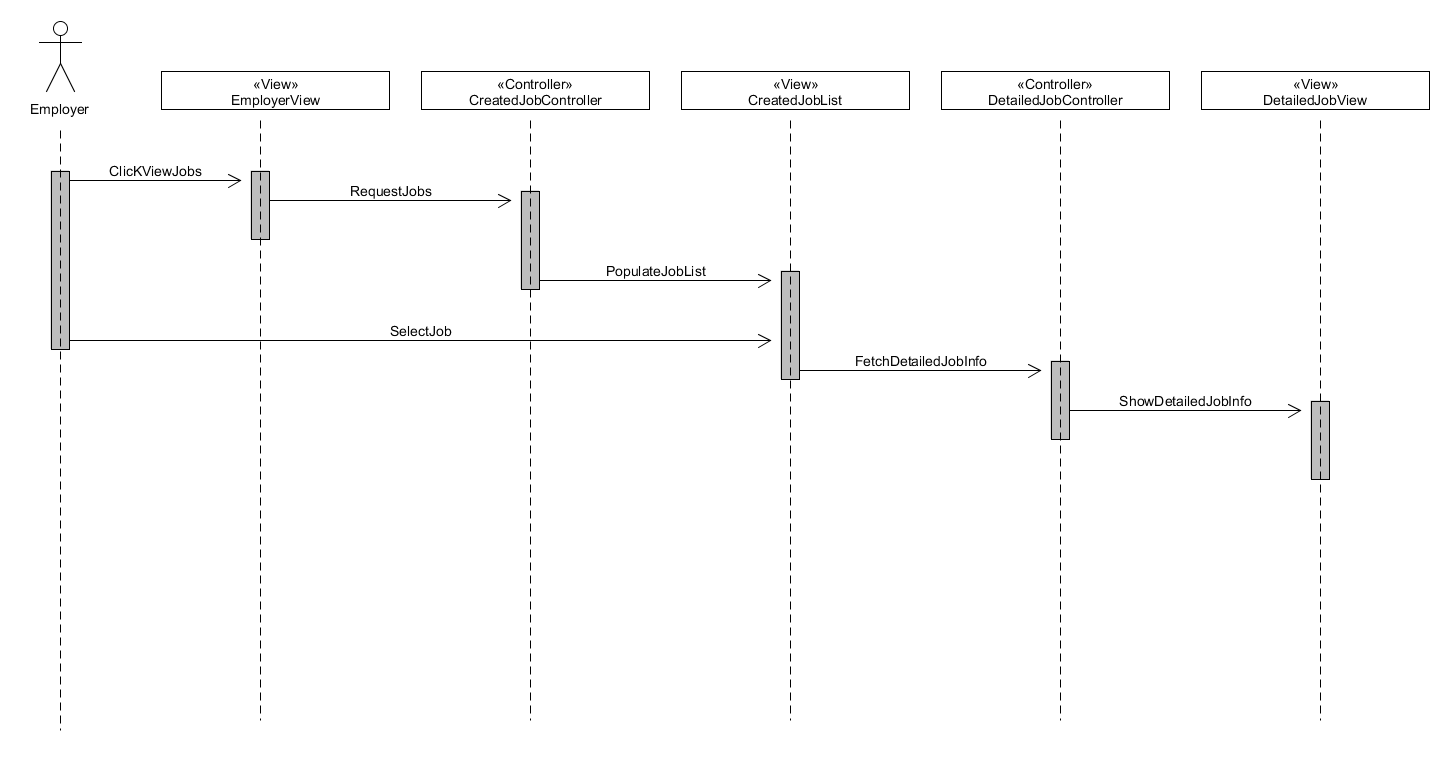


Figure A5 - User Story #804 - View Job Sequence Diagram

## 

Figure A6 - User Story #805 - View a Job (Staff) Sequence Diagram

## 

Figure A7 - User Story #806 - Edit A Job Sequence Diagram

## 

Figure A8 - User Story #837 - View Assigned Staff Sequence Diagram

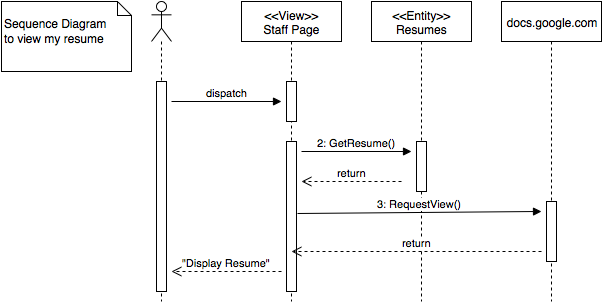


Figure A9 - User Story #839 - View Resume Sequence Diagram

