****

**Resume Organizer**

*MITR Team 2*

*Sean Hale, Andrew L’Italien, Alden Ford, Jonathan Wong, Kolby Fisher, Varun Ravichandran*

*Professor Matthew Grill*

*April 29th, 2021*

# **Section 1: Executive Summary**

ECS is a technology company headquartered in Fairfax, Virginia where they provide services and solutions in science, engineering, and advanced technologies. These advanced technologies include cloud development, cybersecurity, artificial intelligence (AI), machine learning (ML), and IT modernization. ECS maintains partnerships with leading cloud, cybersecurity, and AI/ML providers and holds specialized certifications in their technologies. Partners include AWS, Microsoft, McAfee, Elastic, Google Cloud, IBM Cloud, and RSA.

ECS’s need to implement a robust resume management tracking system was important to its long-term ability to gather essential information from employees’ resumes. Stephen Ahern, the project manager on ECS’ side, wanted to build a resume management tracking system for quite some time, but other business needs had taken priority. One group member, Sean Hale, had a connection in ECS and was able to reach out to inquire about creating a potential IT solution for ECS for a class project in Managing Information Technology Resources (MITR) at Rensselaer Polytechnic Institute (RPI). The team consists of Sean Hale, Jonathan Wong, Alden Ford, Andrew L’Italien, Kolby Fisher, and Varun Ravichandran.

The team chose to use the MERN Stack, which is composed of MongoDB, Express, React, and Node.js, to create the resume management tracking system. The MERN Stack satisfied the project requirements effectively, as it allowed the team to make a dynamic front-end as well as make an efficient and effective back-end. The team also utilized AWS cloud hosting, as it was a convenient, scalable option that would make the transition of resources easier at the end of the project. Lastly, ECS was familiar with these technologies so it would make the project turnover that much easier.

In terms of the project management methodologies used within the project, the team employed a hybrid of the Agile and Waterfall methodologies. The team met on a consistent basis with each other and ECS to discuss work-related ideas, requirements, problems, or updates. Some of the related software that the team used includes Discord for communication and voice chat, Microsoft Teams for the ECS calls, Google Drive for storing documents, GitHub for the Kanban board, and Gantter to create a waterfall chart.

A cost benefit analysis (CBA) was created after talking with ECS about the feasible present and future financials for the project scope. Given the numbers that were provided by the client, the CBA showed that the NPV at a 12% discount rate would be a slightly more than two and a half million US dollars. This shows the massive positive impact that the resume management tracking system would have on ECS.The team also evaluated a number of risks and their respective probabilities and impacts and determined that the relative risk for the project was quite low. The main three risks are: a team member leaving RPI, the companies project manager or employees tied to the project potentially leaving ECS, or the team being unable to complete the project due to technical difficulties. The team mitigated risks depending on their severity, as described in section 10.

After the CBA was completed, the team met with ECS to discuss IS/IT requirements that would be implemented into the project if both parties agreed to them. The initial meetings were spent going over the IS/IT requirements that were brainstormed and later proposed to ECS. After some discussion, the IS/IT requirements were settled based on both parties' agreements. The functional requirements included the process, people, and database while the non-functional requirements included hardware, software, network, security, scalability, and look/feel. In terms of the functional requirements, the team implemented a modern-looking website interface that matched ECS’ colors and themes. A searching mechanism was created to filter a resume database for the candidates who possessed skills that managers or human resources could utilize to their advantage. On top of that, the team also made it a priority to be able to scale the application when needed. If there was ever an influx of users on the application, the infrastructure could scale up or down depending on the incoming traffic using the website. All of the requirements, both functional and non-functional, were met by the completion of the application.

On top of creating the application, ECS stressed the importance of having well written documentation and the team made it a combined effort to provide it. This was mutually beneficial, as it allowed the ECS team to pick up where RPI’s team left off if extra functionality was desired and reduced the need for the RPI students to spend time in meetings explaining functionality. The team built the application to make it easy to add any additional features without much hassle by implementing features via a standard API. The client was consistently satisfied with the work completed after showcasing the application through demonstrations and weekly meetings.

While the project was challenging at times, the team showed resiliency and pushed through, completing deliverables for both ECS and RPI’s MITR class. The team gained a lot of valuable knowledge from this experience including accountability, communication, and new technology, all of which would become valuable assets later down the road.

