***Enhanced Job and Candidate Application***

***Design Specification***

C:\Users\Simul\Desktop\Winter 2015\CSC 4996\MVC App\User Login\User Login\Content\Images\computech logo.png 

Version: 1.0

03/10/2015

**Prepared by:**

Simul Kadakia

Wesley Trescott

Gagandeep Singh

Contents

[1. Introduction 3](#_Toc413445464)

[1.1 Purpose 3](#_Toc413445465)

[1.2 Scope 3](#_Toc413445466)

[1.3 Definitions, Acronyms and Abbreviations 3](#_Toc413445467)

[1.4 References 3](#_Toc413445468)

[2. Assumptions / Constraints / Standards 4](#_Toc413445469)

[2.1 Design Constraints: 4](#_Toc413445470)

[2.2 Assumptions and Dependencies 4](#_Toc413445471)

[2.3 Components: 4](#_Toc413445472)

[3. Architecture Design 5](#_Toc413445473)

[3.1 Logical View 5](#_Toc413445474)

[3.2 Hardware Architecture 5](#_Toc413445475)

[3.3 Software Architecture 5](#_Toc413445476)

[3.3 Security Architecture 5](#_Toc413445477)

[3.4 Communication Architecture 5](#_Toc413445478)

[3.5 Performance 5](#_Toc413445479)

[4. System Design 5](#_Toc413445480)

[4.1 Use-cases 5](#_Toc413445481)

[4.1.1 User class – Job seeker 5](#_Toc413445482)

[4.1.2 User class – Admin 11](#_Toc413445483)

[4.2 Sequence Diagram 14](#_Toc413445484)

[4.3 Data-flow diagram 15](#_Toc413445485)

[Level 1: 15](#_Toc413445486)

[Level 2: 16](#_Toc413445487)

[Level 3: 16](#_Toc413445488)

[4.4 Database Design 18](#_Toc413445489)

[4.5 Application Program Interface 19](#_Toc413445490)

[4.6 User Interface Design 19](#_Toc413445491)

[5. Product Design Specification Approval 19](#_Toc413445492)

# 1. Introduction

This section gives a scope description and overview of everything included in this document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

## Purpose

The purpose of this Software Design Specification (SRS) document is to provide a detailed description of the design framework of the ‘Enhanced Job and Candidate Application’ system. It will also provide specific information about the input and desired output, software and hardware architecture, database, security and sequence diagrams and test cases persisting to the application.

## Scope

Enhanced Job and Search Candidate is an application developed by 3 students at Wayne State University for Computech Corporation. The goal of the application is to provide a web application service to job seekers to search for available jobs at Computech Corporation and apply to those that they are interested in. The application will provide filtering options to reduce the jobs displayed based on certain criteria. Job seekers will be able to store their profile information and resume path (resume will be stored in a folder) in database and resume which can be used when they are applying for a job using a web interface. The application will also provide functionalities to admin to deactivate and delete users abusing the system.

## 1.3 Definitions, Acronyms and Abbreviations

* EJCA - Enhanced Job and Candidate Application
* User – Job seeker who uses the application
* Admin – Admin/administrator who manages the users
* Admin portal - Part of the web application that provides special facilities to Admin
* Front End - The part of the application the user interacts with
* Back End – The part of application that manages data and is managed by developers.
* UI – User Interface which is the front end of the application
* Server –Machine that will host the web application as well as database.

## 1.4 References

* Microsoft ASP.NET MVC - <http://www.asp.net/mvc>
* Microsoft SQL Server - <https://msdn.microsoft.com/en-us/sqlserver/aa336270.aspx>
* Razor - <http://www.asp.net/web-pages/overview/getting-started/introducing-razor-syntax-(c)>
* JQuery - <http://jquery.com/>
* Bootstrap – <http://getbootstrap.com>

# 2. Assumptions / Constraints / Standards

## 2.1 Design Constraints:

This is a web application and the context of rendering the user interface based on the type of device used. End users will use the application from a modern internet browser such as Safari, Chrome, or Internet Explorer.

## 2.2 Assumptions and Dependencies

Assumptions to properly use the application include:

* Internet connection with enough bandwidth (about 1 to 2 mbps) to fully render all web application pages
* A modern internet browser with an up to date JavaScript engine and support for HTTP cookies to remember returning user logins

## 2.3 Components:

* **Microsoft ASP.NET MVC framework:** EJCA will be developed using MVC which is a software [architectural pattern](http://en.wikipedia.org/wiki/Architectural_pattern) for implementing [user interfaces](http://en.wikipedia.org/wiki/User_interface). It divides a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information is presented to or accepted from the user
* **Microsoft SQL Server:** EJCA will use MS SQL server database which will store users, jobs and admin information.
* **Razor:** Razor is a view engine which will embed server-based code on the web page.
* **JQuery:** EJCA will be using JavaScript library for more user and mobile friendly EJCA.
* **Bootstrap:** EJCA needs to be compatible on mobile browsers along with traditional desktop and bootstrap framework will enable to use the same layout on multiple platforms.

# 3. Architecture Design

## 3.1 Logical View

## 3.2 Hardware Architecture

## 3.3 Software Architecture

## 3.3 Security Architecture

## 3.4 Communication Architecture

## 3.5 Performance

# 4. System Design

## 4.1 Use-cases

### 4.1.1 User class – Job seeker

#### 4.1.1.1 Search

**Actors:** Job seeker

**Description:** User can search for various jobs available. When the user opens the website, the home page will show featured jobs with an option to conduct a job search. Users can also conduct a search for the same jobs once logged into their profiles.

**Trigger:** User clicks on **Search Jobs** button.

**Pre-conditions:** None

**Post-conditions:** System displays all the available jobs to the user.

**Normal Flow:**



**Alternative flow:** None

**Exception:** None

**Assumptions:** None

#### 4.1.1.2 Register

**Actors:** Job seeker

**Description:** Users can create a new account by providing personal information such as email id, password and first and last name. This information will be stored in database and email id and password will be used to log in user.