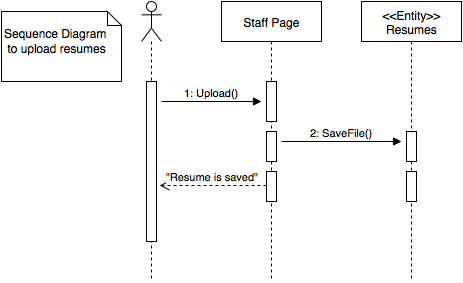


Figure A10 - User Story #838 - Upload Resume Sequence Diagram

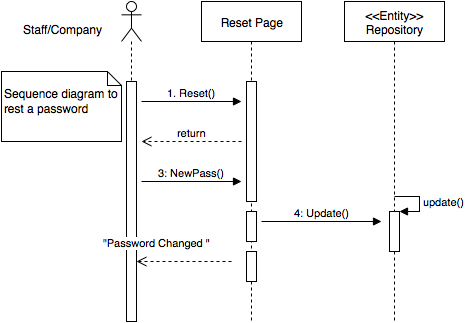


Figure A11 - User Story #852 - Reset Password Sequence Diagram

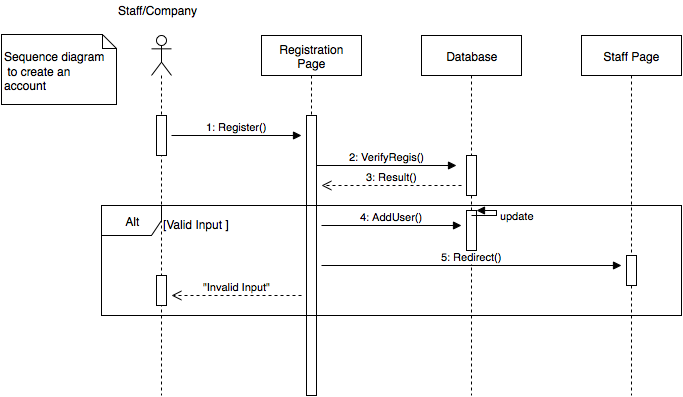


Figure A12 - User Story #807 - Staff & Employer Registration Sequence Diagram

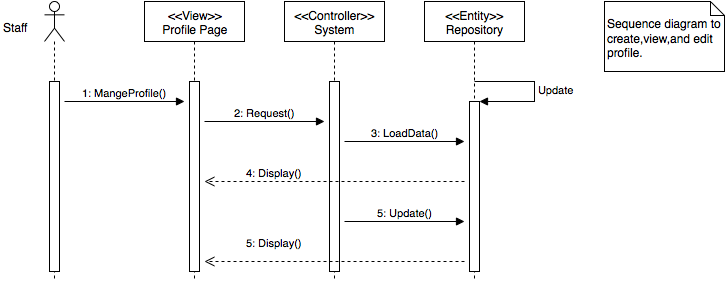