Table of Contents

[**Section 1: Executive Summary** 1](#_Toc70608628)

[**Section 2: Introduction** 4](#_Toc70608629)

[**Section 3: Client Organization** 4](#_Toc70608630)

[**Section 4: Project Team** 4](#_Toc70608631)

[**Section 5: Problem Statement** 5](#_Toc70608632)

[**Section 6: IS/IT Solution** 5](#_Toc70608633)

[**Section 7: Methodology** 6](#_Toc70608634)

[**Section 8: IS/IT Requirements** 7](#_Toc70608635)

[Functional Requirements 7](#_Toc70608636)

[Non-Functional Requirements 8](#_Toc70608637)

[**Section 9: IS/IT Design** 9](#_Toc70608638)

[Use Cases 9](#_Toc70608639)

[Back-End Design 9](#_Toc70608640)

[Workflow Diagram 10](#_Toc70608641)

[Data-Flow Diagram 11](#_Toc70608642)

[Database Design 11](#_Toc70608643)

[Front-End Design 13](#_Toc70608644)

[System Prototype 15](#_Toc70608645)

[Code Base 15](#_Toc70608646)

[Test Plans 15](#_Toc70608647)

[Test Results 15](#_Toc70608648)

[**Section 10: Cost Benefit Analysis with Risk Management** 16](#_Toc70608649)

[**Section 11: Project Plan/Schedule** 19](#_Toc70608650)

[**Section 12: Post-Turnover Plan** 21](#_Toc70608651)

[**Section 13: Results and Client Feedback** 22](#_Toc70608652)

[**Section 14: Conclusions** 22](#_Toc70608653)

# **Section 2: Introduction**

ECS had a need for a robust resume management tracking system to improve the long-term ability to index and search essential information about employees. ECS and RPI students came to an agreement wherein the students would create a resume management tracking system that would be used almost every day. Given that a resume management tracking system is essential to keeping track of current employees, it became clear that the team of RPI students would consequently have a large impact on ECS.

# **Section 3: Client Organization**

The team’s client contact was Stephen (Steve) Ahern, a Director of Operations for the Justice Business Unit of ECS. Steve oversees approximately 100 team members across seven projects and maintains a managerial and support position for each. Steve can be contacted at [steve.ahern@ecstech.com](mailto:steve.ahern@ecstech.com). More information regarding ECS can be found on their website, <https://ecstech.com>.

# **Section 4: Project Team**

Sean Hale is a dual major with Computer Science and Information Technology & Web Science concentrating in machine learning. Sean has ample experience with full-stack development thanks to the ITWS curriculum as well as multiple intern and freelance development projects. Sean was the team’s project manager and also served as a back-end developer.

Jonathan Wong is an Information Technology & Web Science undergraduate student whose concentration is in Web Technologies. He has technical proficiency with many areas in technology, some of those being software development or cloud computing, which were demonstrated in past experiences through RPI undergraduate research, or an internship with General Electric. Jonathan was a technical writer and oversaw any edits to documents.

Alden Ford is a dual Computer Science/Information Technology & Web Science major. He has experience in full-stack web software development across multiple tech stacks through an internship at Sandia National Labs and multiple large projects. Alden was the front-end developer and his focus ranged from designing the interface with wireframes to developing the React application that would display all the relevant information and functionality to end-users.

Andrew L’Italien is a dual major in Computer Science and Information Technology and Web Science major. He has experience with software development, UI design, and full-stack web development, the latter being established through multiple long-term projects done at RPI. Andrew was a back-end developer and performed cloud research and development for this project.

Kolby Fisher is an Information Technology & Web Science major concentrating in Web Technologies. His technical background includes over a year and a half working in information technology security for Progyny Fertility Benefits, as well as academic experience with technologies directly relevant to this project, including front-end development, API access, and secure, optimized database transactions. Kolby was a technical writer and back-end developer who focused on documenting the software development process and assisting with the technical writing portion of the project, as well as working in AWS to set up server instances that the developers used.

Varun Ravichandran is a Business Analytics major with a minor in Information Technology and Web Science concentration. His technical background includes working for 6 months with Mohawk Fine Papers in Cohoes on their ERP implementation, and working full time now with Stellantis, formerly known as Fiat Chrysler Automobiles in the IT Department. His skill sets include some technical skills such as SQL, JavaScript, Python, HTML, as well as business decision making skills. Varun was a technical writer who focused on creating the presentation.

# **Section 5: Problem Statement**

ECS found that their existing enterprise resume tracking technology frequently failed to meet expectations. In addition to candidate tracking, as a government contractor, ECS uses the resumes of its employees to "sell itself" to potential business clients. However, their current resume tracking system makes it difficult to get an overall picture of their employees' skill sets, and employees often fail to update their resumes in a timely fashion.

