

Enhanced Job and Candidate Application

Design Specification



Version: 1.0
03/10/2015

Prepared by:

Simul Kadakia
Wesley Trescott
Gagandeep Singh

Contents

1. Introduction	1
1.1 Purpose	1
1.2 Scope	1
1.3 Definitions, Acronyms and Abbreviations	1
1.4 References	1
2. Assumptions / Constraints / Standards	2
2.1 Design Constraints:	2
2.2 Assumptions and Dependencies	2
2.3 Components:	2
3. Architecture Design	3
3.1 Logical View	3
3.2 Hardware Architecture	4
3.3 Software Architecture	4
3.3 Security Architecture	5
3.4 Communication Architecture	5
3.5 Performance	6
4. System Design	6
4.1 Use-cases	6
4.1.1 Actors	6
4.1.2 List of use cases	6
4.1.3 Use Case Diagrams	7
4.1.4 User class – Job seeker	8
4.1.5 User class – Admin	18
4.2 Sequence Diagram	22
4.3 Data Flow Diagrams	24
4.4 Database Design	28
4.5 Application Program Interface	28
4.5.1 Google Authentication	28
4.5.2 Software Class Diagrams	29
4.6 User Interface Design	30
4.6.1 Home Page	30
4.6.2 Register	31
4.6.3 Login	33

4.6.4 Profile Page	35
4.6.5 Job Search Page	36
4.6.6 Job Details Page	36
5. Product Design Specification Approval	37

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.

1.1 Purpose

The purpose of this Software Design Specification (SDS) 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 pertaining to the application.

1.2 Scope

Enhanced Job and Search Candidate is an application developed by three 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 the database, which can be used when they are applying for a job using a web interface. The application will also provide functionalities to an admin user 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.
- GUID – Global Unique ID

1.4 References

"Using OAuth 2.0 to Access Google APIs." *Google Developers*. Google Corporation, 21 November 2014. Web. 10 March 2015.

Additional helpful resources not directly referenced in this document:

- 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-\(c\)](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:

As a web application, the largest design constraint is the differences in the rendering of the user interface based on the type of device used. End users will use the application from a modern internet browser such as Safari, Google 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:

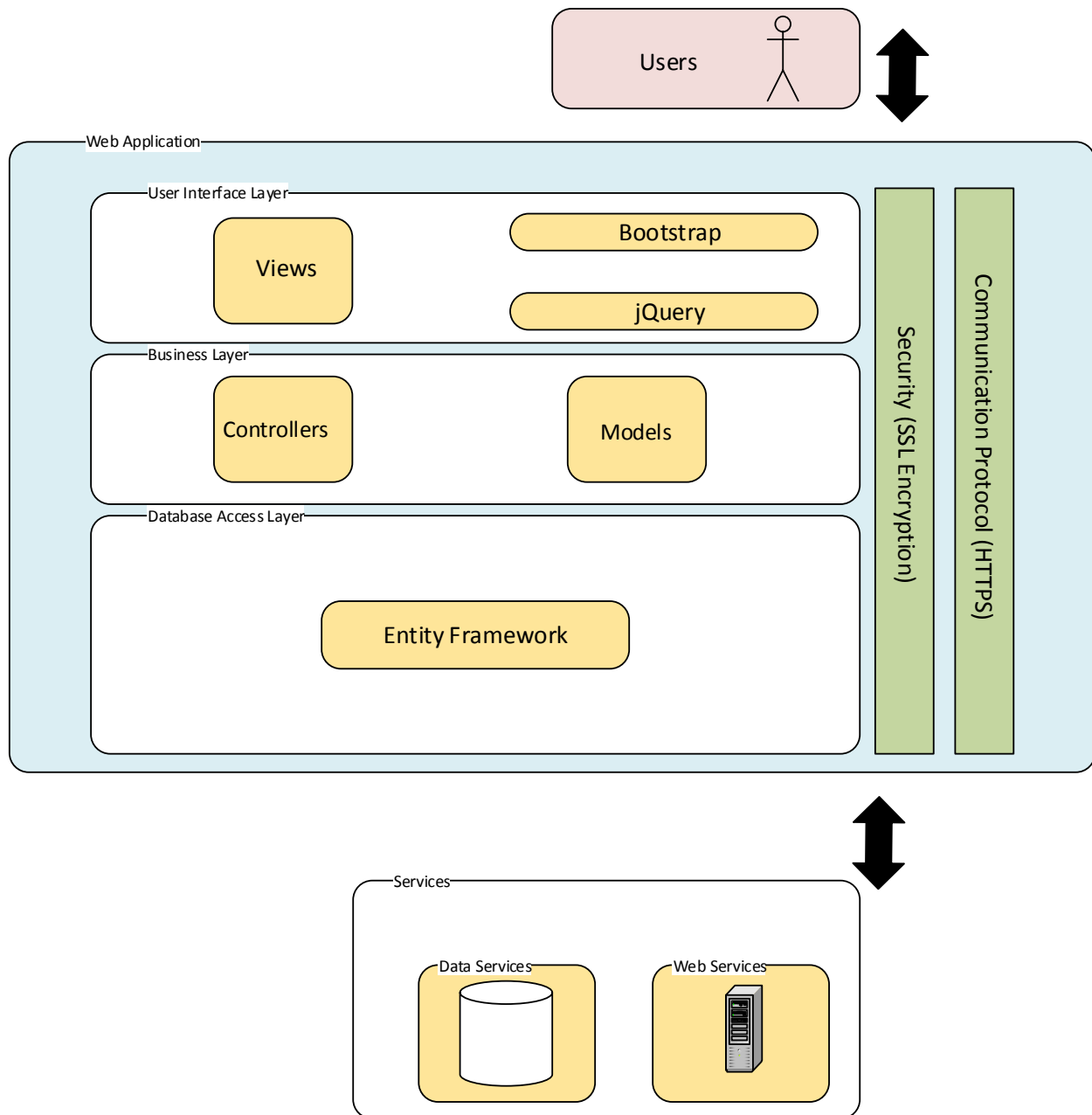
- **Microsoft ASP.NET MVC framework:** EJCA will be developed using MVC, which is a software architectural pattern for implementing user interfaces. It divides a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information 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

This section details the architectural designs of the various components of the (currently under development) EJCA.

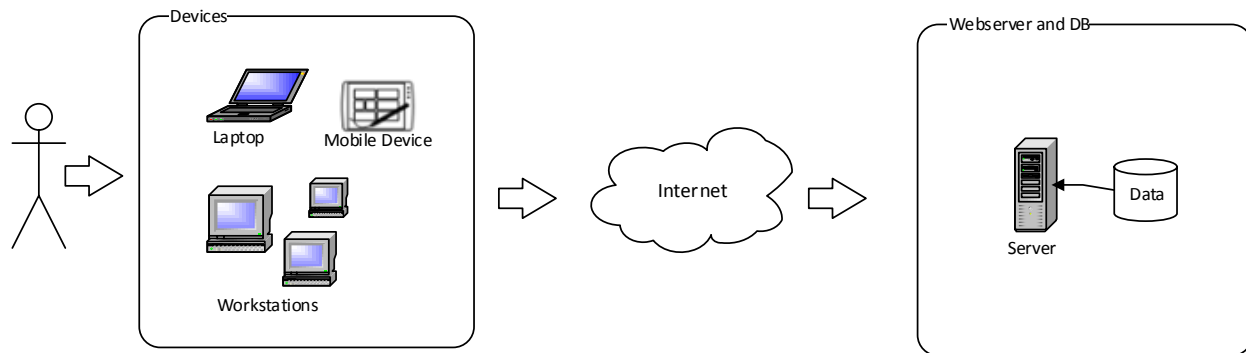
3.1 Logical View

Below is a system-level logic view designed to orient those interested in the web application's logical design to the sections of this document dealing with class, development, process, and physical views of the system architecture. Following are specific subsections on the hardware, software, security, and communication architectures of the system, as well as a section detailing the system architecture's impact on the web application's performance.