Figure A13 - User Story #808 - Create A Profile Sequence Diagram

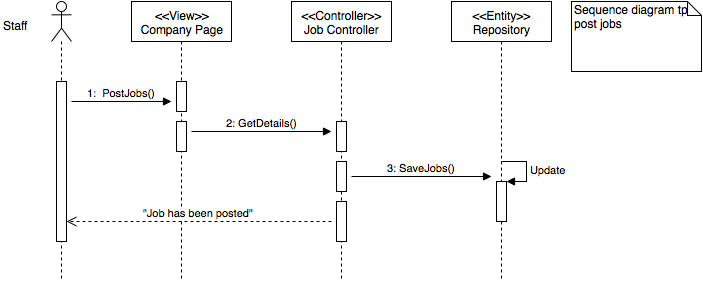


Figure A14 - User Story #829 - Post a Job (Website) Sequence Diagram

## 

## 

## Appendix B - User Interface Design

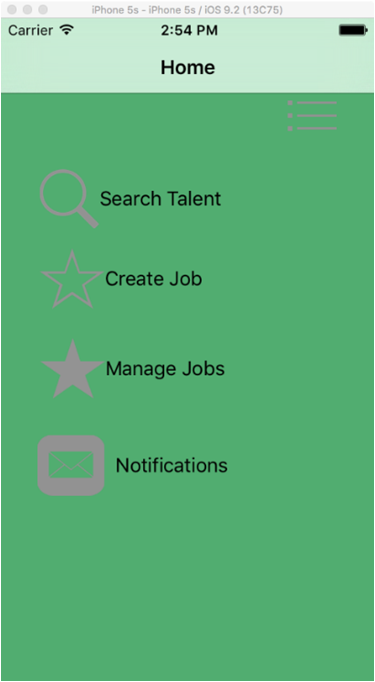


Figure B1 - Employer: Home View

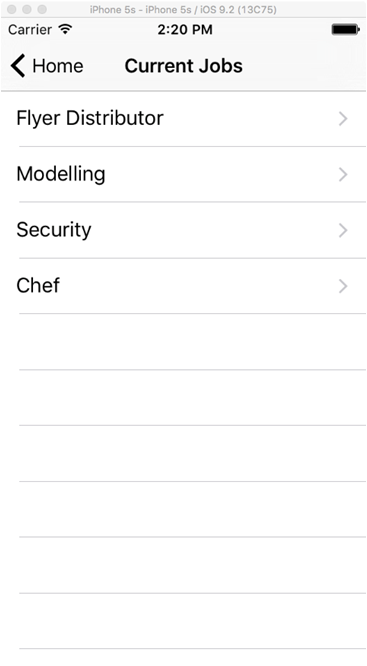


Figure B2 - Employer: Job List



Figure B3 - Employer: Detailed Job