# **Section 6: IS/IT Solution**

To address the needs of ECS, the team created a browser-based desktop web application for reviewing and managing resumes. Applicants or employees login and upload their resumes to the cloud through the website’s interface. Administrators have access to these resumes and have the ability to filter and search through them.

The system back-end focuses on parsing user-uploaded resumes in either a Microsoft Word or Adobe Acrobat PDF format, allowing for a better understanding of the candidate. Resumes are stored in the cloud and are parsed through an EC2 instance. This resume parsing happens on the server side, utilizing Node.js and Python 3.9 for this functionality. Depending on the file type, different methods of text extraction are used to handle them. For PDF formatted resumes, information is extracted directly from the binary PDF file via Adobe’s optical character recognition, OCR; however, as a fallback, OCR libraries can be used to also extract this text if needed. Microsoft document-based resumes are parsed via external Microsoft document interaction libraries for text extraction.

Once the resumes have their text extracted into an easier-to-manipulate format, pre-processing, tokenization, and stop-word filtering are applied to the corpora to extract meaningful tags and attributes about a given candidate. This extracted information is then stored in a knowledge graph, a network of facts and attributes which links resumes to their skills as well as linking applicants together via these skills.

The front-end utilizes React with the Material UI library for user interface components. The front-end focuses on displaying the parsed information back to the administrators where they can view, search, filter, sort, and download these documents based on keywords, Boolean operators, or relevancy. For those with access to the search functionality, it allows users to search resumes for skills using Boolean queries; for example, a search query may look something like Query: (*(React \* Angular) & !python*), which would search for employees who know either React or Angular, but not both, and who don’t know python.

# **Section 7: Methodology**

The project client, ECS, was contacted by Sean Hale who had familiarity with the organization from his time there as an intern in the previous summer. Sean was able to reach out to ECS on behalf of the MITR team to inquire if there were any business needs that could be solved with technology. To ensure it was a good fit, the team discussed strengths and weaknesses of various members, and selected roles based on these discussions. These discussions ensured that work would be assigned to those who were strongest in those specific areas.

The team met with ECS at the beginning of the project to discuss finances and to gather information about the future impacts that the project may have. Using this information, a CBA was created to analyze the numbers and calculate important results such as the net present value and internal rate of return. The calculations made in the CBA highlighted the benefits and risks of the project and helped determine the added value to ECS.

The development methodology employed involved a variety of software to keep track of progress, updates, and issues faced by the team. Additionally, the team employed a hybrid of two project management methodologies in the project: Agile and Waterfall methodologies. The different technologies used include GitHub, Git, Gantter, and Discord. GitHub was used as a central repository for any relevant code and acted as a Kanban board in the issues section which made any ongoing issues or updates much easier to understand and manage. Git was used together with GitHub and made it possible to implement version control into the project. Gantter acted as another way to visualize the team’s progress on the project, informing of any upcoming deliverables for MITR or for ECS. Finally, Discord acted as the primary location for any team communication in the forms of texting or using voice chat to host weekly meetings.

The team met consistently every week using Discord for team meetings and Microsoft Teams for ECS meetings. During the team meetings, the team would discuss any issues or key updates, and would establish the work that would be accomplished within the next week. The client meetings kept ECS’ employees up to date with the progress being made on the project, as well as providing an opportunity to demonstrate the application twice during the development process. ECS would often point out any pros and cons of the application during the demo and would give feedback on what they thought was good, and what could be improved. Occasionally, the team would split into smaller groups to work on implementing certain parts of the project. Team members were assigned roles during the beginning of the project and worked closely with each other to integrate their working parts by scheduling one-on-one calls. One team member worked on the front-end, while another worked on the back-end, and another handled using AWS hosting. Once completed, these individual parts were connected to form the full application.

# **Section 8: IS/IT Requirements**

The team met with ECS to discuss potential features and IS/IT requirements which was the foundation to create a formal requirements document. Following meetings were held to review the document with ECS and to solidify both the functional and non-functional requirements. After reviewing the IS/IT requirements, the team and ECS settled on a set of requirements that was unanimously approved by all stakeholders.

