***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 1](#_Toc414752655)

[1.1 Purpose 1](#_Toc414752656)

[1.2 Scope 1](#_Toc414752657)

[1.3 Definitions, Acronyms and Abbreviations 1](#_Toc414752658)

[1.4 References 1](#_Toc414752659)

[2. Compatibility Testing 2](#_Toc414752660)

[2.1 Test Risks / Issues 2](#_Toc414752661)

[2.2 Items To Be Tested 2](#_Toc414752662)

[2.3 Test Approach 2](#_Toc414752663)

[2.4 Test Pass / Fail Criteria 3](#_Toc414752664)

[2.5 Test Entry / Exit Criteria 3](#_Toc414752665)

[2.6 Test Deliverables 4](#_Toc414752666)

[2.7 Test Suspension / Resumption Criteria 8](#_Toc414752667)

[2.8 Test Environmental Needs 8](#_Toc414752668)

[3. System Testing 8](#_Toc414752669)

[3.1 Functional Testing 8](#_Toc414752670)

[3.1.1 Test Risks / Issues 8](#_Toc414752671)

[3.1.2 Items To Be Tested 8](#_Toc414752672)

[3.1.3 Test Approach 9](#_Toc414752673)

[3.1.4 Test Pass / Fail Criteria 9](#_Toc414752674)

[3.1.5 Test Entry / Exit Criteria 9](#_Toc414752675)

[3.1.6 Test Deliverables 11](#_Toc414752676)

[3.1.7 Test Suspension / Resumption Criteria 19](#_Toc414752677)

[3.1.8 Test Environmental Needs 19](#_Toc414752678)

[3.2 Nonfunctional Testing 20](#_Toc414752679)

[3.2.1 Test Risks / Issues 20](#_Toc414752680)

[3.2.2 Items To Be Tested 20](#_Toc414752681)

[3.2.3 Test Approach 20](#_Toc414752682)

[3.2.4 Test Pass / Fail Criteria 20](#_Toc414752683)

[3.2.5 Test Entry / Exit Criteria 20](#_Toc414752684)

[3.2.6 Test Deliverables 21](#_Toc414752685)

[3.2.7 Test Suspension / Resumption Criteria 21](#_Toc414752686)

[3.2.8 Test Environmental Needs 22](#_Toc414752687)

[4. Integration Testing 22](#_Toc414752688)

[4.1 Test Risks / Issues 22](#_Toc414752689)

[4.2 Items To Be Tested 22](#_Toc414752690)

[4.3 Test Approach 23](#_Toc414752691)

[4.4 Test Pass / Fail Criteria 23](#_Toc414752692)

[4.5 Test Entry / Exit Criteria 23](#_Toc414752693)

[4.6 Test Deliverables 24](#_Toc414752694)

[4.7 Test Suspension / Resumption Criteria 28](#_Toc414752695)

[4.8 Test Environmental Needs 28](#_Toc414752696)

[5. Performance and Load Testing 28](#_Toc414752697)

[5.1 Test Risks / Issues 28](#_Toc414752698)

[5.2 Items To Be Tested 28](#_Toc414752699)

[5.3 Test Approach 29](#_Toc414752700)

[5.4 Test Pass / Fail Criteria 29](#_Toc414752701)

[5.5 Test Entry / Exit Criteria 29](#_Toc414752702)

[5.6 Test Deliverables 29](#_Toc414752703)

[5.7 Test Suspension / Resumption Criteria 30](#_Toc414752704)

[5.8 Test Environmental Needs 30](#_Toc414752705)

[6. Security Testing 30](#_Toc414752706)

[6.1 Test Risks / Issues 30](#_Toc414752707)

[6.2 Items To Be Tested 30](#_Toc414752708)

[6.3 Test Approach 31](#_Toc414752709)

[6.4 Test Pass / Fail Criteria 31](#_Toc414752710)

[6.5 Test Entry / Exit Criteria 31](#_Toc414752711)

[6.6 Test Deliverables 31](#_Toc414752712)

[6.7 Test Suspension / Resumption Criteria 33](#_Toc414752713)

[6.8 Test Environmental Needs 33](#_Toc414752714)

[7. User Acceptance Testing 33](#_Toc414752715)

[7.1 Test Risks / Issues 33](#_Toc414752716)

[7.2 Items To Be Tested 33](#_Toc414752717)

[7.3 Test Approach 34](#_Toc414752718)

[7.4 Test Pass / Fail Criteria 34](#_Toc414752719)

[7.5 Test Entry / Exit Criteria 34](#_Toc414752720)

[7.6 Test Deliverables 35](#_Toc414752721)

[7.7 Test Suspension / Resumption Criteria 38](#_Toc414752722)

[7.8 Test Environmental Needs 39](#_Toc414752723)

# 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 Identifier

## 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 the Safari web browser on the iPhone 6 and iPad Air with iOS 8.1.3, and on the Google Chrome browser on Samsung Galaxy S5 with Android 5.0 (Lollipop). It should also be accessible on the Internet Explorer, Firefox, and Safari browsers on desktop/laptop environments, given that all browsers have up-to-date JavaScript engines.