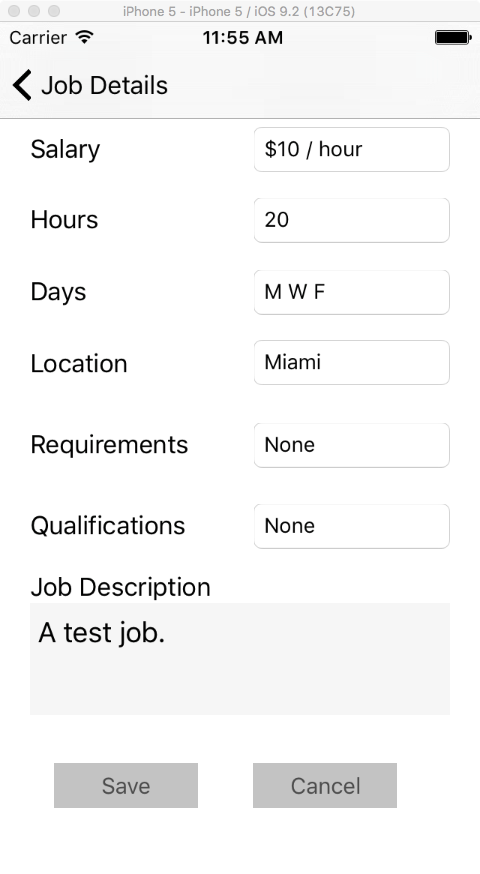


Figure B4 - Employer: Edit Job

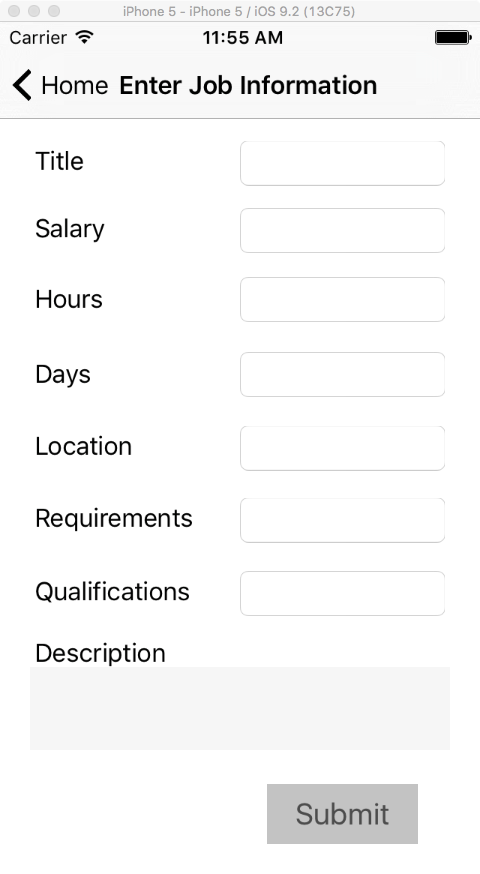


Figure B5 - Employer: Create Job

## 

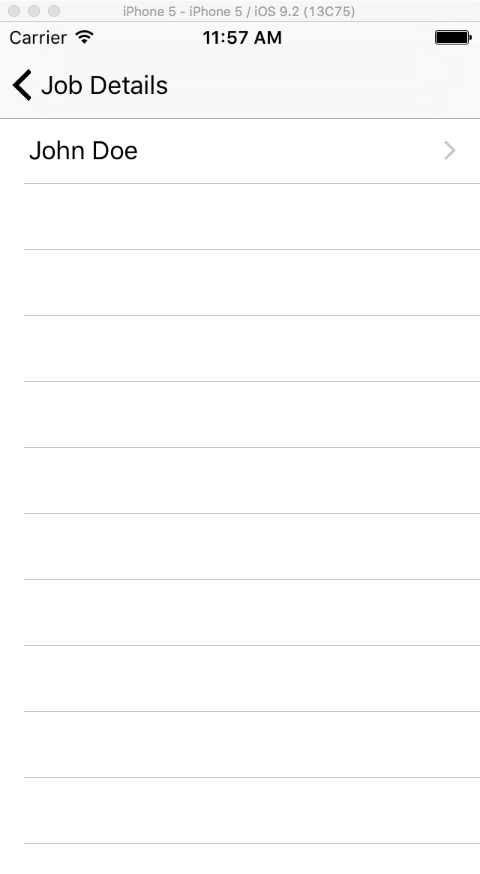


Figure B6 - Employer: View Assigned Staff

## 

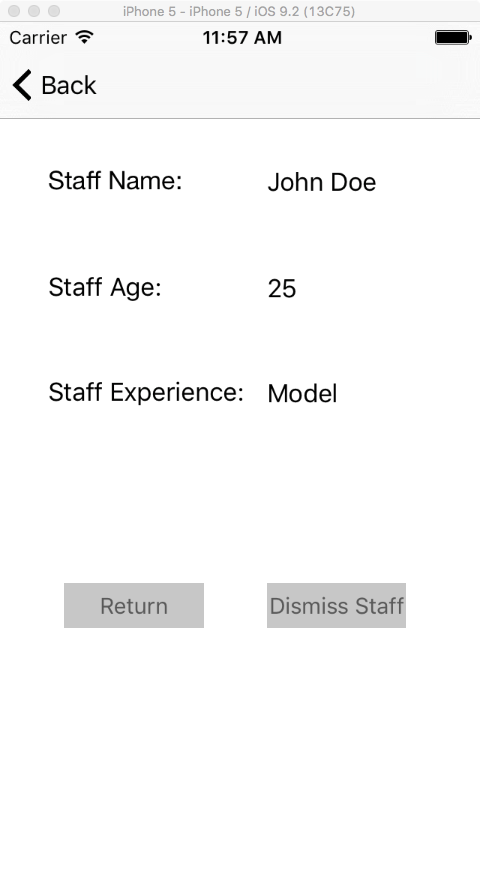


Figure B7 - Employer: View Assigned Staff Details

## 

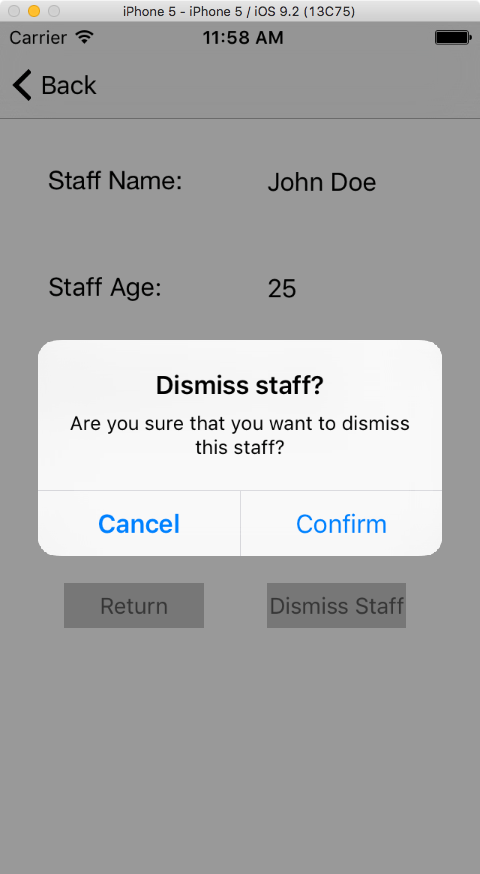


Figure B8 - Employer: Dismiss Staff?