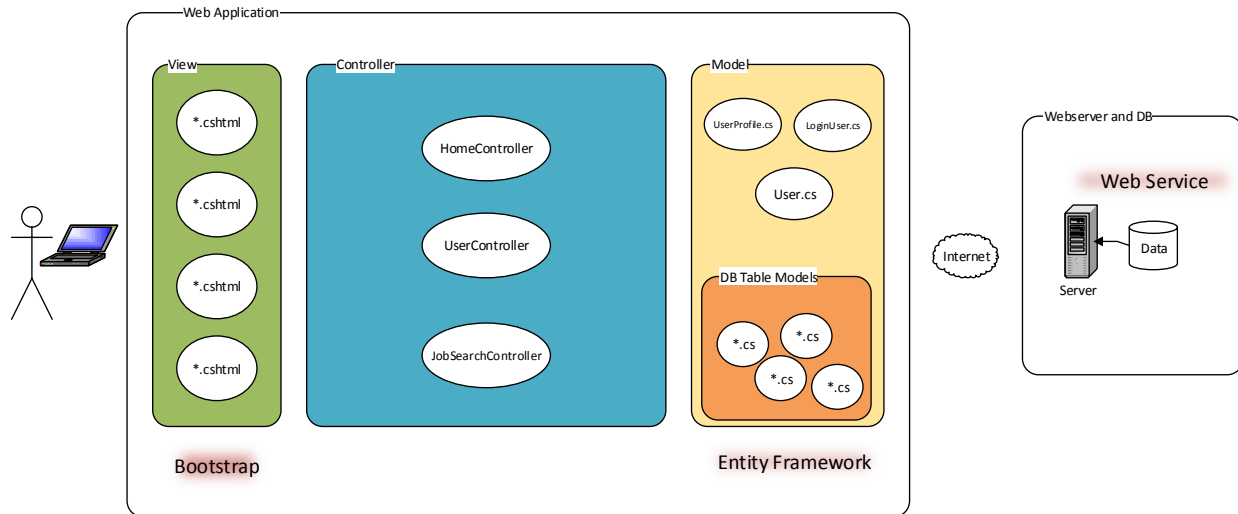
3.2 Hardware Architecture

The hardware architecture of the EJCA was given by the client and consists of two types of hardware interfaces: a webserver and the user's machine. The webserver is the physical machine hosting the site, including the server software it runs as well as the database. The server to be used is a Windows Server 2008 R2 Standard with 16 GB of RAM, and the software it runs is an IIS 7.5 webserver and a Microsoft SQL Server 2008 R2 database. This piece of hardware is owned by Computech and is used to receive HTTP requests and provide HTTP responses, ensuring the constant availability of the application. Additionally, each user of the application will access it using his or her own machine, whether that be in a desktop environment, or through a laptop, smartphone, or tablet. The desktop or laptop devices supported are limited to any device running a modern internet browser with an up to date JavaScript engine and support for HTTP cookies, such as the latest versions of Google Chrome or Safari. The tablet and smartphone devices supported are the iPhone 6 and iPad Air running the iOS 8.1.3 operating system and the Samsung Galaxy S5 running the Android v5.0 Lollipop operating system. Below is a diagram showing an overview of the system hardware architecture.



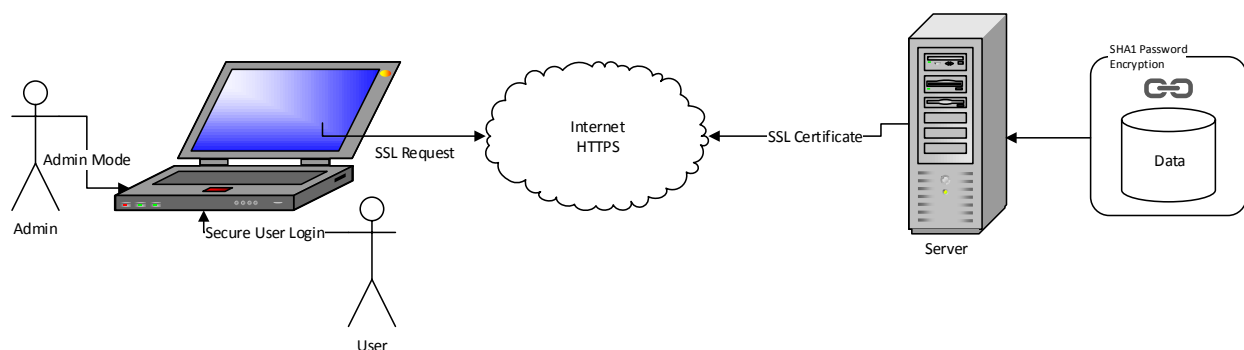
3.3 Software Architecture

EJCA runs on the Model View Controller (MVC) software architecture, a design pattern offered as part of the Microsoft ASP.NET framework. Since this architecture provides a simple three-tier system for displaying pages and managing data, it is ideal for the needs of EJCA. Being a web application, EJCA will also require a web service, which will run on a web server. The EJCA has three controllers, which provide the logic and data processing functionality of the web application. They are the Home controller, handling the logic of the site's homepage, the User controller, handling the logic of all user-related activities, and the JobSearch controller, handling the logic of searching for jobs. Each controller has views it controls, which are displayed to the user as .cshtml web pages. Additionally, the application uses three user model C# classes to interface with the database, accessing and manipulating user data. Also, the application makes use of Microsoft's Entity Framework to provide a model for the jobs database so that users may filter and search for jobs. From a language perspective, the controllers and models are written in C# and the views are written in CSHTML using the Razor syntax. For the frontend user interface, the Bootstrap framework is used both for its professional and modern look and also for its mobile device compatibility. Bootstrap is accessed by using its Cascading Style Sheets (.css files) and JavaScript library, which allow for an easy mobile rendering when screen pixel widths of under 768px are detected. Below is a diagram showing an overview of the system software architecture.



3.3 Security Architecture

The security architecture of the EJCA is composed of four components: the admin mode, user password encryption, login validation, and SSL encryption. The functionality of the admin mode gives the administrator the ability to view and deactivate user accounts, ensuring that users abusing the application will be removed. Also, all user passwords will be encrypted using the SHA1 encryption algorithm before being stored in the database. SHA1 was developed by the NSA and is an example of a cryptographic hash function, considered nearly impossible to decode, thus ensuring the security of user password information from the system administrator. Users will receive an account validation email upon registering an account and will likewise have the ability to reset passwords with email validation, as system administrators will also not have access to passwords. Additionally, user logins are validated so that incorrect user credentials will not result in access to the application, thus maintaining a secure login portal for all users. Also, the Computech webserver, IIS 7.5, uses an SSL certificate to authenticate and encrypt data transmitted between users and the site using HTTPS, so that user data flows securely between the client and server. This is activated for each application use, as the EJCA is coded to request a secure connection channel from the server during the initial handshake. Below is a diagram showing an overview of the system security architecture.



3.4 Communication Architecture

The web application communication architecture consists of the communication between the different components of the software system and the communication between the running application on the

user's device and the Computech server. The software system utilizes Microsoft's ASP.NET MVC architecture, so communication between the model, view, and controller components of the system is dictated by the workings of the framework. Generally, the C# controller contains a C# model object which it queries for data. This data is then passed to the .CSHTML view, where it is rendered for the user to view. Though communication between each of these components is handled via the .NET framework, communication between the application itself and users is handled by the HTTP protocol. The messages passed will be GET (in the case of requesting a web page) and POST (in the case of submitting a form) requests from the user or admin, which will trigger application calls to insert, delete, or select data from the database.

3.5 Performance

This performance of the system architecture provides metrics of how well it operates during user interaction with the software. This includes response time to user logins, registrations, job searches, and admin logins and activities, as well as web site availability. Included in this metric are page transition times, as for example from the home screen to the job search screen. Since database queries and requests in the application deal with either basic user information or job listings being retrieved from or entered into the database, load times will be a factor of how much RAM is available on the local machine and the speed of the server hosting the site. The webserver hosting EJCA has 16 GB of RAM, with a target speed, based on this configuration, of 100 KB per second. Additionally, given that the user machine has 500 MB of available RAM, and internet download speeds of at least 2 MBPS, overall loading quickness in the completed application will be at a maximum of 3 seconds. The system will also be available for use whenever it is run, barring any user internet connectivity problems. This means that Computech's webserver will function properly and that the application will not require periods of unavailability due to maintenance needs.