## Functional Requirements

* Process
  + The website must have an authentication system.
  + The website has different pages and features for administrators and regular users.
* People
  + There are two main users who use the website. The first is the manager, and the second is a regular employee.
  + Managers are given higher up access to the website and can perform a broader variety of tasks.
  + Regular users can be seen as employees.
* User Interface
  + The user interface consists of different features depending on if the user is a manager or employee.
  + Managers have access to more web pages with additional features.
  + Regular users should only be able to edit their own resume through the resume database.
* Database
  + The database is a NoSQL (MongoDB) database instance hosted by AWS and stores all resumes as well as other documents.
  + The resume parser has access to the database and can search through uploaded resumes for necessary skills, certifications, or qualifications.

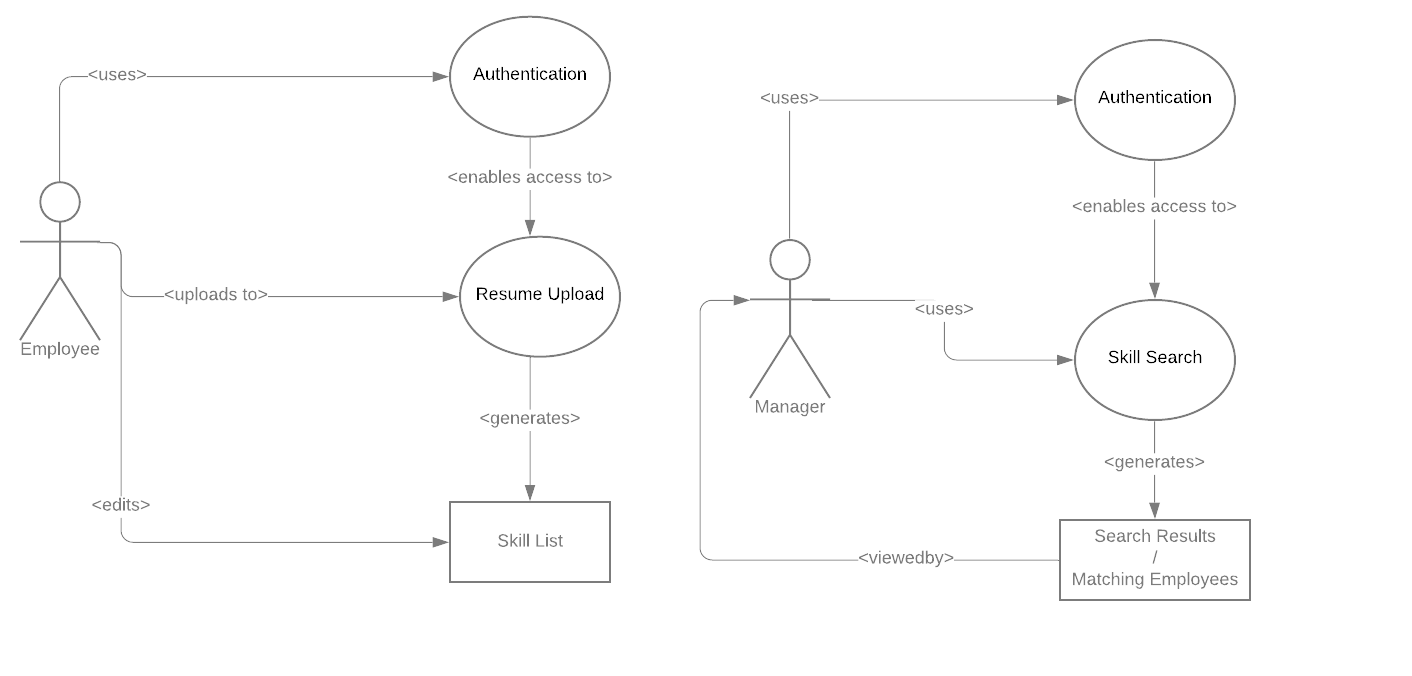
## Non-Functional Requirements

* Hardware
  + Hardware provided through AWS (EC2 instances)
* Software
  + Use the MERN Stack (consisting of MongoDB, Express, React, and Node.js).
  + Use AWS for various forms of hosting, making it easy for the team to work in tandem.
* Networking
  + Users need an internet connection to be able to use the website.
* Scalability
  + The application scales to meet demands given the amount of activity on the website.
* Security
  + AWS Users must be given permission to use certain services upon request.
* Look/Feel
  + The website has a design that closely resembles ECS and its website.
  + Create a usable and user-friendly website that the clients and employees can easily navigate.