## 

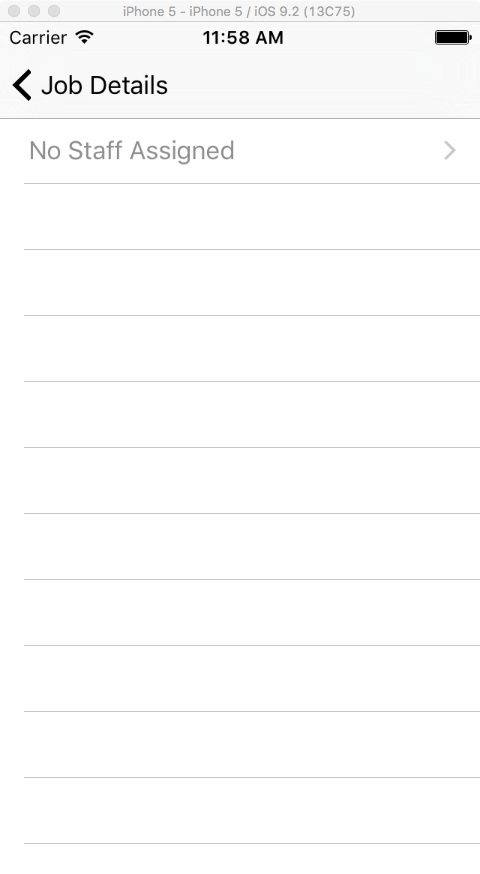


Figure B9 - Employer: Staff Dismissed/No Assigned Staff

## 

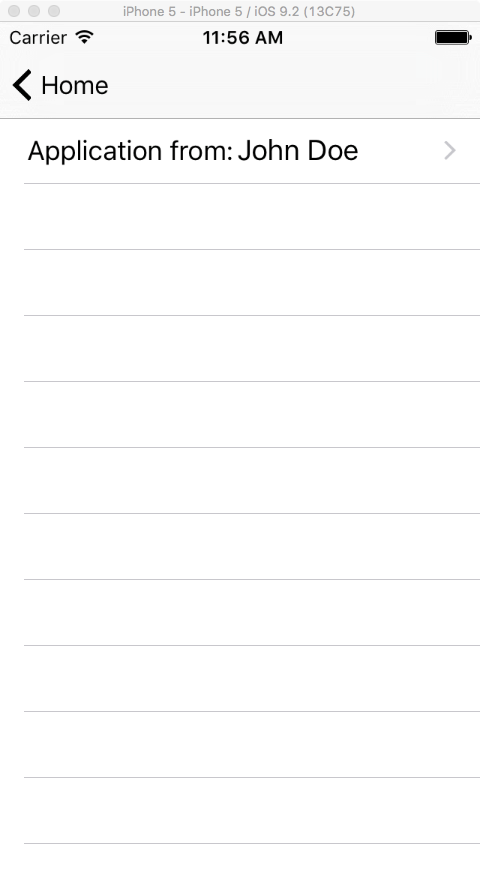


Figure B10 - Employer: Notifications

## 

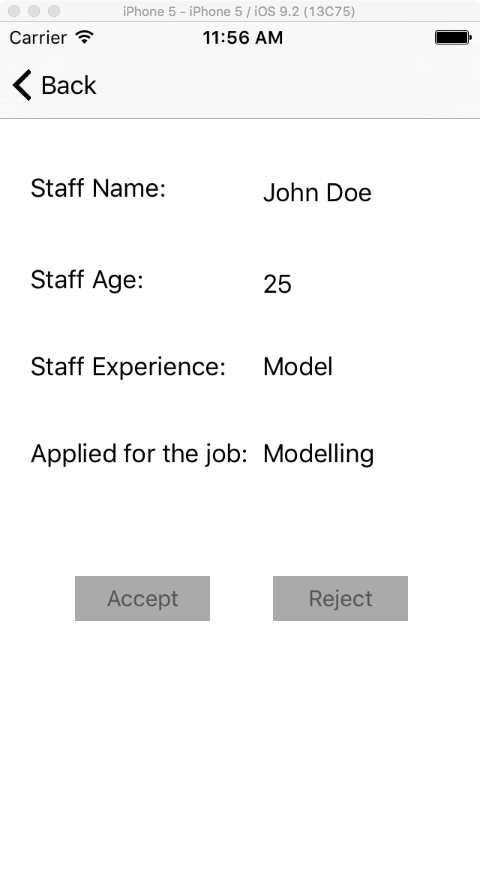


Figure B11 - Employer: Notification Details

## 

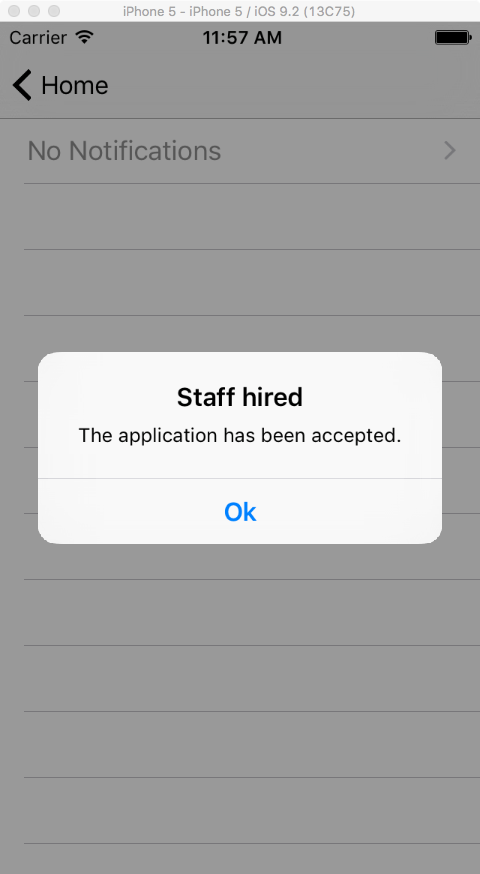


Figure B12 - Employer: Staff Hired

## 

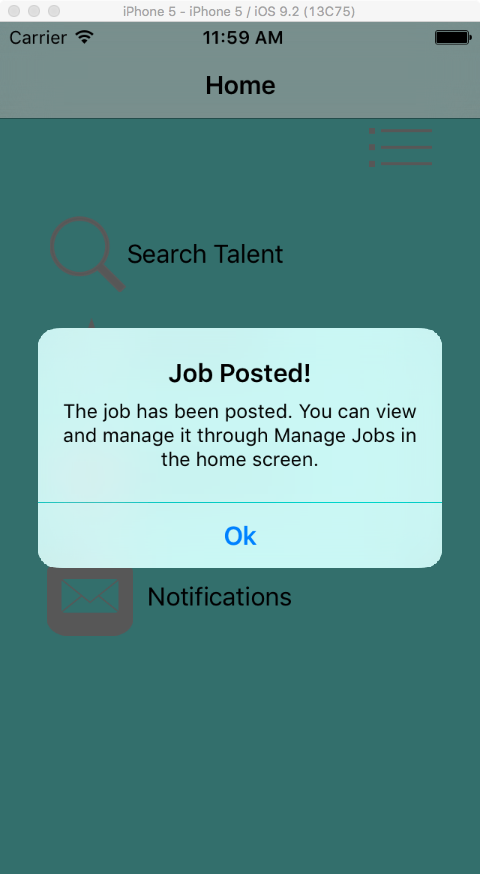


Figure B13 - Employer: Job Posted

## 

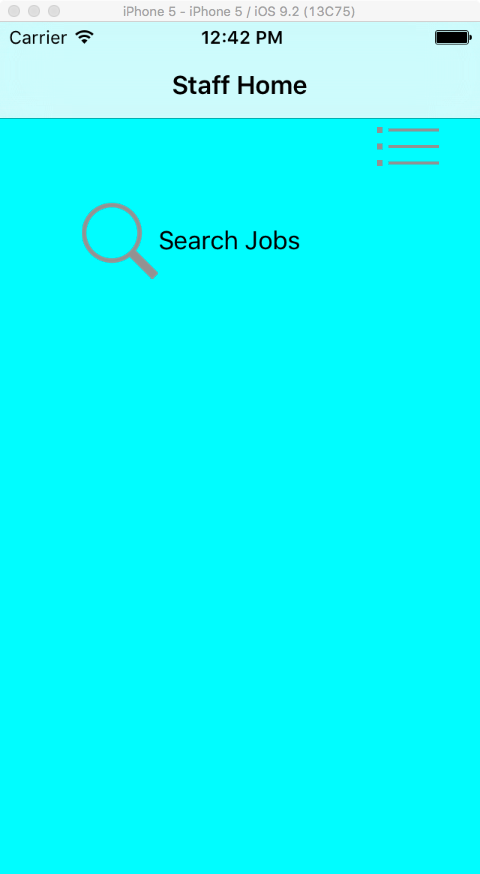


Figure B14 - Staff: Home View

## 

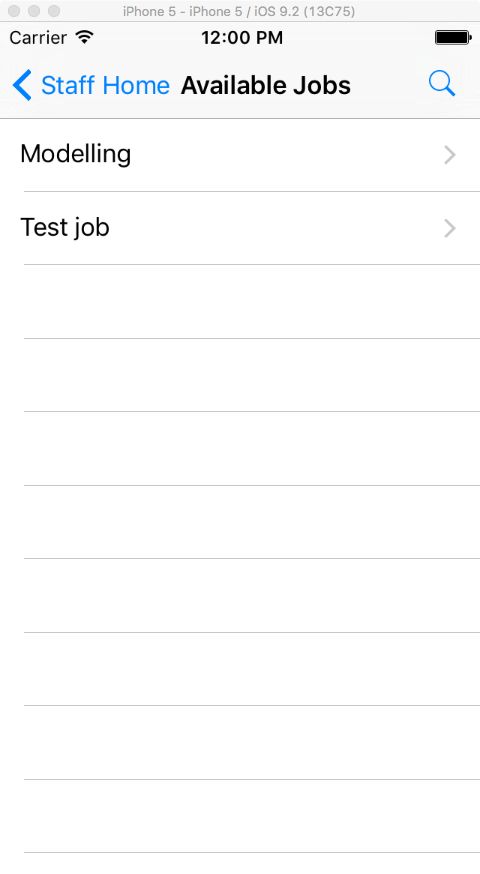


Figure B15 - Staff: Available Jobs



Figure B16 - Staff: Job Details

## 

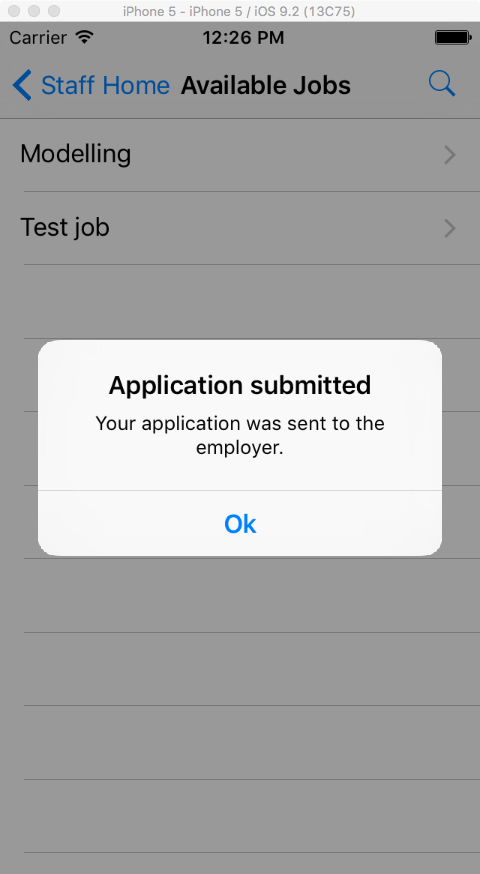


Figure B17 - Staff: Applied For Job

## Appendix C - Sprint Review Reports

**Sprint 1 Report**

Date: February 01, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-homme, Eduardo Garcia

Start time: 1:30PM

End time: 2:00PM

After discussion, the velocity of the team was estimated to be 30 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* Create A Job
* Create Profile

The team members indicated their willingness to work on the following user stories.

* Daniel Gonzalez: user story #803, Create a job
* Stephenson Petit-homme: user story #808, Create profile

**Sprint 2 Report**

Date: February 12, 2016

Attendees: Daniel Gonzalez, Stephenson, Eduardo Garcia

Start time: 1:00 PM

End time: 1:45 PM

