**Software Requirements Specification**

**for**

College Jobs

**Version 1.0 approved**

**Prepared by Team D**

**COS 420**

**10/05/2023**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Initial Version | 10/05/23 | Initial | 0.0.1 |
| Updated Doc | 10/11/23 | Finishing Initial | 0.0.2 |
| Updated Doc | 10/22/23 | instructor feedback from deliverable 0,1 | 0.0.3 |
|  |  |  |  |

# **Introduction**

## **Purpose**

College Jobs is a job listing aggregator designed to help college students find part time and freelance jobs with good pay that will further the skills they are learning in college. Our users will be able to search for part time jobs *and* freelance work on the same site, and the platform will automatically filter out jobs that are not compatible with the requirements of students in college.

## **Document Conventions**

This Document was created based on the IEEE template for System Requirement

Specification Documents.

## **Intended Audience and Reading Suggestions**

This document is for developers, project managers, users and testers.

## **Product Scope**

For the architecture design of CollegeJobs, we use a combination of Model-View-Controller architecture and Client-Server architecture. MVC is ideal for building interactive web applications. We use React for our front-end, which is built on the MVC architecture, so that is how our user interface is architected. Since we scrape jobs from other sites, a time and compute intensive process, we decided to introduce a central server to scrape and cache job listings. This also simplifies and speeds up the client side web interface, because we can send everything through our server API, which offloads resources requirements from client to server.

## **References**

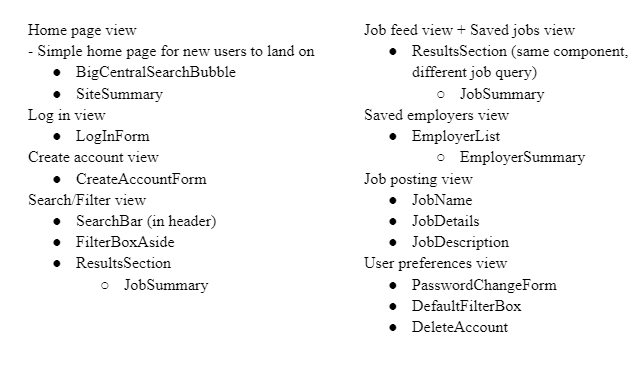
*Team D GitHub link-* [*https://github.com/COS420-Fall23/Team-D.git*](https://github.com/COS420-Fall23/Team-D.git)

# **Overall Description**

## **Product Perspective**

*College Jobs is an application meant to replace the tedious job search sites and find a better fit for college students. This will allow students to find jobs with just skill based experience and have it be part time. College Jobs will scrape exciting job sites for more of the part time/freelance jobs college students are looking for.*

## **Product Functions**

**

## **User Classes and Characteristics**

*College Jobs is created using the React App, with HTML, CSS, and Typescript. We are assuming users are not blocking HTML. If so then the application will not work on the user's device.*

## **Operating Environment**

*The hardware that will be supported are windows 10 and 11, Mac OS, android modern day, and Linux kali, mint, Ubuntu, etc.*

## **Design and Implementation Constraints**

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>*

## **User Documentation**

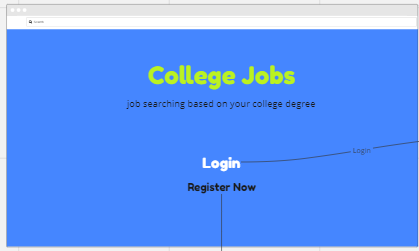
*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