# **Section 9: IS/IT Design**

## Use Cases

There are two primary use cases for the resume parsing tool as shown in **Figure 9.1**. First, employees are required to login and upload their resume to the system. It parses their resume and displays the skills parsed to the user, and they can edit that list if it does not correctly reflect their skill set. Second, a manager can log in to access the search feature for the database. This enables them to search all employees for specific skills, or sets of skills, using Boolean logic searches, and view the list of matching employees.



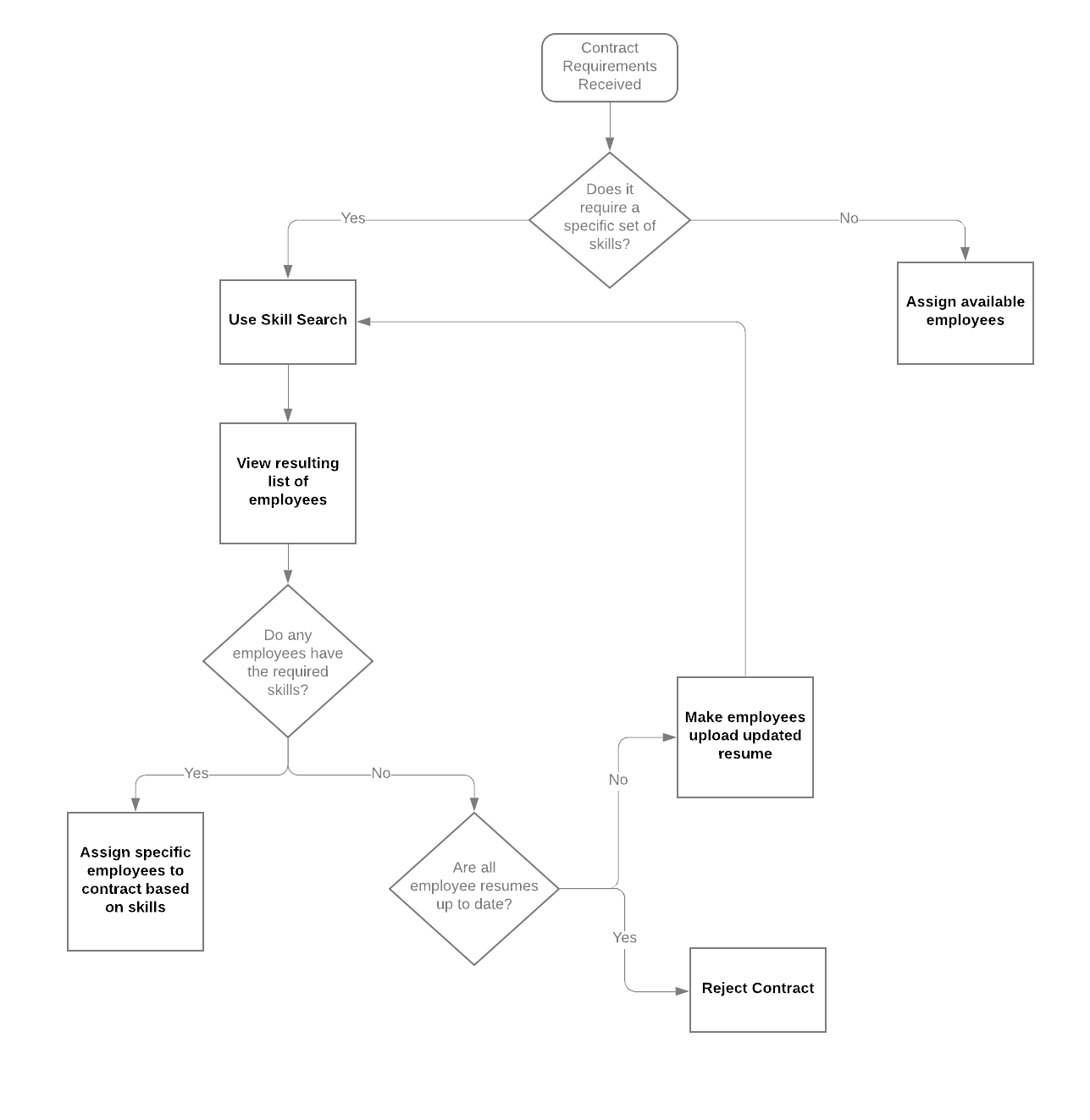
**Figure 9.1. Use Case**

## Back-End Design

The back-end of the project uses an Express server running on Node.js and was chosen to complement the front-end’s technologies which is talked about in the front-end design section. The server handles requests via API endpoints, which intake data and resumes and output the data requested, such as skill lists or resume output. For certain tasks, the server calls Python 3.9 resume parsing script, which returns a JSON object when a resume is uploaded. Each feature exists as an API endpoint; for example, the search feature intakes a query and returns an array of MongoDB ID keys.