We’ve evaluate the velocity of the team at 30 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* Employer - View Jobs
* Create Profile

The team members indicated their willingness to work on the following user stories.

* Daniel Gonzalez: user story #817, Create A Job
* Stephenson Petit-Homme: user story #808, Monitor Account

**Sprint 3 Report**

Date: February 25, 2016

Attendees: Daniel Gonzalez, Stephenson, Eduardo Garcia

Start time: 1:00 PM

End time: 1:45 PM

We’ve evaluate the velocity of the team at 30 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* Staff - View Jobs
* Create Account Management Page
* Employer - Edit Jobs
* Search For Jobs

The team members indicated their willingness to work on the following user stories.

* Daniel Gonzalez: user story #805, Staff - View Jobs
* Daniel Gonzalez: user story #806, Employer - Edit Jobs
* Stephenson Petit-Homme: user story #813, Create Account Management Page
* Stephenson Petit-Homme: user story #811, Search For Jobs

**Sprint 4 Report**

Date: March 17, 2016

Attendees: Daniel Gonzalez, Stephenson, Eduardo Garcia

Start time: 3:00 PM

End time: 3:55 PM

We’ve evaluate the velocity of the team at 30 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* Post A Job
* Filter Available Jobs

The team members indicated their willingness to work on the following user stories.

* Daniel Gonzalez: user story #843, Filter Available Jobs
* Stephenson Petit-Homme: user story #829, Post A Job

**Sprint 5 Report**

Date: April 02, 2016

