***Enhanced Job and Candidate Application***

***Test Plan***

computech logo



Version: 1.0

03/24/2015

**Prepared by:**

Simul Kadakia

Wesley Trescott

Gagandeep Singh

Contents

[1. Introduction 3](#_Toc414648845)

[1.1 Purpose 3](#_Toc414648846)

[1.2 Scope 3](#_Toc414648847)

[1.3 Definitions, Acronyms and Abbreviations 3](#_Toc414648848)

[1.4 References 3](#_Toc414648849)

[2. Compatibility Testing 4](#_Toc414648850)

[2.1 Test Risks / Issues 4](#_Toc414648851)

[2.2 Items To Be Tested 4](#_Toc414648852)

[2.3 Test Approach 4](#_Toc414648853)

[2.4 Test Pass / Fail Criteria 5](#_Toc414648854)

[2.5 Test Entry / Exit Criteria 5](#_Toc414648855)

[2.6 Test Deliverables 6](#_Toc414648856)

[2.7 Test Suspension / Resumption Criteria 10](#_Toc414648857)

[2.8 Test Environmental Needs 10](#_Toc414648858)

[3. Compatibility Testing 10](#_Toc414648859)

[3.1 Test Risks / Issues 10](#_Toc414648860)

[3.2 Items To Be Tested 10](#_Toc414648861)

[3.3 Test Approach 11](#_Toc414648862)

[3.4 Test Pass / Fail Criteria 11](#_Toc414648863)

[3.5 Test Entry / Exit Criteria 11](#_Toc414648864)

[3.6 Test Deliverables 12](#_Toc414648865)

[3.7 Test Suspension / Resumption Criteria 21](#_Toc414648866)

[3.8 Test Environmental Needs 21](#_Toc414648867)

# 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 Test Plan document is to provide a detailed description of the testing plan of the ‘Enhanced Job and Candidate Application’ system. It will also provide specific information about the compatibility, functional, non-functional, integration, system and user acceptance testing.

## 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.

## 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

## 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>

# Compatibility Testing

## 2.1 Test Risks / Issues

There are no significant risks to the compatibility testing.

## 2.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| CT\_1 | Search Jobs | Users are able to search available job listing. |
| CT\_2 | Register | Users are able to create an account. |
| CT\_3 | Email verification | Users are able to open the link sent in the verification email. |
| CT\_4 | Login | Users are able to login. |
| CT\_5 | Forgot Password | Users are able to request link and change their password. |
| CT\_6 | Change Password | Users are able to change their password after logging in. |
| CT\_7 | Filter Jobs | Users are able to select criteria and filter jobs. |
| CT\_8 | Profile Information | Users are able to access their profile page and update their information. |
| CT\_9 | Apply | Users are able to apply to jobs. |
| CT\_10 | Resume Upload | Users are able to upload their resume. |

## 2.3 Test Approach

The User Interface will be one of the first features to be completed on the project and therefore will be among the first set of tests executed. The test cases for compatibility will be run manually through user interaction.

As outlined in the requirements document, the EJCA application must be functional on Chrome browser\* on iPhone 6 and iPad Air with iOS 8.1.3, Samsung Galaxy S5 with Android 5.0 (Lollipop). It should also be accessible on IE\*, Firefox\* and Safari\* browsers on desktop/laptops.

\*- Browsers should be compatible with up-to-date JavaScript engine.

We will test on hardware devices that are available to us.

|  |  |  |
| --- | --- | --- |
| Device | Operating System | Browsers |
| Laptop / Desktop | * Windows * OS X | * Chrome * Firefox * Safari * Internet Explorer > 9 |
| Tablets   * iPad Air | * iOS 8.1.3 | * Safari |
| Phone   * iPhone 6 * Samsung Galaxy S5 | * iOS 8.1.3 * Android v5.0 (Lollipop) | * Safari * Chrome |

## 2.4 Test Pass / Fail Criteria

Pass when 100% of test cases completed

* Fully working in Chrome on all devices listed above.
* Fully working in IE, Firefox and Safari on desktop/laptop.