We will test on the following hardware devices that are available to us:

|  |  |  |
| --- | --- | --- |
| Device | Operating System | Browsers |
| Laptop / Desktop | * Windows (any) * OS X | * Google Chrome * Mozilla 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 * Google Chrome |

## 

## 2.4 Test Pass / Fail Criteria

Pass when 100% of test cases completed

* Fully working in Chrome and Safari on the mobile devices listed above.
* Fully working in IE, Firefox, and Safari on desktops/laptops.

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 his email address |
| Test Steps | Repeat the below process for each supported device and browser   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.

# System Testing

## 3.1 Functional Testing

## 3.1.1 Test Risks / Issues

Unit test code needs to thoroughly test all application functionality and have at least 75% code coverage, as inadequate tests will not detect actual bugs in the code and might display unreliable output.

## 3.1.2 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. |

## 3.1.3 Test Approach

Unit test cases will be written to test each method 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. The JobSearch controller will also be tested to verify whether the jobs are displayed and filtered based on criteria.

## 3.1.4 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 testing will be restarted from the beginning.

## 3.1.5 Test Entry / Exit Criteria

Entry Criteria:

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

Exit criteria:

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

## 3.1.6 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 |  |

## 3.1.7 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.

## 3.1.8 Test Environmental Needs

Functional testing can be performed in any supported environment.

## 3.2 Nonfunctional Testing

## 3.2.1 Test Risks / Issues

Thorough testing of nonfunctional aspects such as reliability and availability is necessary so that unanticipated problems such as long system maintenances do not make the application unusable.

## 3.2.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| NT\_1 | Application Reliability | Job search and recommendation algorithms return correct results within reasonable time |
| NT\_2 | Application Availability | Web application is loaded within a reasonable time in user’s web browser whenever it is requested |

## 3.2.3 Test Approach

Given that the functionality of the web application has proved to be working correctly, nonfunctional testing will verify that the overall functionality of the site is available and reliable. This is done by two simple user-based tests in which the reliability of the search results and site availability are verified. The reliability test involves examination of search results for logical soundness and the availability test involves timed access of the site from a user device.

## 3.2.4 Test Pass / Fail Criteria

Both of the tests pass if web pages are loaded in the two second target time, i.e. if the application is available from the webserver. The reliability test may fail if the correct data is not retrieved from the database, which would signal an incorrect query. Also, the availability test may fail if the tester receives an error notification in the web browser. In either case, the bug would have to be fixed and the test would have to be rerun.

## 3.2.5 Test Entry / Exit Criteria

To run these nonfunctional tests, the web application must first be hosted on the Computech server and available via a URL. The exit criteria for both nonfunctional tests are that they have all passed without errors.

## 3.2.6 Test Deliverables

|  |  |
| --- | --- |
| ID | NT\_1 |
| Item to Test | Application Reliability |
| Pre-Conditions | The web application is hosted on the Computech server |
| Test Steps | Execute the following ten times:   1. In the job search window select criteria from each filter 2. Execute the search and start stopwatch 3. Compare results to items selected in search window |
| Expected Results | Search results match criteria selected by the user logically AND together; all search results are loaded to the web page within two seconds |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | NT\_2 |
| Item to Test | Application Availability |
| Pre-Conditions | The web application is hosted on the Computech server |
| Test Steps | Execute the following ten times:   1. Type the web application URL into the web browser and press enter 2. Begin stopwatch |
| Expected Results | Web application loads within in an average of two seconds and doesn’t display a connectivity error message |
| Priority | High |
| Pass/Fail |  |

## 3.2.7 Test Suspension / Resumption Criteria

Testing will be suspended as soon as any bug is found that would make normal use of the application difficult. After a solution to the bug is implemented, testing will resume from the beginning.

## 3.2.8 Test Environmental Needs

Testing must be carried out in each of devices and browsers mentioned in the compatibility requirements section above to ensure that environment differences do not affect usability of the web application.

# 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.

# 5. Performance and Load Testing

## 5.1 Test Risks / Issues

We will be testing performance using both a Visual Studio-generated load stress test and also a self-conducted loading time test, with the risk that these tests will not adequately reflect the performance of the application in the event of simultaneous use of the site more than fifty users.