**Trigger:** User clicks on **Register** button under Users dropdown menu on the header.

**Pre-conditions:** None

**Post-conditions:** User can log in to the system next time and apply for job.

**Normal Flow:**



**Alternative flow:** N/A

**Exception:**



**Assumptions:** User has not previously registered with the same email address.

#### 4.1.1.3 Login

**Actors:** Job seeker

**Description:** User can login by entering email id and password entered during registration process. **Trigger:** User clicks on **Login** button under Users dropdown menu on the header.

**Pre-conditions:** User has registered with the same email address.

**Post-conditions:** User can view and update their profile as well as apply for jobs.

**Normal Flow:**



**Alternative flow:** N/A

**Exception:**



**Assumptions:** User email address and password exists in the database.

#### 4.1.1.4 Retrieve Password

**Actors:** Job seeker

**Description:** User should then be able to retrieve password through email. User can request to change password which will send further instruction to his or her email

**Trigger:** User clicks on **Forgot Password** link under Users dropdown menu on the header.

**Pre-conditions:** User has registered with the same email address.

**Post-conditions:** User can reset the password and will be able to login in to their account.

**Normal Flow:**



**Alternative flow:** N/A

**Exception:** For security purpose, there will be no exception in password retrieval controller. Even if an email id doesn’t exist in the database, user will redirected to Success page but email will not be sent.

**Assumptions:** User email id exists in the database.

#### 4.1.1.5 Apply

**Actors:** Job seeker

**Description:** User can apply to any jobs which they are interested in.

**Trigger:** User clicks on **Apply** button on **Details** page.

**Pre-conditions:** User has logged in.

**Post-conditions:** User is taken to page displaying ‘Application submitted successfully’ message.

**Normal Flow:**



**Alternative flow:**



**Exception:** User cannot apply if user is not logged in. As shown in alternate path above, if such a user clicks on Apply, user will be redirected to Login page and after validating credentials, user will be redirected back to Job Details page and then it follows the same process.

**Assumptions:** User has logged in.

#### 4.1.1.6 Filter by skills, location, pay rate, and experience:

**Actors:** Job seeker

**Description:** User can filter jobs based on skills, location, pay rate, and experience required for the job.

**Trigger:** User clicks on **Filter Jobs** button on Home page or Job Search page.

**Pre-conditions:** User selects at-least one criteria.

**Post-conditions:** System will display jobs based on selected criteria.

**Normal Flow:**



**Alternative flow:** None

**Exception:** None

**Assumptions:** User selects at-least one criteria.

#### 4.1.1.7 Profile Information

**Actors:** Job seeker

**Description:** User can enter his or her information such as address, phone number, skills and experience in his or her profile page.

**Trigger:** User clicks on **Update Profile** button on under the menu button on the header.

**Pre-conditions:** User has logged in or user has registered.

**Post-conditions:** User information will be saved in the database.

**Normal Flow:**



**Alternative flow:** None

**Exception:**



**Assumptions:** User has registered with the same email address.

#### 4.1.1.8 Change Information

**Actors:** Job seeker

**Description:** User can change information stored in his or her profile page. User should be able to change any fields stored in his or her profile page except the email ID.

**Trigger:** User clicks on **Update Profile** button on under the menu button on the header.

**Pre-conditions:** User has logged in.

**Post-conditions:** User information will be updated in the database.

**Normal Flow:**



**Alternative flow:** None

**Exception:** 

**Assumptions:** User has registered their account and saved information in their profile page.

#### 4.1.1.9 Upload Resume

**Actors:** Job seeker

**Description:** User can upload one resume and attach them to his or her profile.

**Trigger:** User clicks on **Upload Resume** button on **Profile** page.

**Pre-conditions:** User has logged in.

**Post-conditions:** None

**Normal Flow:**



**Alternative flow:** None

**Exception:** 

**Assumptions:** User has registered with the same email address.

### 4.1.2 User class – Admin

#### 4.1.2.1 Login

**Actors:** Admin

**Description:** Admin can login by entering email id and password stored in database.

**Trigger:** User clicks on **Admin** button on under the menu button on the header.

**Pre-conditions:** Admin email id is stored in the Tbl\_Admin table in database.

**Post-conditions:** Admin will be directed to Admin Dashboard.

**Normal Flow:**



**Alternative flow:** None

**Exception:**



**Assumptions:** Admin email address exists in the database.

#### 4.1.2.2 Search

**Actors:** Admin

**Description:** Admin can search for all registered users by their first and/or last names.

**Trigger:** User enters either first name or last name or both and click on **Search User** button.

**Pre-conditions:** Admin has logged in.

**Post-conditions:** Users matching the criteria will be displayed.

**Normal Flow:**



**Alternative flow:** None

**Exception:**



**Assumptions:** Admin’s email address and password exists in Tbl\_Admin.

#### 4.1.2.3 Deactivate Users

**Actors:** Admin

**Description:** Admin can deactivate users which will disable them to login into their account..

**Trigger:** Admin clicks on **Deactivate User** button next to user’s name.

**Pre-conditions:** Admin has logged in.

**Post-conditions:** User will be assigned 0 in **IsActive** entity in **Tbl\_User** which will lock the user and disable them to login again with the same email address.