Fail on any found fault with UI compatibility except for CT\_10 (Resume Upload) which is compatible only on desktop/laptop environment.

## 2.5 Test Entry / Exit Criteria

Entry Criteria:

* Verify that the Test devices are ready to use.
* Verify if the proper browsers are installed with up-to-date JavaScript engine and ready to use.

Exit criteria:

* All tests have been run successfully and no high priority bugs are left uncorrected.

## 2.6 Test Deliverables

|  |  |
| --- | --- |
| ID | CT\_1 |
| Item to Test | Search Jobs |
| Pre-Conditions | None |
| Test Steps | Repeat the below process for each supported Device and Browser type     1. Click on **Search Open Positions** button |
| Expected Results | System should display all available jobs to the user. |
| Priority | Medium |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_2 |
| Item to Test | Register |
| Pre-Conditions | None |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on Users dropdown list on the header. 2. Click on Register link. 3. Fill in First Name, Last Name, Email Address, Password and Confirm Password. 4. Click on Register |
| Expected Results | System should send an email to user containing email verification link and display confirmation page to user. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_3 |
| Item to Test | Email Verification |
| Pre-Conditions | User has created an account |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on link in the email received. |
| Expected Results | System should display verification confirmation page to user and redirect user to login page within 7 seconds. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_4 |
| Item to Test | Login |
| Pre-Conditions | User has verified their email address |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on Users dropdown list in the header. 2. Click on Sign In link. 3. Fill in Email address and password 4. Click on **LogIn** button. |
| Expected Results | System should redirect user to User dashboard |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_5 |
| Item to Test | Forgot Password |
| Pre-Conditions | User has created an account before. |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on Users dropdown list in the header. 2. Click on Sign In link. 3. Click on **Click here if you forgot your password** link. 4. Enter email address in the textbox on the Forgot Password page 5. Click on **Submit** button |
| Expected Results | System should send an email to user containing change password link and display confirmation page. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_6 |
| Item to Test | Change has create an account. |
| Pre-Conditions | None |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on link in the **forgot password** email. 2. Enter password and confirm password. 3. Click on **Submit** button. |
| Expected Results | System should display confirmation message. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_7 |
| Item to Test | Filter Jobs |
| Pre-Conditions | None |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on link **Search Open Positions** button 2. Select criteria from City and/or Customer 3. Click on **Filter Jobs**  button. |
| Expected Results | System should display jobs matching the criteria. |
| Priority | Medium |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_8 |
| Item to Test | Profile Information |
| Pre-Conditions | User is logged in |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on link hamburger menu button on the header. 2. Click on **Update Profile** link. 3. Change information in any text box.   (First Name, Last Name, Street, City, State, Country are required fields)   1. Click on **Save Information** button. |
| Expected Results | System should display confirmation message. |
| Priority | Medium |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_9 |
| Item to Test | Apply |
| Pre-Conditions | User is logged in |
| Test Steps | Repeat the below process for each supported Device and Browser type   1. Click on **Details** button (either in JobSearch page, Featured Jobs or Recommended Jobs) 2. Click on **Apply** button. 3. Verify Information. 4. Click on **Submit** button. |
| Expected Results | System should display confirmation page. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_10 |
| Item to Test | Resume Upload |
| Pre-Conditions | User is logged in |
| Test Steps | 1. Click on link hamburger menu button on the header. 2. Click on **Update Profile** link. 3. Click on **Browse File** button. 4. Select any file with pdf, doc or txt extension. 5. Click on **Upload Resume** button. |
| Expected Results | System should display confirmation message. |
| Priority | Medium |
| Pass/Fail |  |

## 2.7 Test Suspension / Resumption Criteria

Testing will be suspended as soon as 2 bugs are found and find solution to fix the bug. Once implemented, testing will resume from the beginning.

