Software Requirements Specification

for

Enhanced Job and Candidate Application

**Prepared by:**

Simul Kadakia

Wesley Trescott

Gagandeep Singh

**Table of Contents**

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 References 1

1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 3

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 3

3.2.1 <Functional Requirement or Feature #1> 3

3.2.2 <Functional Requirement or Feature #2> 3

3.3 Non-Functional Requirements 3

3.3.1 Performance 4

3.3.2 Reliability 4

3.3.3 Availability 4

3.3.4 Security 4

3.3.5 Maintainability 4

3.3.6 Portability 4

3.4 Design Constraints 4

3.5 Logical Database Requirements 4

3.6 Other Requirements 4

4. Analysis Models 4

4.1 Data Flow Diagrams (DFD) 4

A. Appendices 4

A.1 Appendix 1 4

A.2 Appendix 2 4

# 1. Introduction

This section gives a scope description and overview of everything included in this SRS 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 Requirements Specification (SRS) document is to provide a detailed description of the functionalities of the ‘Enhanced Job and Candidate Application’ system. This document will cover each of the system’s intended features, as well as offer a preliminary glimpse of the software application’s User Interface (UI). The document will also cover hardware, software, and various other technical dependencies. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

## 1.2 Scope

‘Enhanced Job and Candidate Application’ is a web application where Wayne State University students can search for jobs based on their skill sets and apply to those they are interested in. The web application is not specific to a particular device and needs to be compatible with smartphones, tablets, and desktop environments.

Users can log into a secure portal where they can access their saved data such as email address, contact information, and experience. They can filter job searches based on specific criteria, such as location and industry, which will narrow the search results of the available jobs. Students can apply to the jobs that they are interested in, with the option of uploading a resume. To apply, users can either manually enter their data or populate the fields from saved data previously entered in their user profile. Based on users’ saved information, the application with provide the user with an automatic job matching functionality that displays to the user a list of jobs potentially fitting their credentials.

Additionally, the application will feature an admin mode that will interface with the Pinpoint system to delete or ban users, lock accounts, and search for users based on name, email, or location.

## 1.3 Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
| EJCA | Enhanced Job and Candidate Application |
| User | Students who uses the application |
| Admin/Administrator | System administrator who is given specific permission for managing and controlling the system |
| Admin portal | Part of web application which provides special facilities to Admin/Administrator |
| Stakeholder | Any person who interacts with the application except developers |
| API (Application Programming Interface) | A generalized interface used to allow EJCA  to interact with external software/application (Pinpoint). |
| Front end | The aspect of application that user interacts with |
| Back end | The aspect of application that is developed and maintained by developers. |
| UI (User Interface) | Front end of EJCA |
| DESC | Description |
| DEP | Dependency |
| MUST | The minimum level required to avoid failure |
| PLAN | The level at which good success can be claimed |
| WISH | A desirable level of achievement that may not be attainable through available means |
| DEFINED | The official definition of a term |

## 1.4 References

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

# 2. General Description

## 2.1 Product Perspective

The project is built from scratch, and thus there is no initial foundation to build our system upon. The system will consist of a backend and a frontend. The backend will handle the storage of jobs listing and user information data, as well as the querying and delivery of this data back to the frontend to be displayed. The frontend will allow a user to interact directly with a webpage which will allow a user to specify a particular filter and then send a query to the backend to retrieve data.

## 2.2 Product Functions

* User can login
* User can browse open job listings
* User can search/filter based on certain criteria
* User can apply to jobs
* User can check what jobs they have applied to
* Admin can login
* Admin can search for users by their first and last name
* Admin can lock/ban/delete users

## 2.3 User Characteristics

We expect two different types of users for this project: Students and Admin.

Students are considered the most important users because they are the ones who search for jobs and apply to the ones that they are interested in. They are also going to be the majority of site views because the web app is mainly focused on delivering content to them. Most potential users will simply browse job listings until they find job listing(s) they are interested in at which point they can apply.

Admin are the second most important users of our site. They can search for users based on their first and last name. They can also lock, ban and remove users who abuse the application and apply to multiple jobs in short period of time without any relevant experience/skills.

## 2.4 General Constraints

This is a web application and the details about the operating environment are not important. End users will use the application from a modern internet browser such as Firefox, Chrome and Internet Explorer.

## 2.5 Assumptions and Dependencies

Assumptions to properly use the application include:

* Internet connection with enough bandwidth to fully render various displays
* A modern browser with up to date technologies, such as a modern JavaScript engine

# 3. Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

User should be able to see the job search page directly when the application is opened. Here the user chooses the type of search he/she wants to conduct. There will also be a Login link if the user wants to login to their account and Register link if the user wants to create a new account.