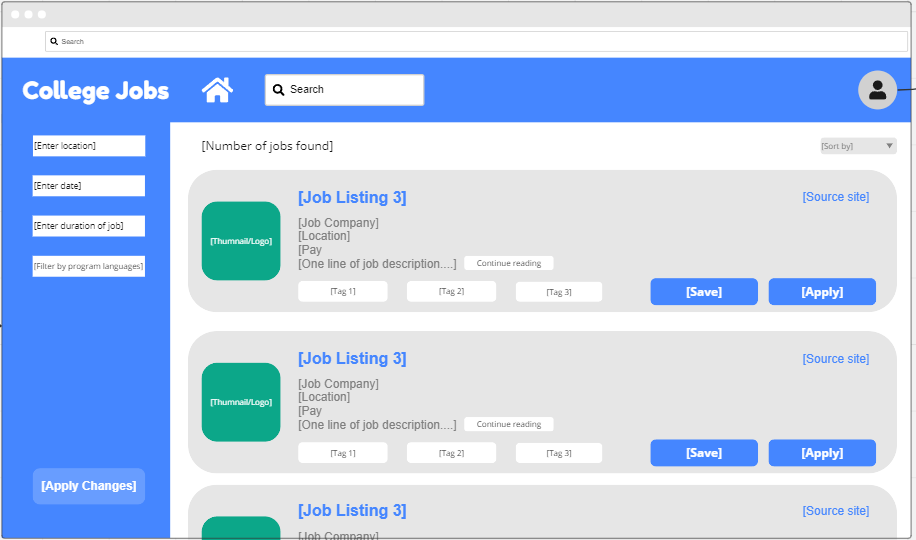
## **Assumptions and Dependencies**

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

# **External Interface Requirements**

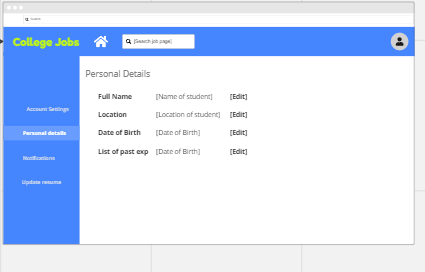
## **User Interfaces**





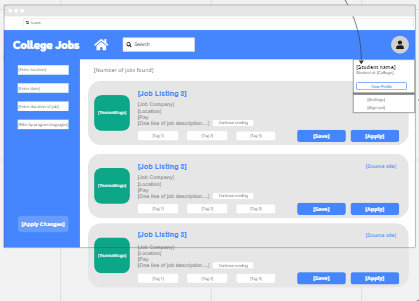
This is the basic view of the home page to log into the application. Along with a basic view of the search page. In order to be able to apply to jobs or change information you will need to be logged in.

## **Hardware Interfaces**

**

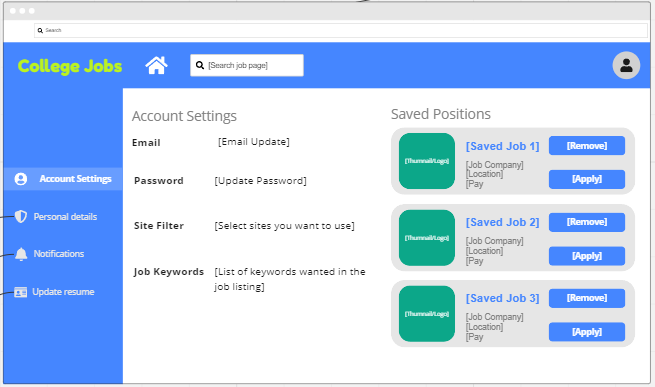
*This is a basic page for changing or adding personal information.*

## **Software Interfaces**

**

*The basic look of the search page and filter button.*

## **Communications Interfaces**



*The basic page for account settings and saving jobs. Allows users to save a place to apply later. Also will be the spot to choose who can contact you.*

# **System Features**

## **System Feature 1**

REQ-1:The system shall allow a user to view job listings

REQ-2: *The system must allow an user to search for jobs*

REQ-3: The system shall show a single job location for a job posting

REQ-4: The system must show job listings in computer science field

REQ-5: The system must show a users skill set in their profile

REQ-6: *The system must provide a privacy setting, such as who can view the users profile*

REQ-7: *The system will not allow unverified users from applying to jobs*

REQ-8: *The system must verify the user is human by use of a captcha check*

REQ-9: *The system must give the option for an authenticator app, like google authenticator*

REQ-10: *The system will scrape for job listings*

REQ-11: *The system will have an option to filter work opportunities by a filter button*

REQ-12: *The system will have an option to sort work opportunities by date posted, expected pay, and distance.*

REQ-13: *The system will provide links to full listings and more information for a specified work opportunity when a user requests it, under more information tab on listings*

REQ-14: The system will show a user the risks associated with that job, e.x high voltage, gas, etc

REQ-15: The system will scrape for job offerings for the user by using api from other job sites to pull it to this sites listings

REQ-16: The system shall allow students to filter their search based on a 5 star rating scale for companies.

REQ-17:The system shall allow users who are students to follow companies they are interested in.

REQ-18: The system shall allow companies to create an “easy apply” form for students to fill out.

## **System Feature 2 (and so on)**

**Application process**

4.2.1 Description and Priority

The application process for this service should be simple and straightforward. The application feature on the service will be in the form of a button on job listings that allows a user to

4.2.2 Stimulus/Response Sequences

When a user logs into the service, the database should check the username and password given to it to see if it matches the correct username and password of the user. In terms of response, he system shall authenticate username and password and display:

1. If the user logs in with the correct username and password, the service should give them access to the job listings/whatever the home screen should be.
2. If the user logs in with an incorrect password or username, the service should make note that “Password and/or username is invalid. Please try again or try to reset your password.”

# **Other Nonfunctional Requirements**

## **Performance Requirements**

NON-REQ 1- The system shall allow users to use the system 5 seconds following registration.

NON-REQ 2- The system shall have an at least 99.9% uptime.

NON-REQ 3- The system shall have a search functionality that returns results within 5 seconds 95.9% of the time.

NON-REQ 4- The system shall be available for 24hrs a day, 7 days a week to all users, 99% of the time.

NON-REQ 5- The system should limit the photo file size to 500MB.

## **Safety Requirements**

NON-REQ 6- The system shall be General Data Protection Regulation (GDPR) compliant.

## **Security Requirements**

NON-REQ 7- The system shall allow users to register so long as they have an email that is not previously registered.

NON-REQ 8- The system shall only allow a user to register if they have verified their email during the registration process. with their email address

NON-REQ 9- The system shall store user login information securely by using encryption.

## **Software Quality Attributes**

NON-REQ 10- The system shall have a login and registration process with 5 login attempts

NON-REQ 11- The system shall be easily usable on mobile android, windows, linux, and apple devices in modern day

NON-REQ 12- The system shall be capable of handling a typical workload of 5,000 users at the same time 95% of the time.

## **Business Rules**

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

# **Other Requirements**

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*