4. System Design

4.1 Use-cases

4.1.1 Actors

There are two types of actors for EJCA:

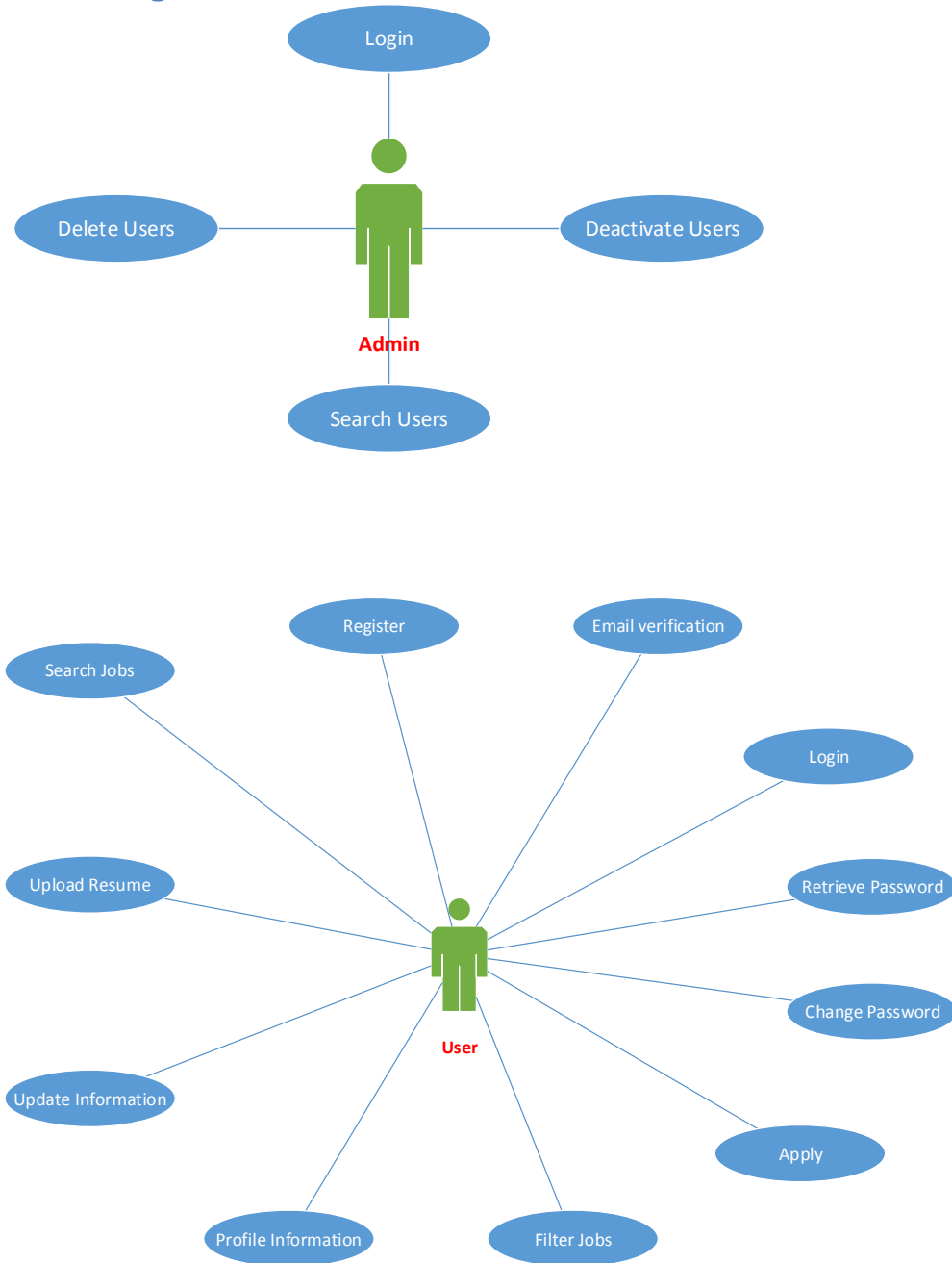
- Users – Job seekers who search for jobs and apply to them.
- Admin – Administrator who manages the users.

4.1.2 List of use cases

- Users:
 1. Search Jobs
 2. Register
 3. Email verification
 4. Login
 5. Forgot Password
 6. Change Password
 7. Apply
 8. Filter by location, customer , skills
 9. Profile Information
 10. Upload Resume

- Admin:
 - 11. Login
 - 12. Search Users by first and last name
 - 13. Deactivate users
 - 14. Delete users

4.1.3 Use Case Diagrams



4.1.4 User class – Job seeker

Use Case ID:	UseCaseID_1		
Use Case Name:	Search jobs		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who can search all available jobs who will be referred as User.		
Description:	This use case describes how a user can search all jobs stored in the database.		
Trigger:	User clicks on Search Jobs button available on both home screen and user dashboard.		
Preconditions:	None.		
Postconditions:	System will display all jobs stored in the database to the user.		
Normal Flow:	<ol style="list-style-type: none">1. User opens website.2. User clicks on Search Jobs button.3. System displays all available jobs.		
Alternative Flows:	None		
Exceptions:	None.		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	None.		

Use Case ID:	UseCaseID_2		
Use Case Name:	Register		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who wants to create an account who will be referred as User.		
Description:	This use case describes registration process.		
Trigger:	User clicks on Register button in the Register page.		
Preconditions:	Email ID has not been registered before.		
Postconditions:	System will store user information in User table in database and send an email verification link to user.		
Normal Flow:	<ol style="list-style-type: none"> 1. User clicks on Register button under Users ddl in the header. 2. User fills in First Name, Last Name, Email, Password and Confirm Password. 3. System displays confirmation page with next steps. 		
Alternative Flows:	None		
Exceptions:	Email ID has been registered before. Fields left empty. (all fields are required during sign-up) Password not between 6-20 characters Password and Confirm Password doesn't match		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	None.		

Use Case ID:	UseCaseID_3		
Use Case Name:	Email verification		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who registered their email address and wants to verify their email, who will be referred as User.		
Description:	This use case describes email verification process.		
Trigger:	User clicks on the link sent to their email.		
Preconditions:	User has registered their email address.		
Postconditions:	Is_Active field in User table will be set to 1 which will allow user to log in.		
Normal Flow:	<ol style="list-style-type: none"> 1. User clicks on the link in the email. 2. System verifies GUID in the link to the one stored in database. 3. User redirected to log in page. 		
Alternative Flows:	None		
Exceptions:	Email ID has not been registered. User has already registered before.		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None		
Assumptions:	Email ID is valid which will ensure verification link is received. Email ID and GUID in the link are not altered.		

Use Case ID:	UseCaseID_4		
Use Case Name:	Login		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who verified their email address and wants to login, who will be referred as User.		
Description:	This use case describes login process.		
Trigger:	User clicks on the Login button in the Login page.		
Preconditions:	User has verified their email address and has not been deactivated by the admin.		
Postconditions:	User will be redirected to User Dashboard screen.		
Normal Flow:	<ol style="list-style-type: none"> 1. User clicks on the login link under the Users dropdown list in the header. 2. User enters Email ID and password and clicks on Log In. 3. System displays User Dashboard screen 		
Alternative Flows:	<p>User clicks on the verification link again (if user has already verified, they will be redirected to login page).</p> <p>User has not logged in but clicks on Apply button in Job Details page.</p>		
Exceptions:	<p>Email ID has not been registered before.</p> <p>Email ID has not been verified.</p> <p>Incorrect password.</p> <p>User has been deactivated by the admin.</p>		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	Email ID has been registered and verified which will set User's Is_Active field set to 1 in the database.		

Use Case ID:	UseCaseID_5		
Use Case Name:	Forgot Password		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who forgot their password and wants to change it, who will be referred as User.		
Description:	This use case describes password retrieval process.		
Trigger:	User clicks on Forgot Password link in the login page.		
Preconditions:	User has registered their Email ID.		
Postconditions:	User will be sent a link to their email to change their password.		
Normal Flow:	<ol style="list-style-type: none">1. User clicks on Forgot Password link on the login page.2. System displays Confirmation page and sends an email to user with a link.3. User clicks on the link.4. System displays Change password page.		
Alternative Flows:	None.		
Exceptions:	For security purpose, there will not be any exception. Instead, the confirmation page will be displayed even if the email address is not found in the database.		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	Email ID has been registered. User has Is_Active field set to 1 in the database.		

Use Case ID:	UseCaseID_6		
Use Case Name:	Change Password		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who wants to change their password, who will be referred as User.		
Description:	This use case describes password changing process.		
Trigger:	User clicks on Change Password button in the Change Password page.		
Preconditions:	User has logged in.		
Postconditions:	System will store their updated password hash in the database.		
Normal Flow:	<ol style="list-style-type: none"> 1. User clicks on Change Password link under the 'hamburger' icon on the header. 2. System displays Change Password page. 3. User enters current password, new password and confirm new password fields. 4. System displays Confirmation page. 		
Alternative Flows:	None.		
Exceptions:	Current password field doesn't match password in the database. New password not between 6-20 characters. New password and confirm new password doesn't match.		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None		
Assumptions:	Password hash matches the hash stored in User table in database.		