Every page will include a standard header with important links. Some preliminary user authentication dialogs are shown below:

|  |  |
| --- | --- |
|  |  |
|  |  |

### 3.1.2 Hardware Interfaces

EJCA does not use any hardware interfaces.

### 3.1.3 Software Interfaces

EJCA will run on MVC architecture. The messages passed would be GET and POST requests from the user or admin, as well as calls to insert, delete, or select data from our database.

### 3.1.4 Communications Interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the server where the application is deployed on.

## 3.2 Functional Requirements

### 3.2.1: User class - Students

### 3.2.1.1 Search:

### DESC: User can search for various jobs available.

### REQ: None.

### 3.2.1.2 Register:

### DESC: User can create a new account by providing personal information such as email id.

### REQ: None.

### 3.2.1.3 Login:

### DESC: Given that a user has registered, then the user can login by entering email id and password entered during registration process.

### DESC: FR 3.2.1.2

### 3.2.1.4 Retrieve password:

### DESC: Given that a user has registered, then the user should be able to retrieve password through email.

### REQ: FR 3.2.1.2

### 3.2.1.5 Apply:

### DESC: Given that a user is logged in, the first page they can see is the home page. User can search/filter jobs and apply for jobs.

### REQ: FR 3.2.1.3

### 3.2.1.6 Filter by skills:

### DESC: User can filter jobs based on skills required for the job.

### REQ: None

### 3.2.1.7 Filter by location:

### DESC:User can filter jobs based on location.

### REQ: None

### 3.2.1.8 Filter by experience:

### DESC: User can filter jobs based on experience required.

**REQ**: None

### 3.2.1.9 Profile Information:

### DESC: Given that a user is logged in, then the user can enter their information such as skills and experience in profile page.

**REQ**: FR 3.2.1.2

### 3.2.1.10 Change Information:

### DESC: Given that a user is logged in, then the user can change information stored in their profile page.

**REQ**: FR 3.2.1.2

### 3.2.1.11 Upload Resume:

### DESC: Given that a user is logged in, then the user can upload a resume and attach it to their profile.

### REQ: FR 3.2.1.2

### 3.2.2: User class - Admin

### 3.2.2.1 Login:

### DESC: Admin can login by entering email id and password stored in database.

### REQ: FR 3.2.2.2

### 3.2.2.2 Search:

### DESC: Admin can search for all registered users by their first and/or last name.

**REQ**: FR 3.2.2.2

### 3.2.2.3 Deactivate users:

### DESC:Admin can deactivate users which will disable them to login into their account.

**REQ:**FR 3.2.2.2

### 3.2.2.4 Delete users:

### DESC: Admin can delete users which will delete all of their information from the database.

**REQ**: FR 3.2.2.2

## 3.3 Non-Functional Requirements

The following requirements specify, for the system level, the mandatory performance, reliability, availability, security, maintainability, and portability of the software product.

### 3.3.1 Performance

This requirement specifies the mandatory performance metrics of the system during user interaction with the software. This includes response time to user logins, registrations, job searches, and admin logins and activities. 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 relatively small amounts of information, with either basic user information or a number of job listings being retrieved from or entered into the database, response times should be quick. In particular, we aim for overall loading quickness to be at a maximum of five seconds 100% of the time, with a target of less than two seconds 98% of the time, which will be verified during five hours of testing in our phase 3 system tests, as per our development plan.

### 3.3.2 Reliability

This requirement specifies the extent to which the software application reliably produces the correct job search results based on valid search criteria, as well as suggests jobs to users based on their skills and experience. Also, this metric encompasses report messages to application users regarding lost internet connections or incorrect registration or search inputs. Each of these requirements will be thoroughly tested and evaluated during the phase 2 testing mentioned in the development plan. In particular, we plan to create at least 25 user accounts with both valid and invalid credentials (invalid credentials being, for example, a duplicate email address or password). These user accounts will contain various experience data, which will be used to assess job recommendation functionality. We will also perform at least 50 job searches using these accounts on a limited jobs database to assess correctness of search algorithms. In these tests, we will require correct error reporting 100% of the time, correct job search results 98% of the time, and correct job recommendations 98% of the time.

### 3.3.3 Availability

This requirement specifies the readiness of the system for use whenever it is run. This entails the accessibility of the application from an internet browser not considering network failures. Thus, this requirement assumes that the user has an internet connection so that the application can connect to the database. The availability requirement will be tested in the phase 3 system tests mentioned in the development plan. In testing, we plan for five hours of active use of the application, during which period our target system availability is 100% of the time, with a 98% of the time availability considered passing.