## 2.8 Test Environmental Needs

Testing will be completed on actual hardware devices as well as emulator on Chrome browser. Actual hardware devices are desktop and laptop. IPad Air, iPhone 6 and Samsung Galaxy S5 will be emulated on Chrome browser.

# Compatibility Testing

## 3.1 Test Risks / Issues

Unit test code needs to be thorough as bug in test code will not detect actual bug in the code and might display incorrect bugs.

## 3.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| FT\_1 (\_) | Register | Call User.Register() to create user account |
| FT\_2 (\_) | Email verification | Call User.Confirmed() to verify user email. |
| FT\_3 (\_) | Login | Call LoginUser.IsValid() to verify credentials. |
| FT\_4 (\_) | Forgot Password | Call EmailManager.SendForgotPasswordEmail() to send an email to user. |
| FT\_5 (\_) | Change Password | Call ChangePassword.UpdatePassword() to update password in database. |
| FT\_6 (\_) | Apply | Call Apply.SubmitApplication() to apply to a job. |
| FT\_7 (\_) | Job Details | Call Details() in JobSearchController to display details of a job. |
| FT\_8 (\_) | Filter Jobs | Call Index() in JobSearchController to filter jobs based on parameters. |

## 3.3 Test Approach

Unit test cases will be written to test each module in all of the model classes. The test cases will include both valid and invalid input to test classes such as User, LoginUser, UserProfile, ChangePassword, ForgotPassword and Apply. JobSearch controller will also be tested to verify whether the jobs are displayed and filtered based on criteria.

## 3.4 Test Pass / Fail Criteria

Depending on the model class, if any of the tests fails, testing for that class will be suspended, alternate solution will be implemented and then, testing will be restarted from the beginning.

## 3.5 Test Entry / Exit Criteria

Entry Criteria:

* Test environment is ready to use.
* Adequate test data is available.
* Test plans are completed.

Exit criteria:

* All modules in class are tested.
* All test cases pass successfully.
* Defects found are documented to find alternate solution.