## 5.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| PT\_1 | 50 simultaneous users | We will simulate 50 simultaneous login requests to the server looking for an average response time of 2 seconds or less |
| PT\_2 | Individual page load time | We will use a stopwatch to time the page load and transition times when accessing each page of the web application looking for a response time of 2 seconds or less on average |

## 5.3 Test Approach

We will be conducting a very standard automated load test using the Microsoft Visual Studio tool designed for use with ASP.NET web applications. On average, it is anticipated that only a few users will be accessing the EJCA at the same time, however to be safe we are testing using fifty simulated logins, as this involves not only requests to the webserver but also to the database. Additionally, the self-timed tests will ensure adequate performance in a typical use case.

## 5.4 Test Pass / Fail Criteria

The application will pass performance and load testing when fifty users can simultaneously log into the application with a two second average load time and when each page loads in two seconds on average.

## 5.5 Test Entry / Exit Criteria

The criteria necessary for entry of performance unit tests are:

* All previous functional and nonfunctional unit test cases have been run and passed
* A stress test plan has been configured in the Visual Studio Ultimate testing suite

The exit criteria for performance and load testing are passing of the Visual Studio and self-timed tests.

## 5.6 Test Deliverables

|  |  |
| --- | --- |
| ID | PT\_1 |
| Item to Test | User Login |
| Pre-Conditions | A Visual Studio stress test has been written that does the following:   * Simulate 50 browsing sessions * Request the root URL * Request the user login screen * Enter 50 valid login usernames and passwords |
| Test Steps | Execute the Visual Studio stress test. |
| Expected Results | Two second average load time for all simulated users |
| Priority | High |
| Pass/Fail | Result log from the Visual Studio stress test |

## 5.7 Test Suspension / Resumption Criteria

Given that we have only one single performance or load test, our goal is that our application will pass the test. However, if we encounter a failure, we will suspend testing, investigate and resolve the cause of failure, and retest until we attain a successful outcome.

## 5.8 Test Environmental Needs

Performance testing requires access to a machine running Visual Studio Ultimate in order to make use of the built-in stress testing software tool for ASP.NET web applications. Use of the Visual Studio tool also requires the EJCA project source code. Self-timed tests may be conducted when the site is hosted on the Computech server and require only a simple stopwatch.

# 6. Security Testing

## 6.1 Test Risks / Issues

The only risk is that the security tests do not cover all security loopholes in the web application. Thus, security testing must be performed thoroughly on each security feature of the web application so that users feel safe accessing the portal and submitting their personal data. Security breaches could result in loss or user data, eroding user confidence in the application.

## 6.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| ST\_1 (\_) | Secure User Login | Only users who have registered and created an account may access the portal and apply for jobs |
| ST\_2 (\_) | Admin Mode | Only a single administrator may access the admin portal and manipulate user account information |
| ST\_3 (\_) | Password Encryption | User passwords are stored encrypted in the database and are unreadable to anyone accessing the database |
| ST\_4 (\_) | SSL Encryption | Data transmitted on the internet between the user’s device and the webserver is encrypted and uses HTTPS protocol |

## 6.3 Test Approach

Security tests will be conducted using the web application’s user interface, by accessing Geocerts.com SSL certificate checking site, and by reviewing records in the Users table of the application database.

## 6.4 Test Pass / Fail Criteria

The web application passes security testing when each of the test cases passes. If any of the tests fail, the issue will be debugged and resolved before the application is retested.

## 6.5 Test Entry / Exit Criteria

Security testing will begin after the functional and nonfunctional unit testing of the application is finished and will be complete when all possible security vulnerabilities have been detected and resolved.

## 6.6 Test Deliverables

|  |  |
| --- | --- |
| ID | ST\_1.1 |
| Item to Test | Secure User Login |
| Pre-Conditions | All other functional and nonfunctional application testing is complete |
| Test Steps | Click on Login. Enter random email and password credentials. |
| Expected Results | A message must be shown that invalid user login credentials were entered. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | ST\_1.2 |
| Item to Test | Admin Mode |
| Pre-Conditions | All other functional and nonfunctional application testing is complete |
| Test Steps | Click on Login. Enter valid admin credentials and click on submit.  Select option to deactivate a user. |
| Expected Results | Deactivated user can no longer access the secure portal |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | ST\_1.3 |
| Item to Test | Password Encryption |
| Pre-Conditions | All other functional and nonfunctional application testing is complete |
| Test Steps | Click on application database. View data in Tbl\_Users.  Verify that no user password is a human readable string. |
| Expected Results | Each user password is encrypted in a string that is unreadible |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | ST\_1.4 |
| Item to Test | SSL Encryption |
| Pre-Conditions | All other functional and nonfunctional application testing is complete |
| Test Steps | Navigate to the Geocerts SSL certificate validation site. Fill out the form with server credentials and click submit. |
| Expected Results | A message must be shown displaying the correct server information and verifying that the server has an up-to-date SSL certificate. |
| Priority | High |
| Pass/Fail |  |

## 6.7 Test Suspension / Resumption Criteria

Given the great importance of security to the successful functioning of the web application, and the interconnectedness of the security architecture, security testing will be suspended when a single vulnerability is detected. After the vulnerability is resolved, testing must be restarted to ensure that all security features continue their proper functionality.