## Workflow Diagram

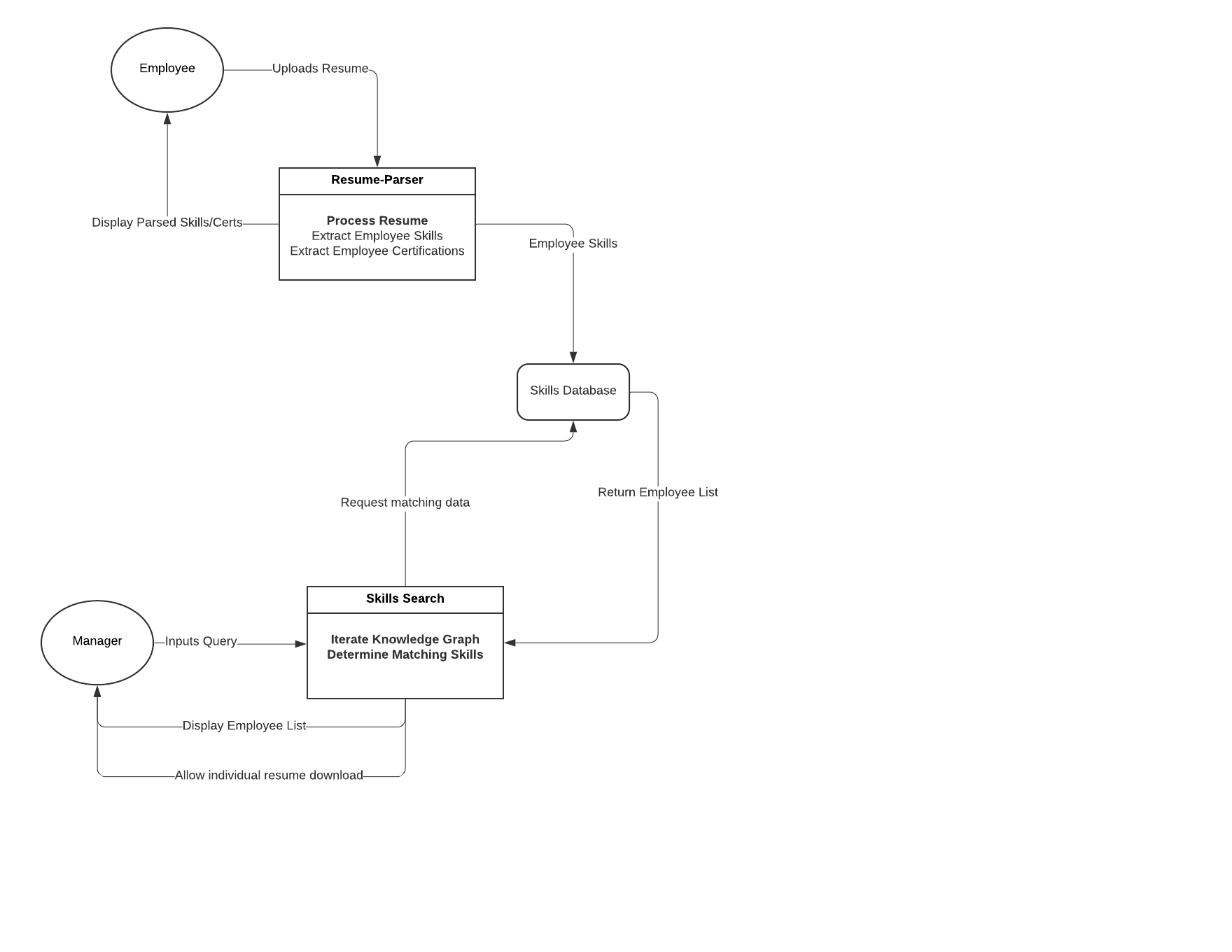
The flow of work for this tool is quite straightforward, as shown in **Figure 9.2**. When a contract is received, a manager evaluates it to see if it requires a specific set of skills. If not, then the resume tool is not used. If it does, then the manager uses the search feature to determine whether or not they have available employees that can fulfill the requirements. From there, they can choose to assign employees to the contract, update employee resumes, or reject the contract entirely.