Use Case ID:	UseCaseID_7		
Use Case Name:	Apply		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who wants apply to a specific job, who will be referred as User.		
Description:	This use case describes job applying process.		
Trigger:	User clicks on Apply button in the Job Details page.		
Preconditions:	User has logged in.		
Postconditions:	System will store Email ID and Job ID in the Job Application table		
Normal Flow:	<ol style="list-style-type: none"> 1. User clicks on Apply button in the Job Details page. 2. System displays Confirm page. 3. User can alter their information if they wish. 4. User clicks on Submit. 5. System displays Confirmation page. 		
Alternative Flows:	If user has not logged in and clicks on Apply page, they will be redirected to Login page. After validating their credentials, user will be able to apply.		
Exceptions:	User already applied to job. Required fields (first name, last name, street, city, state and country) are left empty.		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None		
Assumptions:	User is applying for the first time.		

Use Case ID:	UseCaseID_8		
Use Case Name:	Filter by skills, location, customer		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who wants filter jobs based on certain criteria, who will be referred as User.		
Description:	This use case describes job filtering process.		
Trigger:	User selects parameters and clicks on Filter button.		
Preconditions:	User selects at least on criteria.		
Postconditions:	System will display jobs matching the criteria.		
Normal Flow:	<ol style="list-style-type: none"> 1. User selects criteria either in dropdown list (location, customer) or text (skills). 2. User clicks on Filter Jobs button. 3. System displays matching jobs. 		
Alternative Flows:	None		
Exceptions:	Jobs matching multiple criteria cannot be found. Jobs matching skills cannot be found.		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	None.		

Use Case ID:	UseCaseID_9		
Use Case Name:	Profile Information		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who wants to store their profile information.		
Description:	This use case describes process to access and store information.		
Trigger:	User selects Save Information button		
Preconditions:	User is logged in.		
Postconditions:	User information will be stored in database and system will display a success message to user.		
Normal Flow:	<ol style="list-style-type: none"> 1. User fills information in textboxes. 2. User clicks on Save Information. 3. System displays success message. 		
Alternative Flows:	None		
Exceptions:	<p>Required fields (first name, last name, street, city, state, country) are left empty.</p> <p>Phone number is not in correct format (ex: 1111111111 is correct format whereas 111-111-1111 is incorrect)</p>		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	User is in Update Profile page which is in the 'hamburger' icon dropdown list in the header.		

Use Case ID:	UseCaseID_10		
Use Case Name:	Upload Resume		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be a job seeker who wants to upload their resume.		
Description:	This use case describes process to upload resume.		
Trigger:	User selects their Upload Resume button.		
Preconditions:	User is logged in.		
Postconditions:	Resume will be stored on the Computech server and its path will be stored in the user database.		
Normal Flow:	<ol style="list-style-type: none"> 1. User clicks on Browse Files. 2. User selects their resume. 3. User clicks on Upload Resume. 		
Alternative Flows:	None.		
Exceptions:	No file was uploaded. Resume file extension is invalid (.exe, .bat, .html)		
Frequency of Use:	This use case may be used hundreds of times in a day, but usually on-demand.		
Special Requirements:	None.		
Assumptions:	User is in Update Profile page which is in the 'hamburger' icon dropdown list in the header.		

4.1.5 User class – Admin

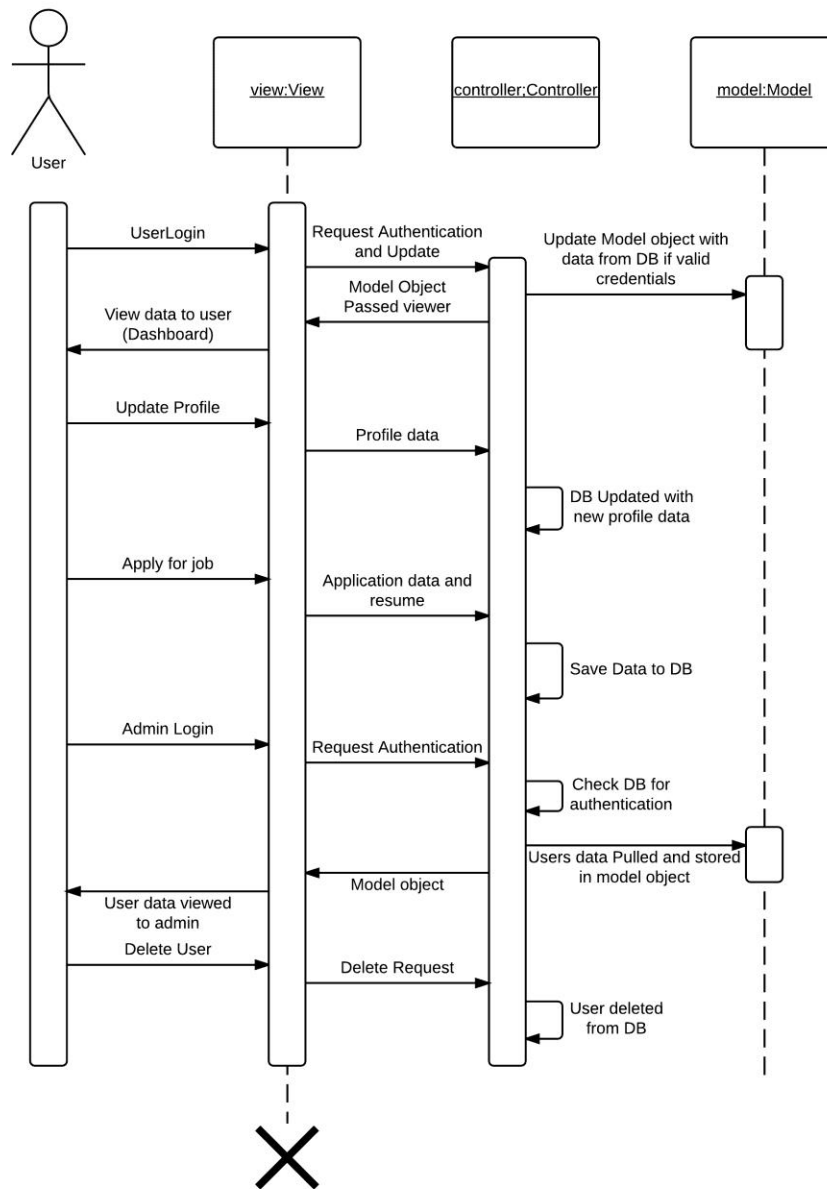
Use Case ID:	UseCaseID_11		
Use Case Name:	Login		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be an administrator who wants to login, who will be referred as Admin.		
Description:	This use case describes login process for admin.		
Trigger:	User clicks on the Login button on the Admin login page.		
Preconditions:	Admin email address is stored in the database.		
Postconditions:	Admin will be redirected to Admin Dashboard screen.		
Normal Flow:	<ol style="list-style-type: none"> 1. Admin clicks on Admin page on the home page. 2. Admin enters Email ID and password and clicks on Log In. 3. System displays Admin Dashboard page. 		
Alternative Flows:	None.		
Exceptions:	Email ID not found in database. Incorrect password.		
Frequency of Use:	This use case may be used once or twice a day.		
Special Requirements:	None.		
Assumptions:	Admin Email ID is stored in the database as there is no option to register new email address for admin.		

Use Case ID:	UseCaseID_12		
Use Case Name:	Search Users		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be an administrator who wants to search users based on their first and/or last name, who will be referred as Admin.		
Description:	This use case describes search user process for admin.		
Trigger:	User clicks on the Search button on the Admin dashboard page.		
Preconditions:	Admin is logged in.		
Postconditions:	System will display list of users matching the criteria.		
Normal Flow:	<ol style="list-style-type: none"> 1. Admin enters user's first and/or last name and clicks on Search. 2. System displays list of all users that matches the criteria. 		
Alternative Flows:	None.		
Exceptions:	User not found in the database.		
Frequency of Use:	This use case may be used once or twice a day.		
Special Requirements:	None.		
Assumptions:	Admin is entering information in at least one textbox.		