Attendees: Daniel Gonzalez, Stephenson, Eduardo Garcia

Start time: 7:00 AM

End time: 7:45 AM

We’ve evaluate the velocity of the team at 30 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* Staff - Accept a Job
* Notify Users of New Jobs

The team members indicated their willingness to work on the following user stories.

* Daniel Gonzalez: user story #828, Staff - Accept Job
* Stephenson Petit-Homme: user story #834, Notify Users of New Jobs

**Sprint 6 Report**

Date: April 02, 2016

Attendees: Daniel Gonzalez, Stephenson, Eduardo Garcia

Start time: 8:50 PM

End time: 9: 45 PM

We’ve evaluate the velocity of the team at 30 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* View Assigned Staff
* Post A Resume
* View Resume
* Reset Password

The team members indicated their willingness to work on the following user stories.

* Daniel Gonzalez: user story #837, View Assigned Staff
* Stephenson Petit-Homme: user story #838, Post A Resume
* Stephenson Petit-Homme: user story #839, View Resume
* Stephenson Petit-Homme: user story #852, Reset Password

**Sprint 7 Report**

Date: April 30, 2016

Attendees: Daniel Gonzalez, Stephenson, Eduardo Garcia

Start time: 3:00 PM

End time: 3: 30 PM

We’ve evaluate the velocity of the team at 32 points.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* This was the final sprint.

The team members indicated their willingness to work on the following user stories.

* This was the final sprint.