**Normal Flow:**



**Alternative flow:** None

**Exception:** None

**Assumptions:** Admin’s email address and password exists in Tbl\_Admin.

#### 4.1.2.4 Delete Users

**Actors:** Job seeker

**Description:** Admin can delete users which will delete all of their information from the database.

**Trigger:** User clicks on **Delete** button next to the user’s name.

**Pre-conditions:** Admin has logged in.

**Post-conditions:** None

**Normal Flow:**

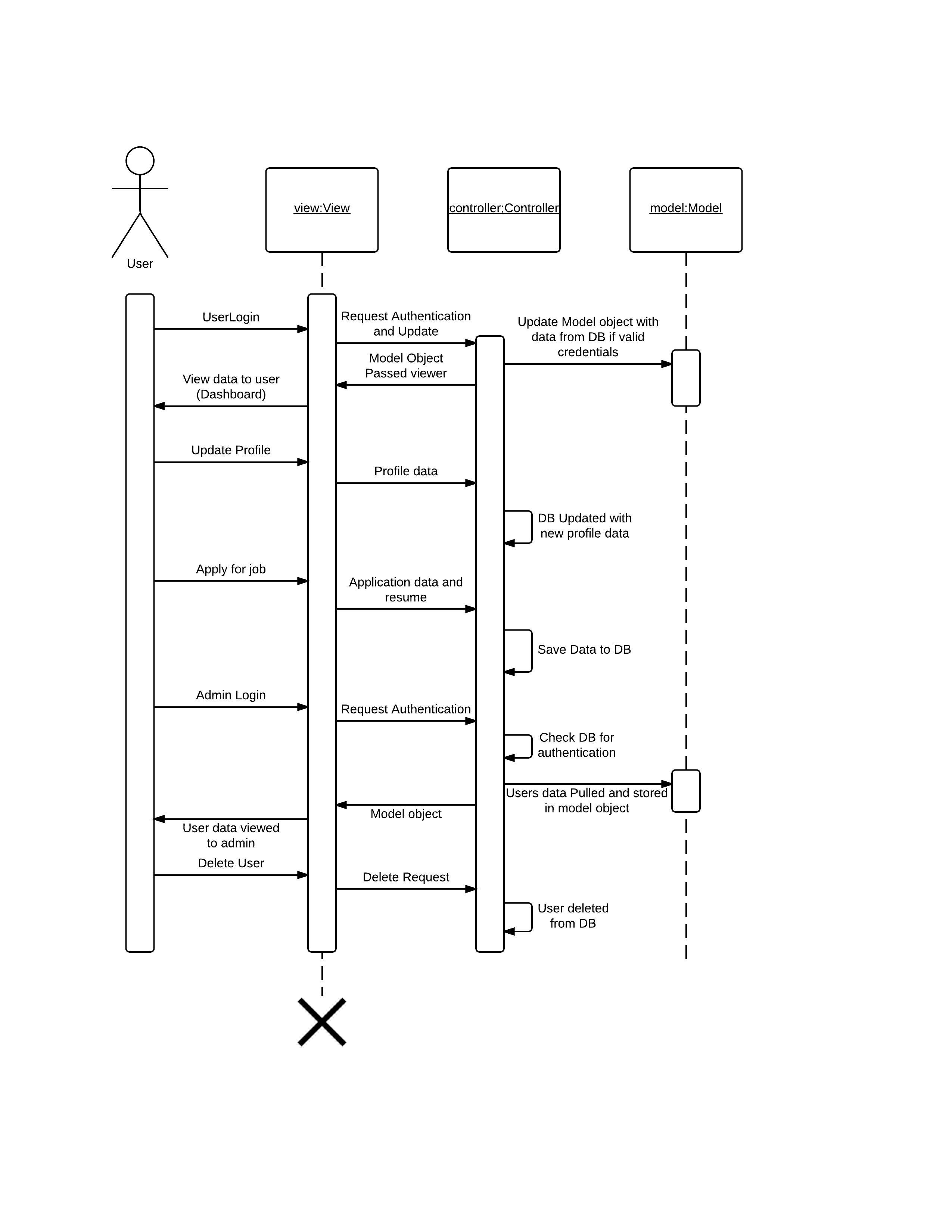


**Alternative flow:** None

**Exception:** None

**Assumptions:** Admin’s email address exists in the database.

## 4.2 Sequence Diagram



## 4.3 Data Flow Diagram

#### Level 1

Shows the flow of data among the user, web application, and the SQL database.



#### Level 2

Provides in-depth view of the data flow among the various components of the web application such as the UI, Application Core i.e. Model and Controller components, Viewer, and the SQL database.