## 6.8 Test Environmental Needs

To complete security testing, the test administrator needs access to the application database to verify user password encryption, as well as access to the Geocerts website to verify SSL certificates.

# 7. User Acceptance Testing

## 7.1 Test Risks / Issues

User Acceptance testing (UAT) is the last step in testing and will be done by the members of Computech quality assurance team.

## 7.2 Items To Be Tested

|  |  |  |
| --- | --- | --- |
| **ID** | **Item to Test** | **Test Description** |
| UAT\_1 | Register | A new user is able to register. |
| UAT\_2 | Email verification | User is able to verify email after registration. |
| UAT\_3 | Login | A new user is able to login after registration. |
| UAT\_4 | Forgot Password | User is able to get link to reset password in email. |
| UAT\_5 | Change Password | A new user is able to change password from the link sent to his email. |
| UAT\_6 | Apply | Once logged in, a user is able to apply for jobs. |
| UAT\_7 | Job Details | A new user is able to view job details. |
| UAT\_8 | Filter Jobs | A new user is able to filter jobs using all the available filters. |

## 7.3 Test Approach

User Acceptance Testing is the last step after all other tests have been passed. UAT will be done by the members of Computech quality assurance (QA) team after the Web Application has been hosted on their server. After Computech QA signs off on UAT, there should be no more bugs that need to be fixed.

## 7.4 Test Pass / Fail Criteria

Pass when 100% of test cases completed successfully and Computech QA team signs off on the Application.

## 7.5 Test Entry / Exit Criteria

Entry Criteria:

Exit criteria:

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

## 7.6 Test Deliverables

|  |  |
| --- | --- |
| ID | UAT\_1.1 |
| Item to Test | Register |
| Pre-Conditions | Web Application is hosted on Computech server. |
| Test Steps | Click on Register. Fill out the form and click submit. |
| Expected Results | A message must be shown that verification link has been sent to email. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_1.2 |
| Item to Test | RegisterFails |
| Pre-Conditions | Account already created with an email. |
| Test Steps | UAT\_1.1 repeated with the already registered email. |
| Expected Results | Message must be shown indicating Registration failed because an account has already been created with the entered email. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_2.1 |
| Item to Test | Email verification |
| Pre-Conditions | Execute UAT\_1.1 |
| Test Steps | Click on the verification link received in email. |
| 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 | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_3.1 |
| Item to Test | Login |
| Pre-Conditions | User has been registered and Email has been verified |
| Test Steps | Click Login. Enter credentials used to register and verify. |
| Expected Results | Login should be successful and user redirected to User Dashboard. |
| Priority | True should be returned. |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_3.2 |
| Item to Test | LoginFails |
| Pre-Conditions | Username and Password have not been used to register. |
| Test Steps | Login with the Username and Password. |
| Expected Results | Error message displayed indicating Login failed due to invalid credentials. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_4.1 |
| Item to Test | Forgot Password |
| Pre-Conditions | User account has been created |
| Test Steps | Click on forgot password. |
| 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. UAT\_4.1 executed. |
| Test Steps | Click on change password link sent to email. |
| Expected Results | User password is successfully changed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_6.1 |
| Item to Test | Apply |
| Pre-Conditions | User has logged in. Click on a job to go to job details. |
| Test Steps | Click on Apply. |
| Expected Results | Takes user to application page. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_7 |
| Item to Test | Job Details |
| Pre-Conditions | None |
| Test Steps | Click on a job on the dashboard. |
| Expected Results | Job details are displayed. |
| Priority | High |
| Pass/Fail |  |

|  |  |
| --- | --- |
| ID | UAT\_8 |
| Item to Test | Filter |
| Pre-Conditions | None |
| Test Steps | Filter by all the filter (One at a time and multiple multiple filters together). |
| Expected Results | Correct filtered jobs are displayed. |
| Priority | High |
| Pass/Fail |  |

## 7.7 Test Suspension / Resumption Criteria

Testing will be not suspended when a bug is found. Bugs will be reported to the development team for fixes. Once fixes are implemented, an updated version of the application will be provided to the QA team to continue UAT with the updated version. UAT will not be suspended until all issues are resolved and Computech QA team signs off on the application.

## 7.8 Test Environmental Needs

Application will be hosted on the Computech server and be accessible to anyone via internet on a supported device and web browser.