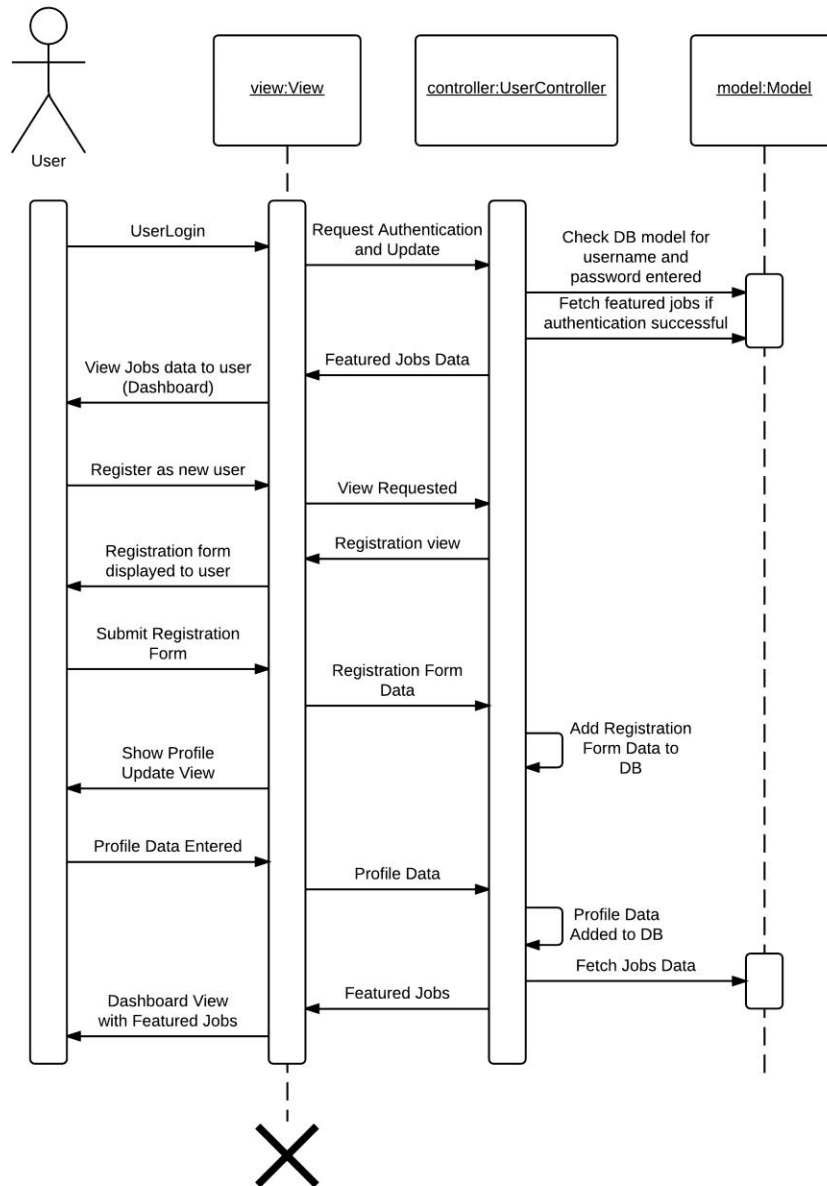
Use Case ID:	UseCaseID_13		
Use Case Name:	Deactivate Users		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be an administrator who wants to deactivate users, who will be referred as Admin.		
Description:	This use case describes user deactivation process for admin.		
Trigger:	User clicks on the Deactivate user link button located on same row as user information.		
Preconditions:	Admin is logged in.		
Postconditions:	System will update the user Is_Active value to 2.		
Normal Flow:	<ol style="list-style-type: none"> 1. Admin clicks on the Deactivate user link button. 2. System displays Success message. 		
Alternative Flows:	None.		
Exceptions:	None.		
Frequency of Use:	This use case may be used once or twice a day.		
Special Requirements:	None.		
Assumptions:	Admin is confident that this particular user is abusing the system as this step cannot be reverted without modifying the database.		

Use Case ID:	UseCaseID_14		
Use Case Name:	Delete Users		
Created By:	Simul Kadakia	Last Updated By:	
Date Created:	03/10/2015	Last Revision Date:	
Actors:	The actor will be an administrator who wants to delete users, who will be referred as Admin.		
Description:	This use case describes user deletion process for admin.		
Trigger:	User clicks on the Delete user link button located on same row as user information.		
Preconditions:	Admin is logged in.		
Postconditions:	System will remove user information from the database.		
Normal Flow:	<ol style="list-style-type: none">1. Admin clicks on the Delete user link button.2. System displays Success message.		
Alternative Flows:	None.		
Exceptions:	None.		
Frequency of Use:	This use case may be used once or twice a day.		
Special Requirements:	None.		
Assumptions:	Admin is confident that this particular user is abusing the system as this step will result in removal of user information from all tables in the database.		

4.2 Sequence Diagram



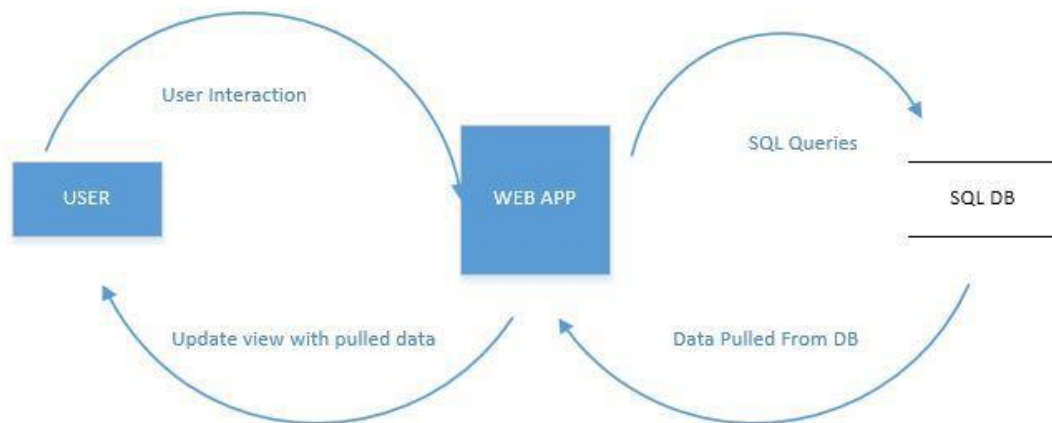
The following System Sequence Diagram covers the user functionality such as login and registration.



4.3 Data Flow Diagrams

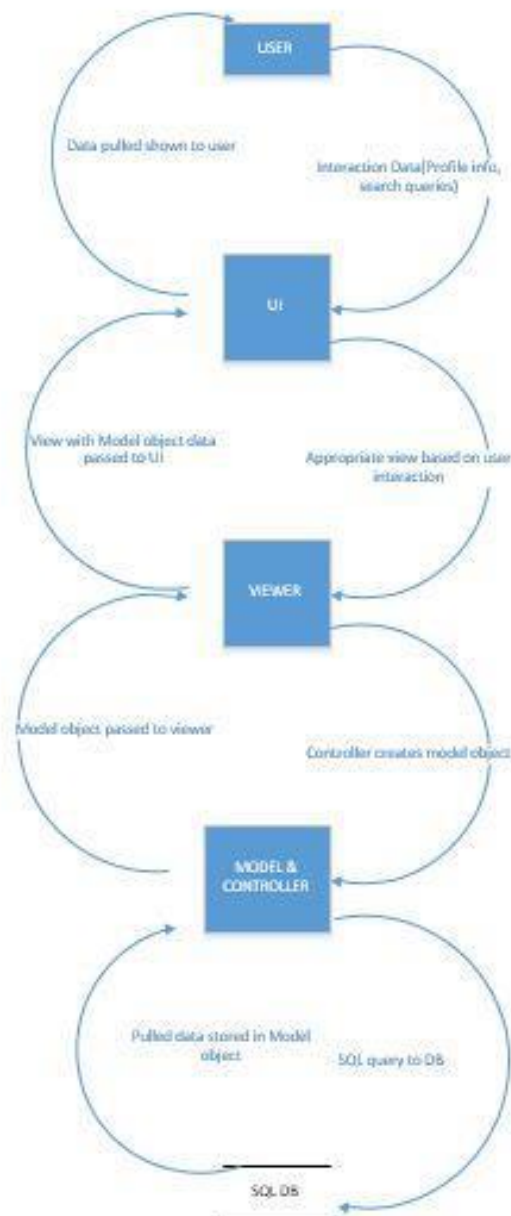
Level 1

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



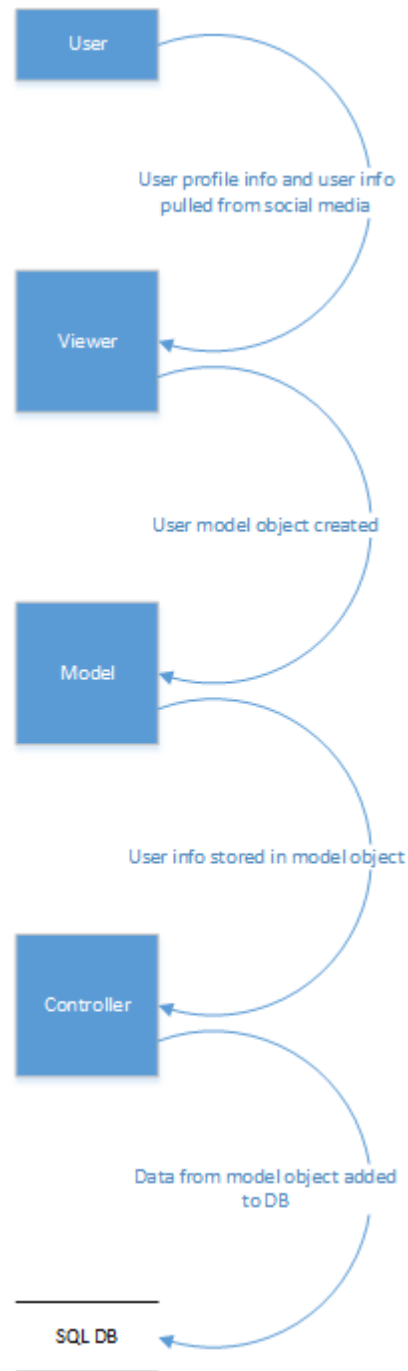
Level 2

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.