#### Level 3

##### **User flow**



##### **Data flow**



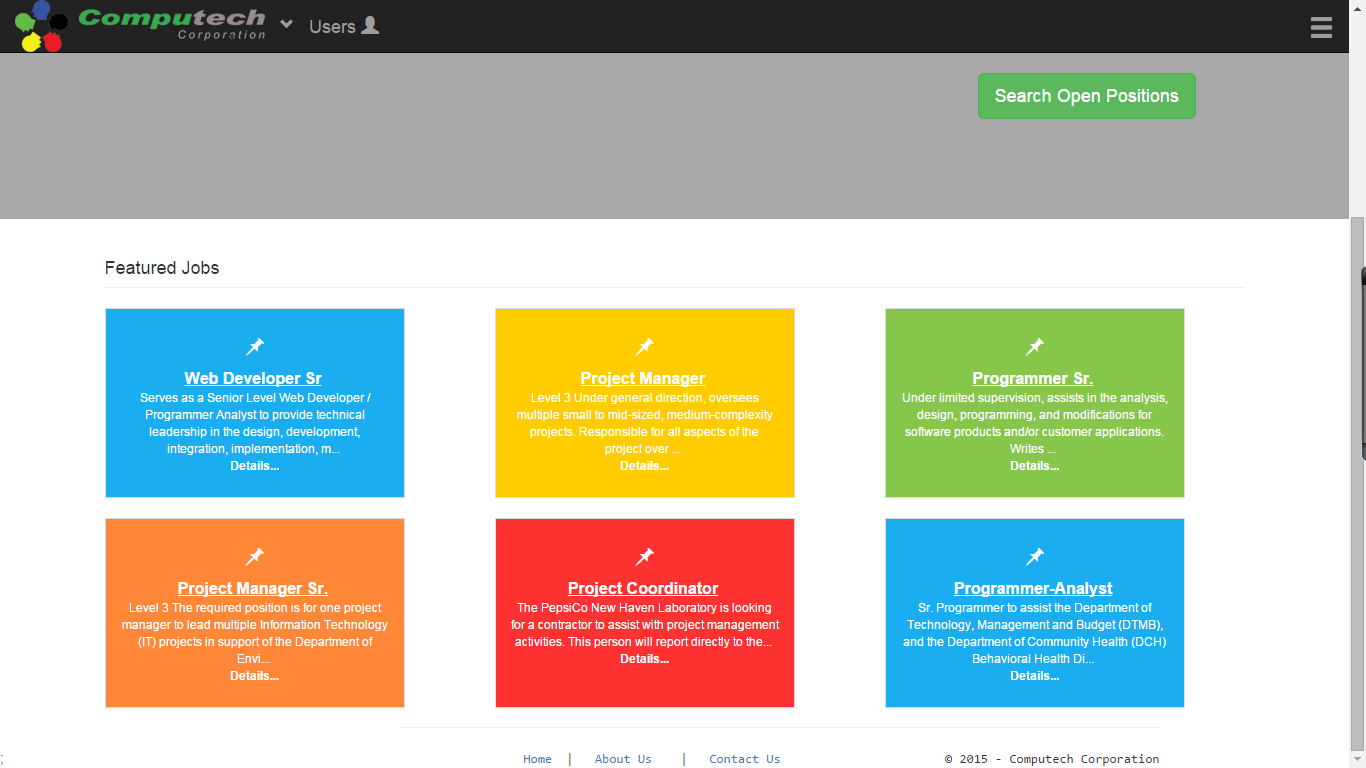
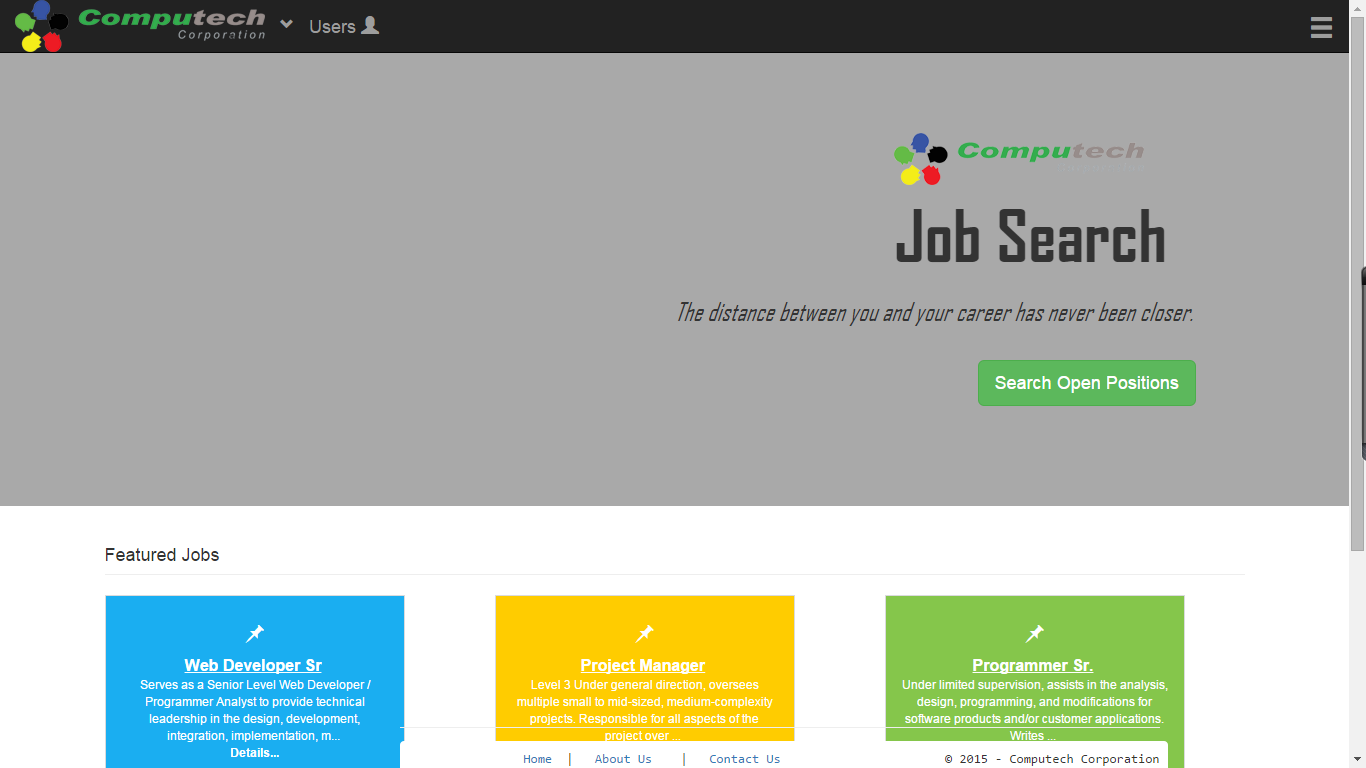
## 4.4 Database Design