## 3.6 Test Deliverables

|  |  |
| --- | --- |
| ID | FT\_1.1 |
| Item to Test | Register |
| Pre-Conditions | None |
| Test Steps | 1. Call User.Register(firstName, lastName, email, password, guid)   {  firstName = “John”,  lastName = “Doe”,  email = “john.doe@email.com”,  password = “123456”,  GUID = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | True should be returned |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_1.2 |
| Item to Test | RegisterFails |
| Pre-Conditions | None |
| Test Steps | 1. Call User.Register(firstName, lastName, email, password, guid)   {  firstName = “”,  lastName = “”,  email = “”,  password = “”,  GUID = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | False should be returned |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_1.3 |
| Item to Test | RegisterFails |
| Pre-Conditions | Account already created with email “john.doe@email.com” |
| Test Steps | 1. Call User.Register(firstName, lastName, email, password, guid)   {  firstName = “John”,  lastName = “Doe”,  email = john.doe@email.com,  password = “123456”,  GUID = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | False should be returned |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_2.1 |
| Item to Test | Email verification |
| Pre-Conditions | Account already created with email “john.doe@email.com” |
| Test Steps | 1. Call User.Confirmed(email)   {  email = “john.doe@email.com”  } |
| Expected Results | Email should be verified with message "Thank you for verifying your email. You can now log in to the account, set up your profile and apply to jobs. The distance between you and your career has never been closer." |
| Priority | True should be returned. |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_2.2 |
| Item to Test | EmailVerificationFails |
| Pre-Conditions | Account already created with email “john.doe@email.com” |
| Test Steps | 1. Call User.Confirmed(email)   {  email = “”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_3.1 |
| Item to Test | Login |
| Pre-Conditions | Email has been verified |
| Test Steps | 1. Call LoginUser.IsValid (email, password)   {  email = “john.doe@email.com”,  password = “123456”  } |
| Expected Results | Login should be successful and user redirected to LoggedIn view. |
| Priority | True should be returned. |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_3.2 |
| Item to Test | LoginFails |
| Pre-Conditions | Email has been verified |
| Test Steps | 1. Call LoginUser.IsValid (email, password)   {  email = “”,  password = “”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_3.3 |
| Item to Test | LoginFails |
| Pre-Conditions | Email has been verified |
| Test Steps | 1. Call LoginUser.IsValid (email, password)   {  email = “”,  password = “123456”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_3.4 |
| Item to Test | Fails |
| Pre-Conditions | Email has been verified |
| Test Steps | 1. Call LoginUser.IsValid (email, password)   {  email = “john.doe@email.com”,  password = “123456abc”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_4.1 |
| Item to Test | Forgot Password |
| Pre-Conditions | User account has been created |
| Test Steps | 1. Call EmailManager.SendForgotPasswordEmail(firstName, email, guid)   {  firstName = “John”,  email = “john.doe@email.com”,  guid = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | System sends an email and message “We sent an email with link to change the password. Please check your email.” is displayed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_5.1 |
| Item to Test | Change Password |
| Pre-Conditions | User account has been created |
| Test Steps | 1. Call User.UpdatePassword(email, currentPassword, password, guid){   email = “john.doe@email.com”,  currentPassword = “”,  password = “123abc”,  guid = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | True should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_5.2 |
| Item to Test | Change Password Fails |
| Pre-Conditions | User has logged in |
| Test Steps | 1. Call User.UpdatePassword(email, currentPassword, password, guid){   email = “john.doe@email.com”,  currentPassword = “123456”,  password = “123abc”,  guid = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | True should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_5.3 |
| Item to Test | Change Password Fails |
| Pre-Conditions | User has logged in |
| Test Steps | 1. Call User.UpdatePassword(email, currentPassword, password, guid){   email = “john.doe@email.com”,  currentPassword = “111111”,  password = “123abc”,  guid = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_5.4 |
| Item to Test | Change Password Fails |
| Pre-Conditions | User has logged in |
| Test Steps | 1. Call User.UpdatePassword(email, currentPassword, password, guid){   email = “john.doe@email.com”,  currentPassword = “”,  password = “”,  guid = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_5.5 |
| Item to Test | Change Password Fails |
| Pre-Conditions | User has logged in |
| Test Steps | 1. Call User.UpdatePassword(email, currentPassword, password, guid){   email = “”,  currentPassword = “”,  password = “123abc”,  guid = “a6d7f3b6-c65d-4319-89d9-e5b518720e39”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_6.1 |
| Item to Test | Apply |
| Pre-Conditions | User has logged in |
| Test Steps | 1. Call Apply.submitApplication(email, jobId){   email = “john.doe@email.com”,  jobId = “17604”  } |
| Expected Results | True should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_6.2 |
| Item to Test | ApplyFails |
| Pre-Conditions | User has logged in |
| Test Steps | 1. Call Apply.submitApplication(email, jobId){   email = “”,  jobId = “17604”  } |
| Expected Results | False should be returned. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_7 |
| Item to Test | Job Details |
| Pre-Conditions | None |
| Test Steps | 1. Pass id in JobSearchController.Details(17604) 2. Verify action |
| Expected Results | Redirected View is “Details” |
| Priority | Medium |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | FT\_8 |
| Item to Test | Filter |
| Pre-Conditions | None |
| Test Steps | 1. Pass parameters in JobSearchController.Index(“Detroit”,”Computech”) 2. Verify action |
| Expected Results | Redirected View is “Index” |
| Priority | Medium |
| Pass/Fail |  |

## 3.7 Test Suspension / Resumption Criteria

Testing will be suspended as soon as 2 bugs are found and find solution to fix the bug. Once implemented, testing will resume from the beginning.

## 3.8 Test Environmental Needs

Functional testing can be performed in any supported environment.

# Integration Testing

## 4.1 Test Risks / Issues

There are no significant risks to the compatibility testing.