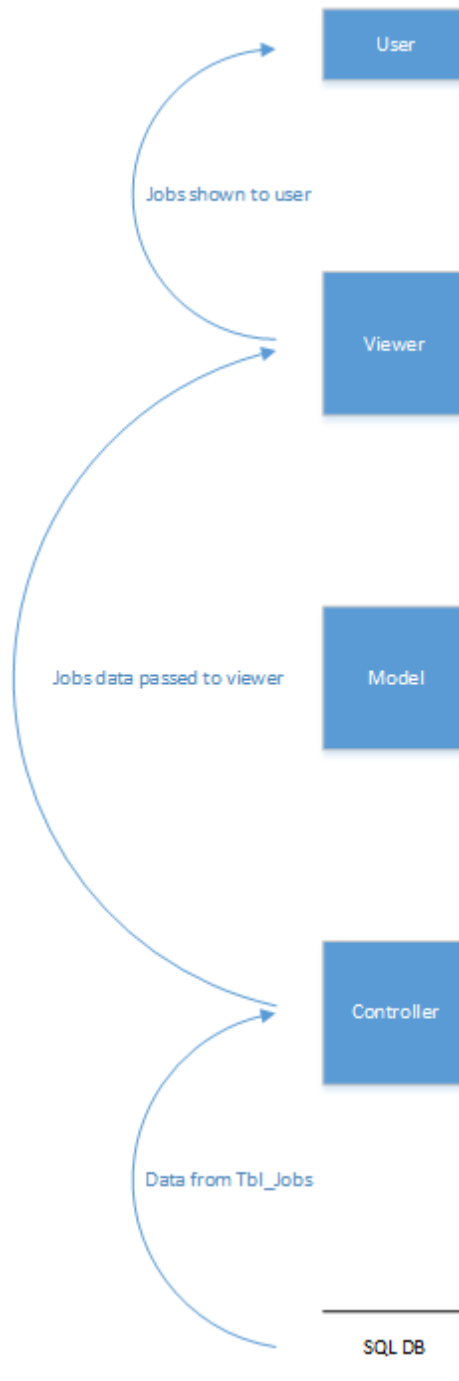
Level 3-1

Shows the flow of user data in EJCA

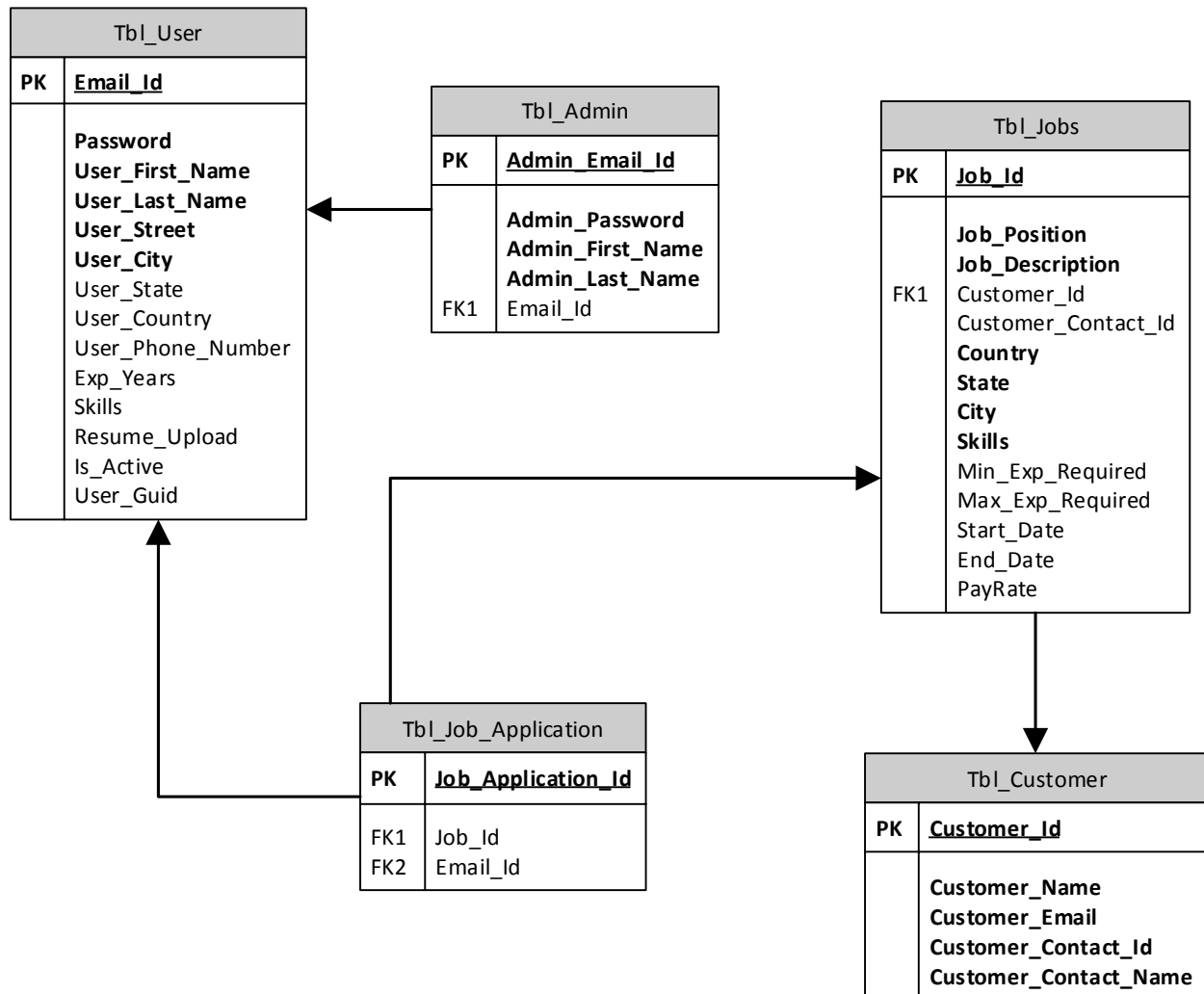


Level 3-2

Shows the flow of jobs data in EJCA



4.4 Database Design



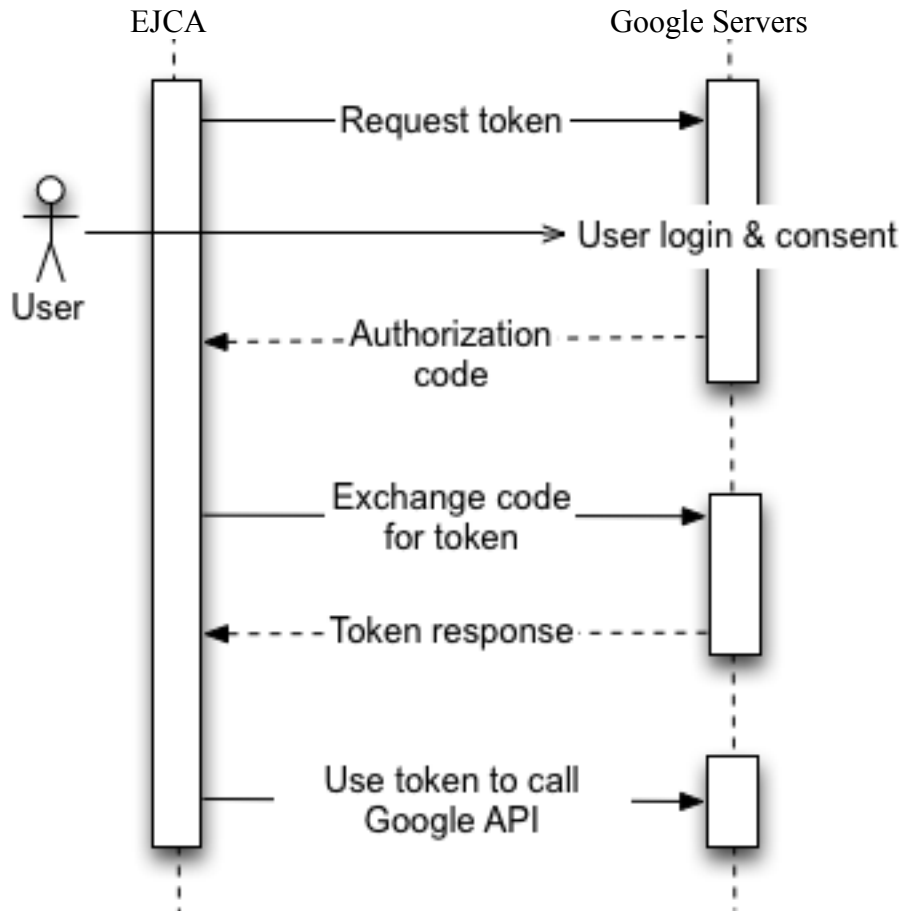
4.5 Application Program Interface

4.5.1 Google Authentication

The user has the ability to log into the EJCA with his/her Google account. This is achieved using the Google API (Microsoft.Owin.Security.Google). The Google API uses the OAuth 2.0 protocol for authentication and authorization. EJCA Web App is registered with Google and OAuth 2.0 credentials were obtained from the Google Developers Console.

When a user selects the option to log in via Google, the user is prompted to log in using Google credentials. After logging in, the user is asked whether he or she is willing to grant the permissions that the application is requesting. This process is called *user consent*.

If the user grants the permission, the Google Authorization Server sends EJCA an access token. If the user does not grant the permission, the server returns an error and the Application goes back to the login page. Once an access token is received, it is passed to the Google API to complete the authentication.



(For image citation, see section 1.4 “References”)

4.5.2 Software Class Diagrams

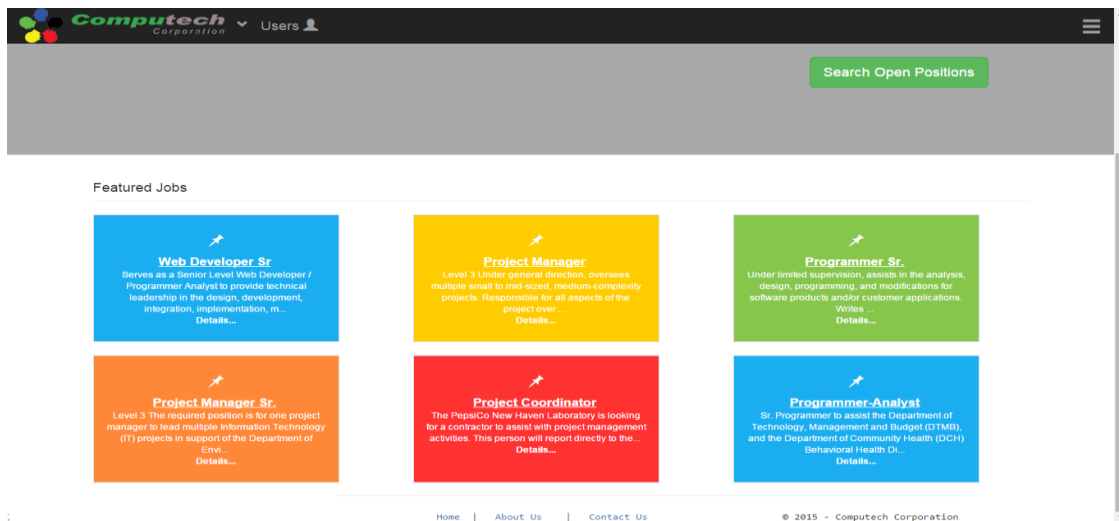
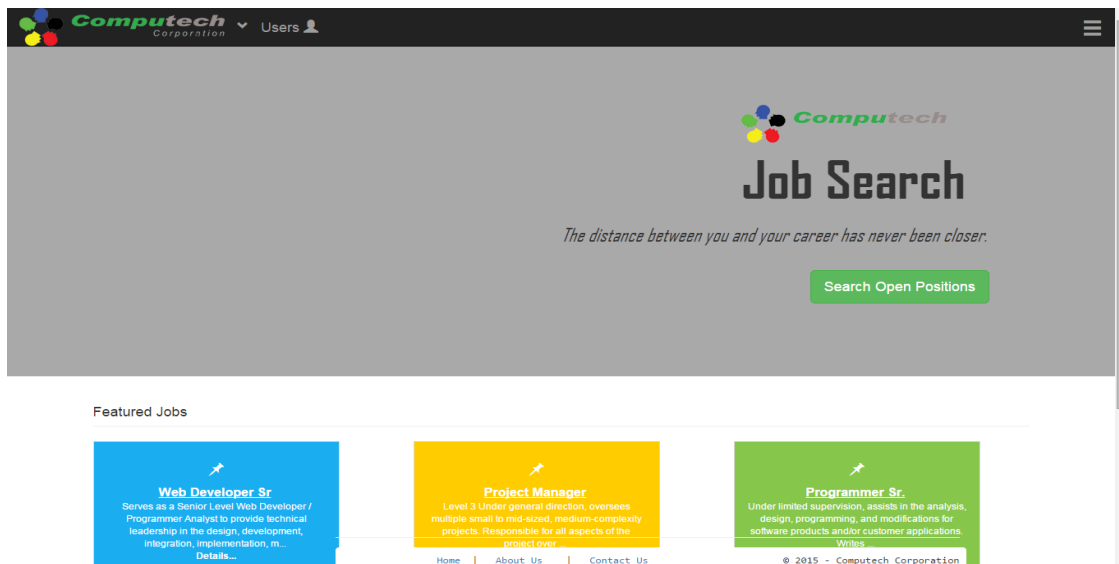
Below is a partial UML class diagram illustrating the design of the controllers in the software system. Note that the lack of relational edged between each class is due to the ASP.NET framework’s low coupling of unrelated software components. Also note that each method is public, and most return an ActionResult associated with a .cshtml web page, allowing the user to access the content of the application. The model classes in the application closely follow the field layout of the database diagrammed above.