## 4.5 Application Program Interface

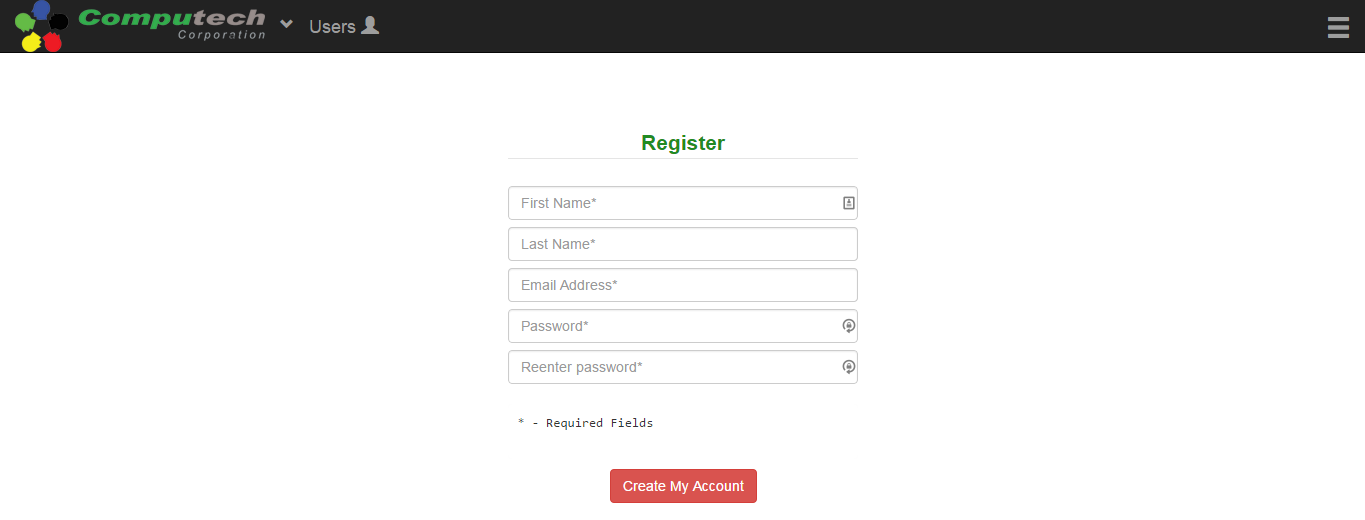
## 4.6 User Interface Design

### 4.6.1 Home Page

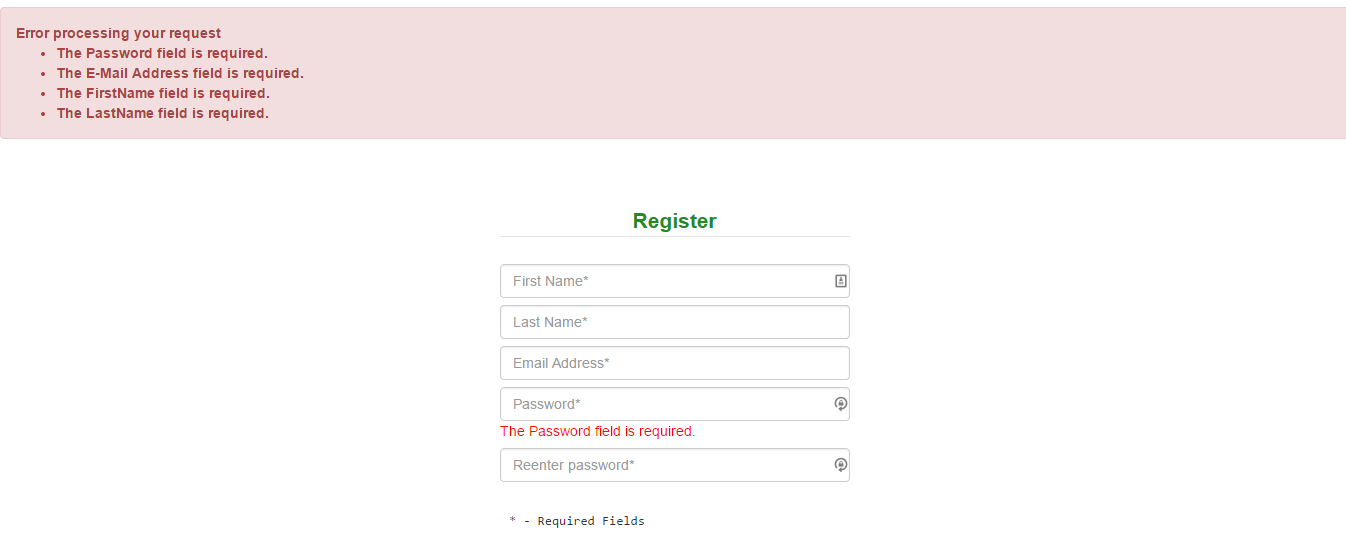


### 4.6.2 Register

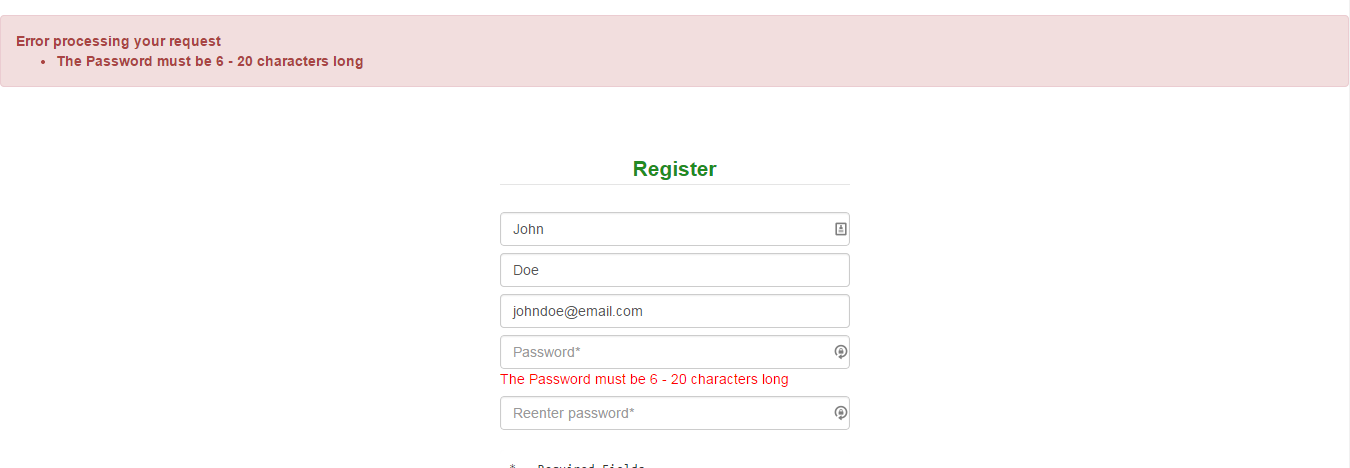