## 4.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| IT\_1 | Home Page | Home page url request goes to the home controller and home page is viewed. |
| IT\_2 | Register | Action Register in User Controller invoked and users are able to create an account. |
| IT\_3 | Email verification | Action Register in User Controller invoked and users are able to open the link sent in the verification email. |
| IT\_4 | Login | Login Action in User Controller invoked and users are able to login. |
| IT\_5 | Forgot Password | ForgotPassword Action in User Controller invoked and users are able to request link and change their password. |
| IT\_6 | Change Password | ChangePassword Action in User Controller invoked and users are able to change their password after logging in. |
| IT\_7 | Profile Information | Profile Action in User Controller invoked and Users are able to access their profile page and update their information. |
| IT\_8 | Resume Upload | UploadResume Action in User Controller invoked and Users are able to upload their resume. |

## 4.3 Test Approach

Unit tests are good to test individual software components, but Integration testing tests the entire software stack working together. So, MvcIntegrationTestFramework along with NUnit will be used for a more reliable and efficient Integration testing.

MvcIntegrationTestFramework allows to write NUnit tests that directly submit a URL, query string, cookies, request headers, etc., into the application – without needing the app to be hosted in any web server but still running in the real (non-mocked) ASP.NET runtime – and get back the response text to make assertions about. Its is also possible with this test framework to get back the ActionResult that was executed, a reference to any unhandled exception it threw, and one can also write assertions about cookie values or the contents of Session.

## 4.4 Test Pass / Fail Criteria

Pass when 100% of test cases completed

* All assertions are passed(True).
* There are no unhandled exceptions thrown.

## 4.5 Test Entry / Exit Criteria

Entry Criteria:

* Verify that the NUnit test cases are written correctly.

Exit criteria:

* All tests have been run successfully and no high priority bugs are left uncorrected.

## 4.6 Test Deliverables

|  |  |
| --- | --- |
| ID | IT\_1 |
| Item to Test | Home Page |
| Pre-Conditions | A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the root url. * Make assertion about the ActionResult. * Make assertion about the rendered cshtml view. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | IT\_2 |
| Item to Test | Register |
| Pre-Conditions | A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the registration url. * Post Registration form with valid data. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | IT\_3 |
| Item to Test | Email Verification |
| Pre-Conditions | A user has been verified after registration.  A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the verification url. * Post Registration form with valid data. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views.   Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | IT\_4 |
| Item to Test | Login |
| Pre-Conditions | A user has been registered with the credentials used in the NUnit Test.  A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the Login url. * Post Login form with valid credentials. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_5 |
| Item to Test | Forgot Password |
| Pre-Conditions | A user has been registered and verified.  A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the Forgot Password url. * Post form with valid email. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed.  System should send an email to user containing change password link and display confirmation page. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | CT\_6 |
| Item to Test | Change Password |
| Pre-Conditions | User has been registered.  User requested and received the change password link.  A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the Change Password url. * Post form with valid password. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed.  System should display confirmation message that password is changed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | IT\_7 |
| Item to Test | Profile Information |
| Pre-Conditions | User has been logged in.  A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the User Profile url. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed.  Profile information is displayed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | IT\_8 |
| Item to Test | Resume Upload |
| Pre-Conditions | User is logged in.  A NUnit test has been written that does the following:   * Simulate a browsing session. * Request the Resume upload url. * Upload resume. * Make assertion about the ActionResults executed. * Make assertion about the rendered cshtml views. * Make assertion about unhandled exceptions. |
| Test Steps | Execute the NUnit test. |
| Expected Results | All assertions are passed.  Resume is saved. |
| Priority | High |
| Pass/Fail |  |

## 4.7 Test Suspension / Resumption Criteria

Testing will be suspended as soon as a bug is found and find solution to fix the bug. Once implemented, testing will resume from the beginning.

## 4.8 Test Environmental Needs

Since MvcIntegrationTestFramework can simulate the Web Application running in the real (non-mocked) ASP.NET runtime environment, Application doesn’t need to be hosted on the computech server. A PC environment with Microsoft Visual Studio Ultimate 2013 and NUnit installed is required to run the NUnit tests.