## Appendix D - Sprint Retrospective Reports

**Sprint 1 Retrospective**

Date: February 01, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-homme

Start time: 1:00 PM

End time: 1:30 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

The velocity was slower than expected.

* Did we do a good job estimating the points (time required) for each user story?

More points were needed than what was expected.

* Did each team member work as scheduled?

Yes, but there were deviations from the estimated schedule.

What went right?

* The documentation for this sprint’s user stories are in place.
* Feedback was received on documentation from product owner. Small changes need to be made.
* We familiarized ourselves with the development process expected of us in following sprints.

How to address the issues in the next sprint?

* How to improve the process?
  + Finish required planning/documentation early on in the week to leave ample time for programming.
  + Make better use of our time. Work more efficiently, with less distractions.
* How to improve the product?
  + Provide design feedback to product owner when it is deemed useful.

**Sprint 2 Retrospective**

Date: February 12, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-Homme

Start time: 1:45 PM

End time: 2:15 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

Yes

* Did we do a good job estimating the points (time required) for each user story?

Somewhat. It was much closer than previous sprint.

* Did each team member work as scheduled?

Yes

What went right?

* We managed to get more user stories done this sprint
* We gained a better understanding of our work environment

How to address the issues in the next sprint?

* How to improve the process?
  + Put in more programming time.
  + Communicating more with the product owner and each other.
* How to improve the product?
  + Help out our product owner with design and user story selection.

**Sprint 3 Retrospective**

Date: February 26, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-Homme

Start time: 2:45 PM

End time: 3:15 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

No

* Did we do a good job estimating the points (time required) for each user story?

Somewhat. It was much closer than previous sprint.

* Did each team member work as scheduled?

Yes

What went right?

* We managed to get more user stories done this sprint
* We gained a better understanding of our work environment

How to address the issues in the next sprint?

* How to improve the process?
  + Put in more programming time.
  + Communicating more with the product owner and each other.
* How to improve the product?
  + Help out our product owner with design and user story selection.

**Sprint 4 Retrospective**

Date: March 18, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-Homme

Start time: 3:45 PM

End time: 4:15 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

Yes

* Did we do a good job estimating the points (time required) for each user story?

Somewhat. It was much closer than previous sprint.

* Did each team member work as scheduled?

Yes

What went right?

* We managed to get more user stories done this sprint
* We gained a better understanding of our work environment

How to address the issues in the next sprint?

* How to improve the process?
  + Put in more programming time.
  + Communicating more with the product owner and each other.
* How to improve the product?
  + Help out our product owner with design and user story selection.
  + Ensure that the overall design matches website theme

**Sprint 5 Retrospective**

Date: April 01, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-Homme

Start time: 5:45 PM

End time: 6:30 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

Yes

* Did we do a good job estimating the points (time required) for each user story?

Yes. It was much closer than previous sprint.

* Did each team member work as scheduled?

Yes

What went right?

* We managed to get more user stories done this sprint
* We gained a better understanding of our work environment

How to address the issues in the next sprint?

* How to improve the process?
  + Put in more programming time.
  + Communicating more with the product owner and each other.
* How to improve the product?
  + Help out our product owner with design and user story selection.
  + Ensure that the overall design matches website theme
  + Ensure that all the requirements are met for each user story

**Sprint 6 Retrospective**

Date: April 16, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-Homme

Start time: 9:45 PM

End time: 10:15 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

Yes

* Did we do a good job estimating the points (time required) for each user story?

Somewhat. It was much closer than previous sprint.

* Did each team member work as scheduled?

Yes

What went right?

* We managed to get more user stories done this sprint
* We gained a better understanding of our work environment

How to address the issues in the next sprint?

* How to improve the process?
  + Put in more programming time.
  + Communicating more with the product owner and each other.
* How to improve the product?
  + Help out our product owner with design and user story selection.
  + Ensure that we are following Eddie’’s guidelines

**Sprint 7 Retrospective**

Date: April 30, 2016

Attendees: Daniel Gonzalez, Stephenson Petit-Homme

Start time: 2:30 PM

End time: 3:00 PM

What went wrong?

* Did we do a good job estimating our team's velocity?

Yes

* Did we do a good job estimating the points (time required) for each user story?

Yes. Estimation was good.

* Did each team member work as scheduled?

Yes

What went right?

* We continued our steady pace of implementation.
* We provided progress feedback to product owner in the middle of the sprint.

How to address the issues in the next sprint?

* How to improve the process?
  + Put in more programming time.
* How to improve the product?
  + Help out our product owner with design and user story selection.
  + Ensure that we are following our product owner’s guidelines

# References

The work of the previous group, that is, the Go Local App1.0 group was used in the making of this report. The documentation of previous group members Luis Castillo and Wilfredo Gomez was paramount to the analysis and understanding of the existing system.