```
+ Apply(id : Nullable<Int32>) : ActionResult
+ Details(id : Nullable<Int32>) : ActionResult
+ GetJobDetails(id : Integer) : List<JobDetail>
```


4.6 User Interface Design


4.6.1 Home Page


There will be a fixed header on all the pages which will redirect to different pages. Clicking on Search Open Positions will display Search Jobs page (§4.6.5). Since User is not logged in, clicking on Users will open a dropdown list with Register (§4.6.2) and LogIn (§4.6.3) link. Clicking on Details link on the jobs displayed under Featured jobs will open Job Details page (§4.6.6).




4.6.2 Register

4.6.2.1 Empty register form

 **Computech**
Corporation

Users 



Register








Create My Account


4.6.2.2 Empty fields error

Error processing your request


- The Password field is required.
- The E-Mail Address field is required.
- The FirstName field is required.
- The LastName field is required.

Register





The Password field is required.




* - Required Fields


4.6.2.3 Password length error

Error processing your request


- The Password must be 6 - 20 characters long

Register





The Password must be 6 - 20 characters long






* - Required Fields

4.6.2.4 Password and confirm password do not match error

Error processing your request

- 'Confirm Password' and 'Password' do not match.

Register

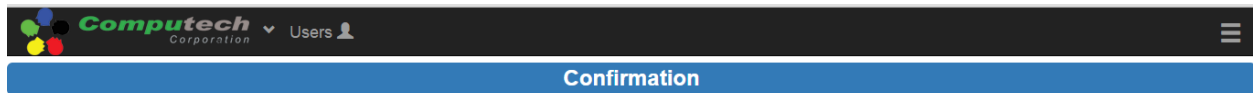
John	
Doe	
johndoe@email.com	
Password*	
Reenter password*	

'Confirm Password' and 'Password' do not match.

* - Required Fields

4.6.2.5 Post registration email confirmation message

Clicking on the register button (§4.6.2) and after success registration, system will redirect user to Confirmation page.