#### 4.6.2.1 Empty register form



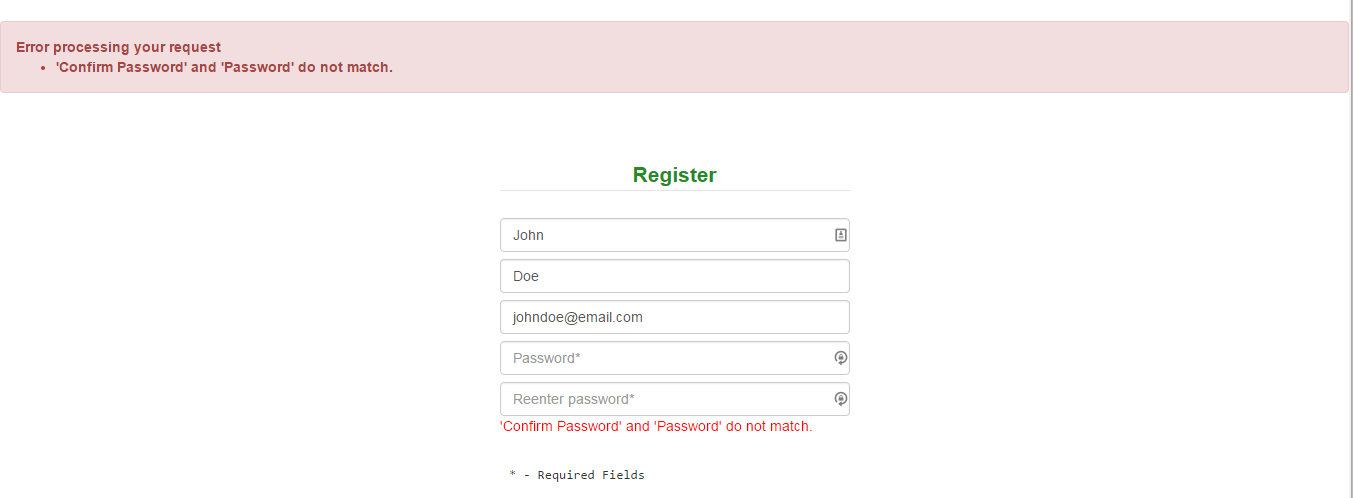
#### 4.6.2.2 Empty fields error



#### 4.6.2.3 Password length error

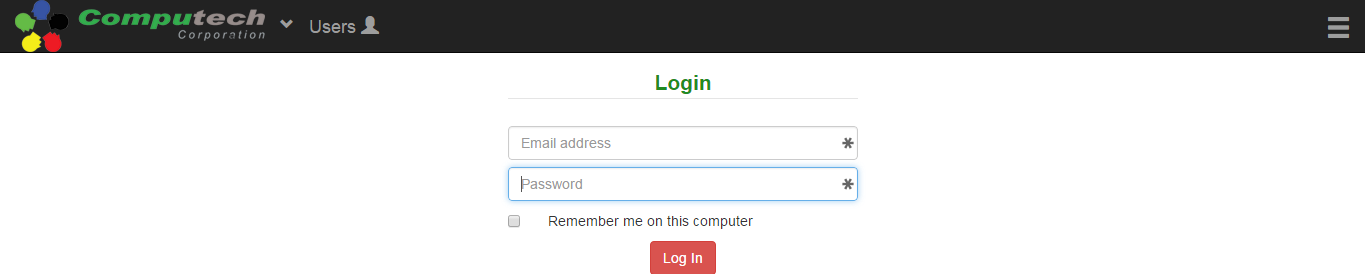


#### 4.6.2.4 Password and confirm password do not match error

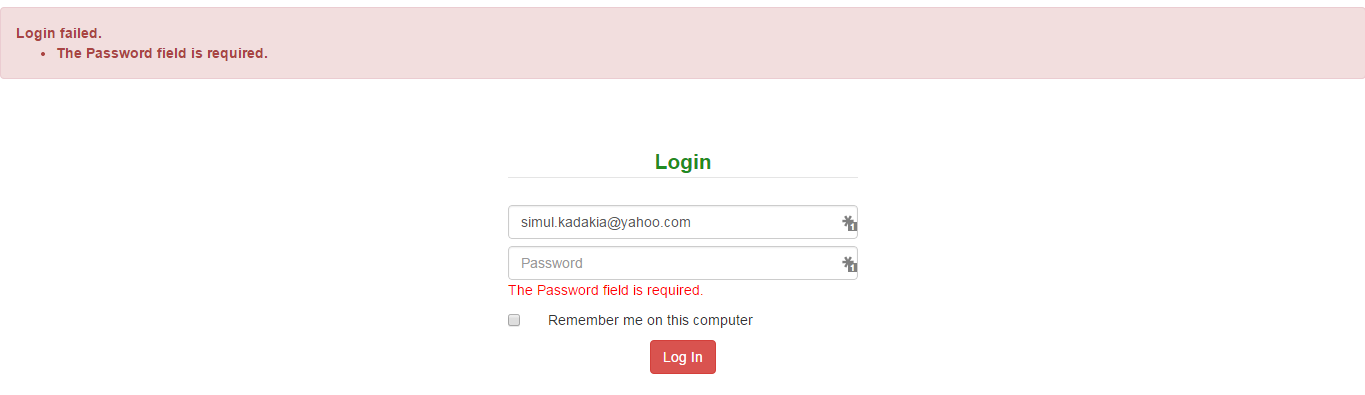