### 3.3.4 Security

This requirement specifies that the admin mode and 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. This requirement also encompasses the validation of user logins, so that incorrect user credentials will not result in access to the application. Furthermore, the functionality of the admin mode ensures that users abusing the application will be removed. Each of these security requirements will be expected to function correctly 100% of the time in our tests.

### 3.3.5 Maintainability

This requirement specifies the degree of extensibility and testability of the EJCA. Firstly, extensibility implies both the easy addition of new features to the application and also the ease of portability to new, previously unsupported devices. This requirement is satisfied by our use of popular technology (ASP.NET), which is a framework widely used in web application development across multiple platforms, therefore attesting to its adaptability to new environments. Also, our program will be written using the MVC design pattern, which allows for flexible addition of new features through new controllers and models. Furthermore, our application supports easy testability by taking advantage of the test suites generated by ASP.NET for all of the functionality of the EJCA.

### 3.3.6 Portability

This requirement specifies that the EJCA be portable to multiple devices. Given the up-front project requirement of compatibility with both mobile and desktop environments, the application will be designed and coded so that all its functionality is compatible with web browsers in PC, Mac, Android, iPhone, and iPad. The effect of this multi-platform design intent is that the application will be easily portable to each supported environment. Complete system testing of all functionality of the application will be carried out on each supported device during phase 3 testing.

## 3.4 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

* RAM (memory) required by the application
* Hard drive space required by the application
* Social media login?

Check about different devices affect webpage layout. Small screen, big screen, mobiles, etc.

Found a good article about constraints: <http://www.webforefront.com/performance/limitedresources.html>

## 3.5 Logical Database Requirements

A SQL Server database will be used in the backend to store information entered by the user, display jobs to users and display registered users to admin. The front-end will be on the user/admin machine.

## 3.6 Other Requirements

**3.6.1 Mobile Platform Compatibilty**

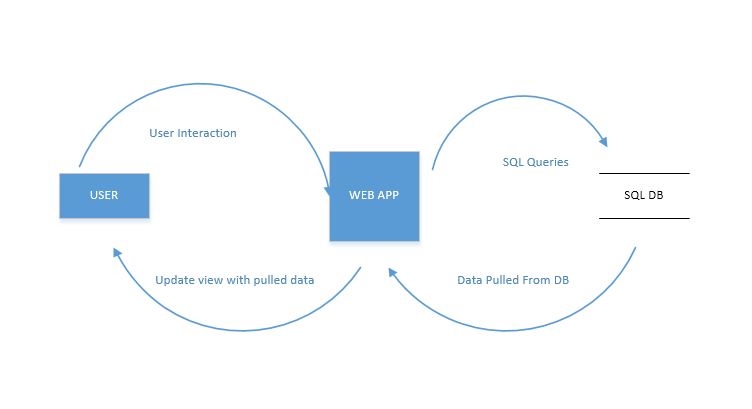
Web Application is compatible with mobile phones and tablets along with PCs, primarily iOS and Android based devices. User view is configured to the device’s screen resolution and can be accessed from any web browser installed on the device. User has the ability to use every functionality in the app on these mobile devices. However, user is unable to upload resumes from iOS based devices.

# 4. Analysis Models

## 4.1 Data Flow Diagrams (DFD)

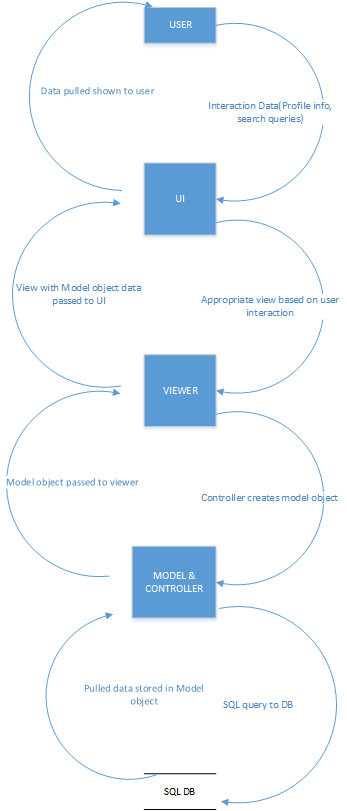
**4.1.1 Level1 DFD**

Level1 DFD shows the flow of data among the user, web application and the SQL database.



**4.1.2 Level2 DFD**

Level2 DFD 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.



# A. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## A.1 Appendix 1

## A.2 Appendix 2