Thank you for registering. You have one more step to complete registration.

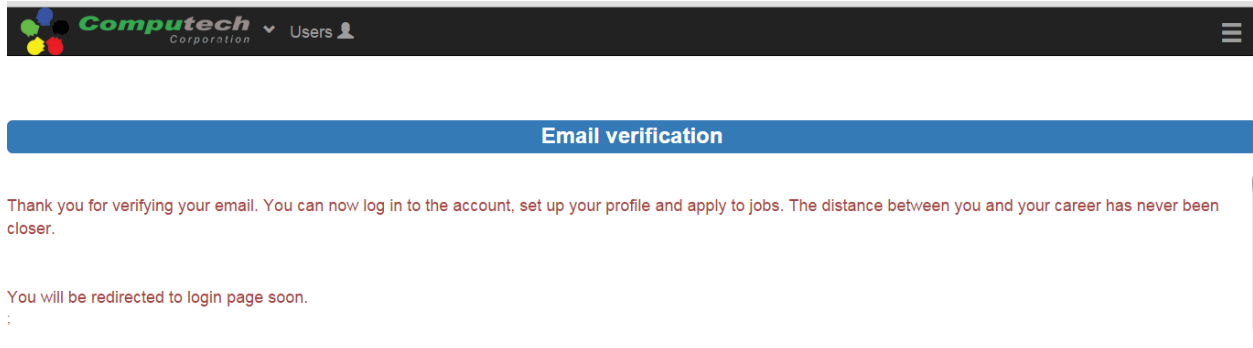
We have sent you an email with registration link. Please check your email account and click on the link to verify your email address.

You will be redirected to home page soon.

;

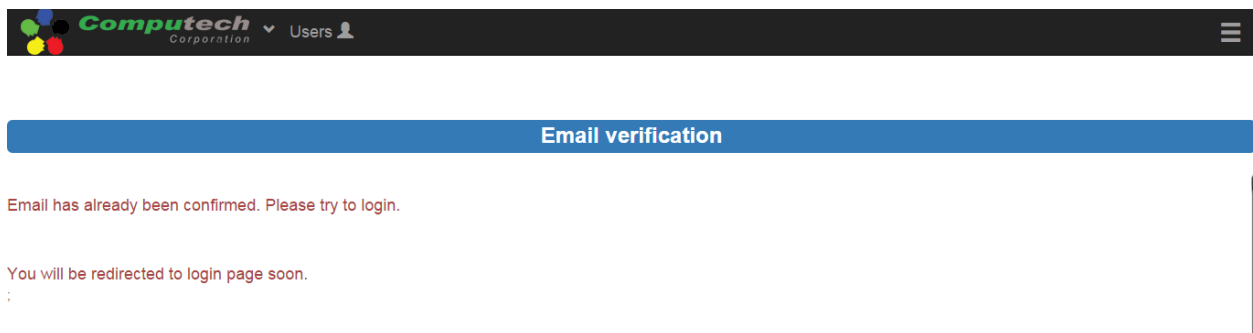
4.6.2.6 Email verification

First time clicking on link sent to the email after registration will redirect user to Verification page and display success message.



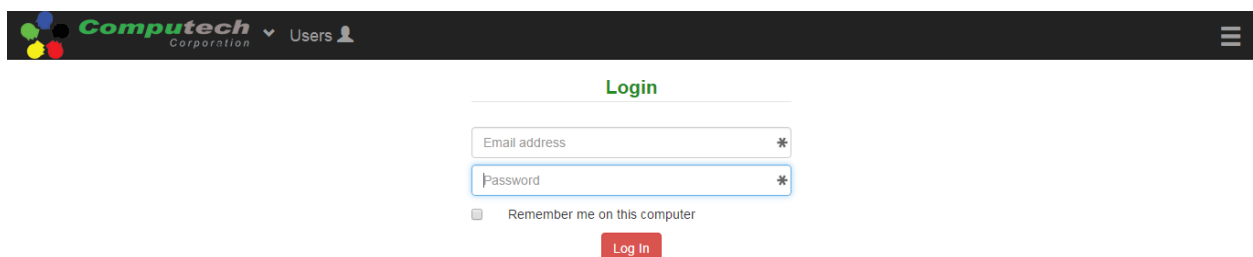
4.6.2.7 Email reconfirmation error

Clicking on link again will redirect user to Verification page but display 'already confirmed' message



4.6.3 Login

4.6.3.1 Empty login form



4.6.3.2 Empty Password error

Login failed.
• The Password field is required.

Login

simul.kadakia@yahoo.com

Password

The Password field is required.

☐ Remember me on this computer

Log In

4.6.3.3 Password length error

Login failed.
• The Password must be 6 - 20 characters long

Login

simul.kadakia@yahoo.com

Password

The Password must be 6 - 20 characters long

☐ Remember me on this computer

Log In

4.6.3.4 Incorrect password/email/login without verifying

Login failed.
• Login data is incorrect!

Login

simul.kadakia@yahoo.com

Password

☐ Remember me on this computer

Log In

4.6.4 Profile Page

4.6.4.1 Update Profile

Computech Corporation johndoe@email.com

Tell us more about you!


Email	johndoe@email.com
First Name	John
Last Name	Doe
Street	Street
City	City
State	State
Country	Country
Phone Number	Phone Number
Skills	Skills
Years of Experience	Years of Experience


[Save Information](#) [User Home Screen](#)


Profile successfully updated!

Email	johndoe@email.com
First Name	John
Last Name	Doe
Street	42 W Warren Ave
City	Detroit
State	MI
Country	USA
Phone Number	1234567890
Skills	C#, ASP.NET, SQL
Years of Experience	3

4.6.5 Job Search Page



Users 




Filter By Location

Select to Filter

Filter By Customer

Select to Filter

Filter Jobs 

Job Position	Job Description	Location	Skills	
Clerk		Detroit Michigan	customer service, MS Office	<div>Details</div>
Property Tax Adjustments Man...		Detroit Michigan	taxes, property tax, accounting,	<div>Details</div>
Capture District Analyst		Detroit Michigan	data, accounting, cost accounting	<div>Details</div>
Benefit Administrator/Clerk		Detroit Michigan	Oracle, PPS, MS Excel, customer service, data entry, b...	<div>Details</div>
Administrative Assistant		Detroit Michigan	MS Office, administrative, clerical, correspondence	<div>Details</div>
Programmer-Analyst Sr	Perform extensive analysis and design working on proj...	Lansing Michigan	JAVA SQL Oracle RAD Javascript	<div>Details</div>
Programmer-Analyst Sr	Financial and Payment Systems • Accounting principles...	Lansing Michigan	JAVA JSP RAD or RSA Struts or Spring	<div>Details</div>
Programmer-Analyst	Finance and Payments • Accounting principles • Under...	Lansing Michigan	JAVA HTML5 JSP Struts	<div>Details</div>
Application/Software Engineer Sr	Software Engineer capabilities with 8 or more years of ...	Lansing Michigan	.Net C# ASP.NET	<div>Details</div>
Application/Software Engineer Sr	The resource selected for this contract is a Oracle PL/S...	Lansing Michigan	PL/SQL Oracle	<div>Details</div>

4.6.6 Job Details Page

Position: Clerk

- Assist with administrative elements and data entry for field collections, allowing for more field collectors to function in the field with up-to-date and accurate data and increase collection rates.
- Assist with the compilation and preparation of outstanding checks for submission to the State through the State's Escheats process.
- Assist with implementation of the delinquent property tax module, including data cleansing during the data conversion process.
- General office and administrative support
- High School Diploma or GED equivalent
- 3 years of general office Experience
- Excellent customer service skills
- Typing skills – 35 wpm
- Knowledgeable and experience in the use of word processing software, spreadsheets, data bases – Microsoft Office Suite.

Description:

Location: Detroit, Michigan, United States

Required Skills: customer service, MS Office

Customer: City of Detroit

Start Date: 9/20/2013 12:00:00 AM

End Date: 6/30/2014 12:00:00 AM

Salary: \$17.85H

Apply

Back

5. Product Design Specification Approval

The undersigned acknowledge they have reviewed the EJCA Product Design Specification document and agree with the approach it presents. Any changes to this Requirements definition will be coordinated with and approved by the undersigned or their designated representatives.

Signature:	_____	Date:	_____
Print Name:	_____		
Title:	_____		
Role:	_____		

Signature:	_____	Date:	_____
Print Name:	_____		
Title:	_____		
Role:	_____		

Signature:	_____	Date:	_____
Print Name:	_____		
Title:	_____		
Role:	_____		