### 4.6.3 Login

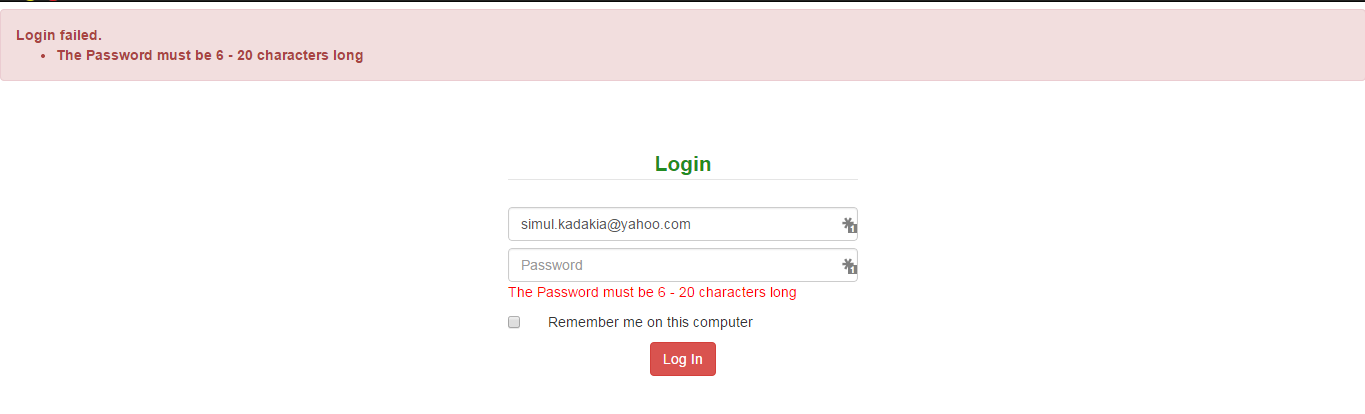
#### 4.6.3.1 Empty login form



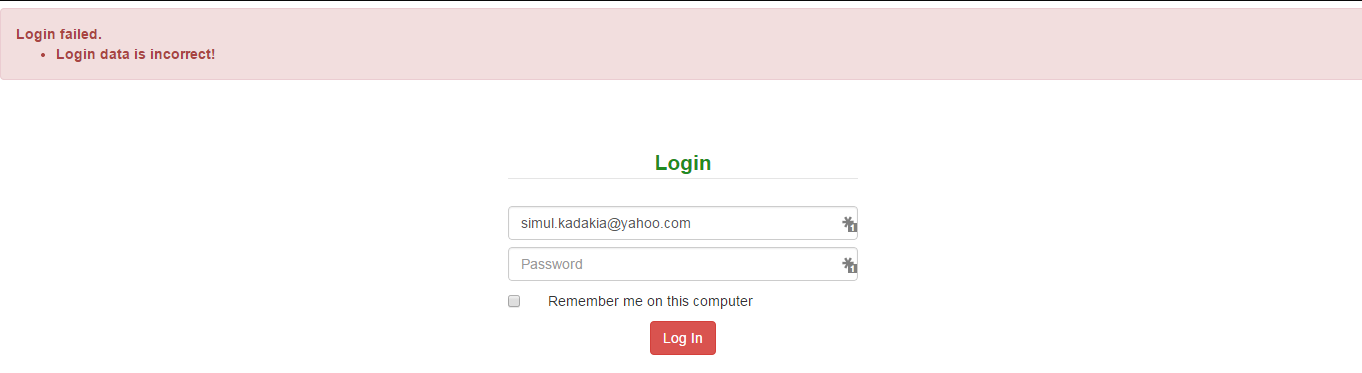
#### 4.6.3.2 Empty Password error



#### 4.6.3.3 Password length error

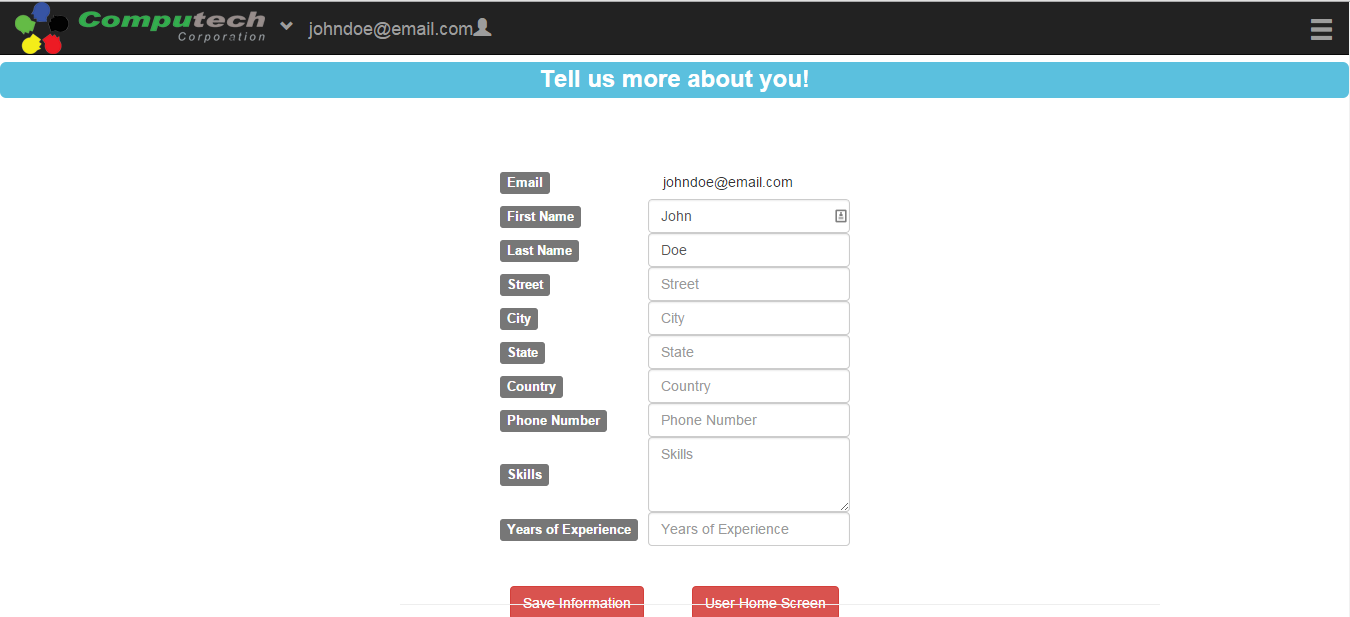


#### 4.6.3.4 Incorrect password

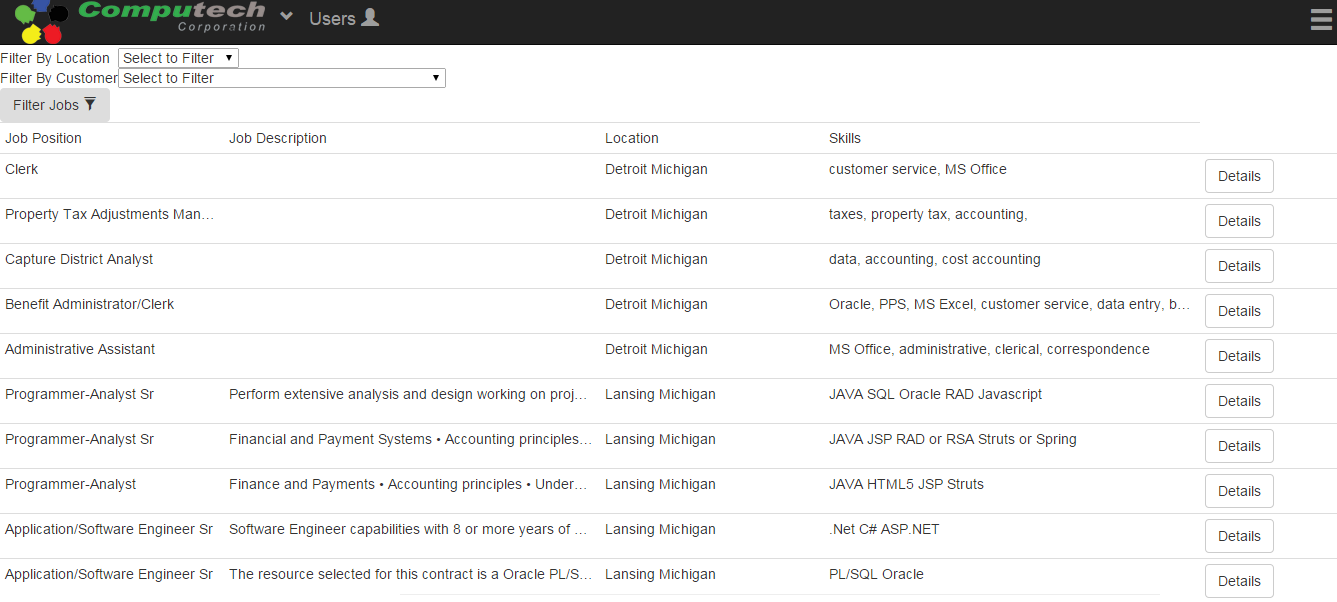


### 4.6.4 Profile Page

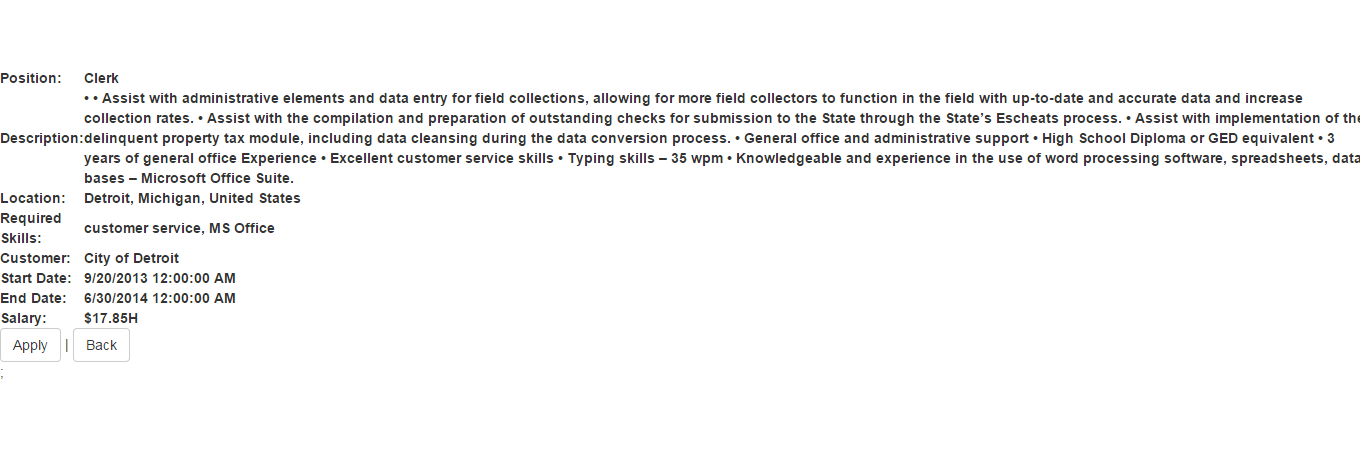
#### 4.6.4.1 Next page (Profile) after successful registration



### 4.6.5 Job Search Page



### 4.6.6 Job Details Page



# 5. Product Design Specification Approval