**Figure 9.2. Workflow Diagram**

## Data-Flow Diagram

The resume tool’s focus is to intake data in the form of resumes from employees of the organization, as shown in **Figure 9.3**. After data is parsed, it can be edited by the employee to ensure that their skills are correctly recorded. After that, managers can analyze the data by running searches. The tool also allows managers to download individual resumes that meet their criteria.



**Figure 9.3. Data-Flow Diagram**

## Database Design

The database used is MongoDB, meaning it is unstructured and consists of individual documents of data, as shown in **Figure 9.4**. MongoDB was chosen because of its effectiveness to store resumes and its ability to easily integrate into the other back-end technologies. Each table above represents one document, which may hold various forms of data. The largest table is “employee data” and stores several strings of information about the employee, as well as a reference to that specific employee’s resume document. The specific resume document is stored in binary under the resumes table, in a one-to-one relationship with the employee document. It also stores a list of skills of the employee. References to the resumes are also stored in a many-to-one relationship with the arrays in “resume\_skills” and “skills\_assoc”. This allows for quick searches for resumes that contain a particular set of skills. Finally, there is a table for strings that were parsed out of resumes that are not skills maintained to decrease parse time.

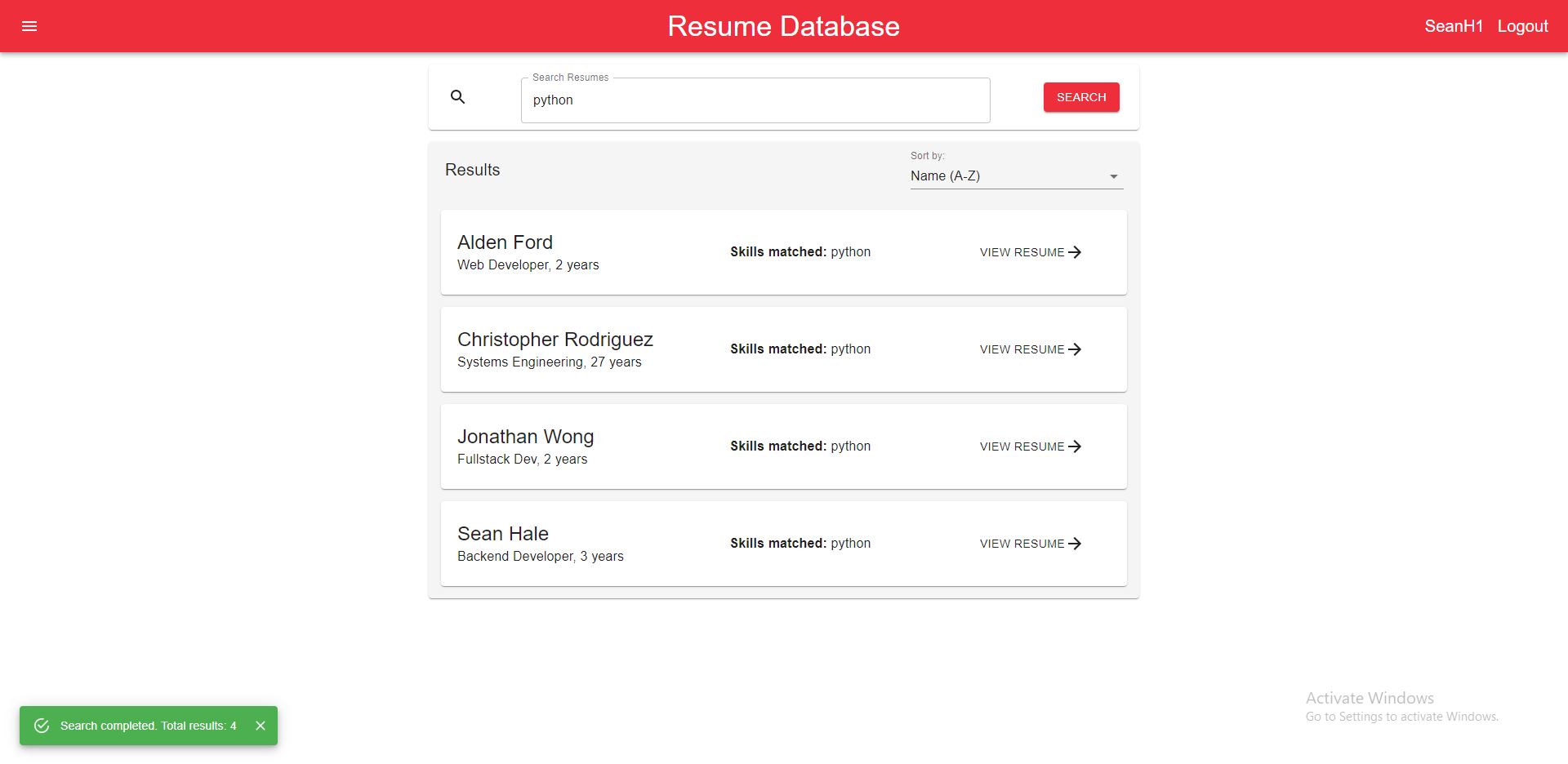


**Figure 9.4. Database Design**

## Front-End Design

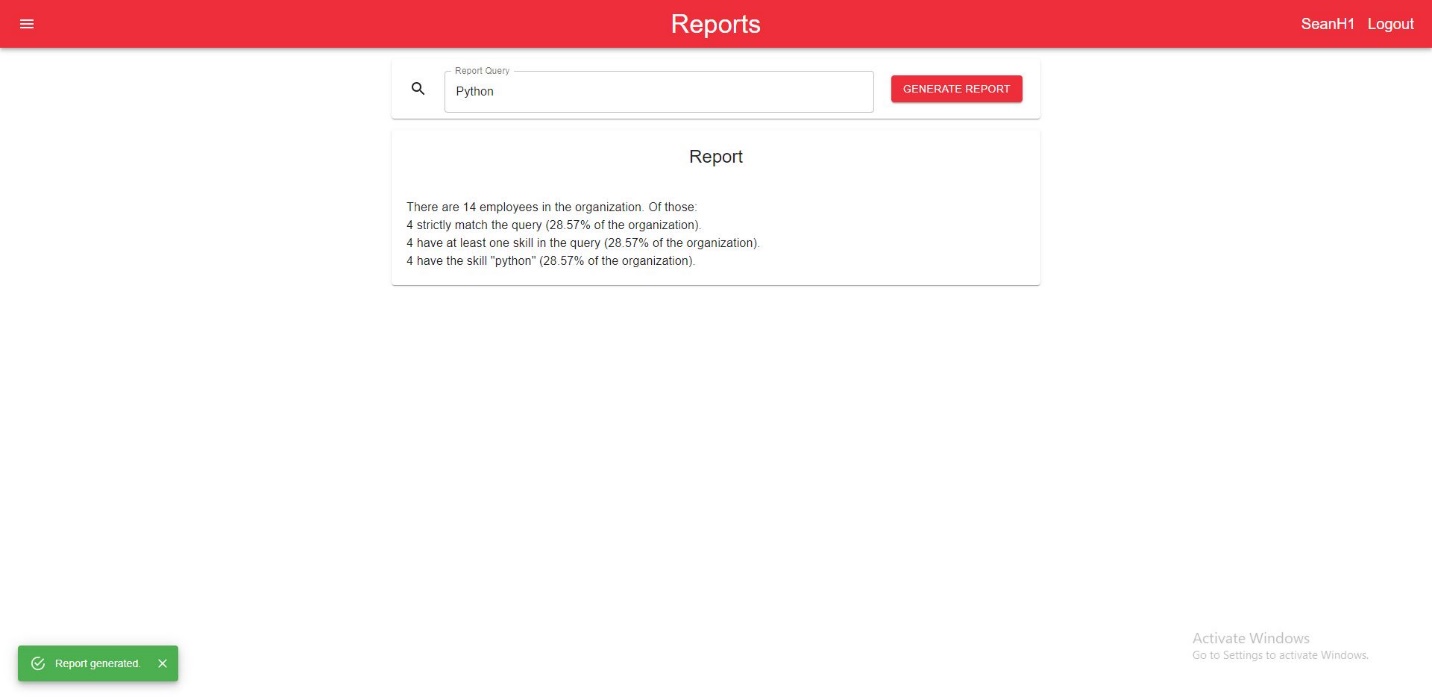
The front-end technologies included React and Material UI, which were both technologies that easily integrated with the back-end. The team created three pages to be accessed by managers or employees who must be authenticated by ECS’ authentication system. The users are then brought to different pages depending on their permission status which can be found in the different figures below.

**Figure 9.5** illustrates the landing page allowing certain employees, such as HR or managers, to filter through submitted resumes. This can be done by interacting with the search bar which updates the results found in the table below the search bar. Managers have the ability to narrow down potential candidates based on the skills they possess by using the advanced options feature and searching for specific skills.

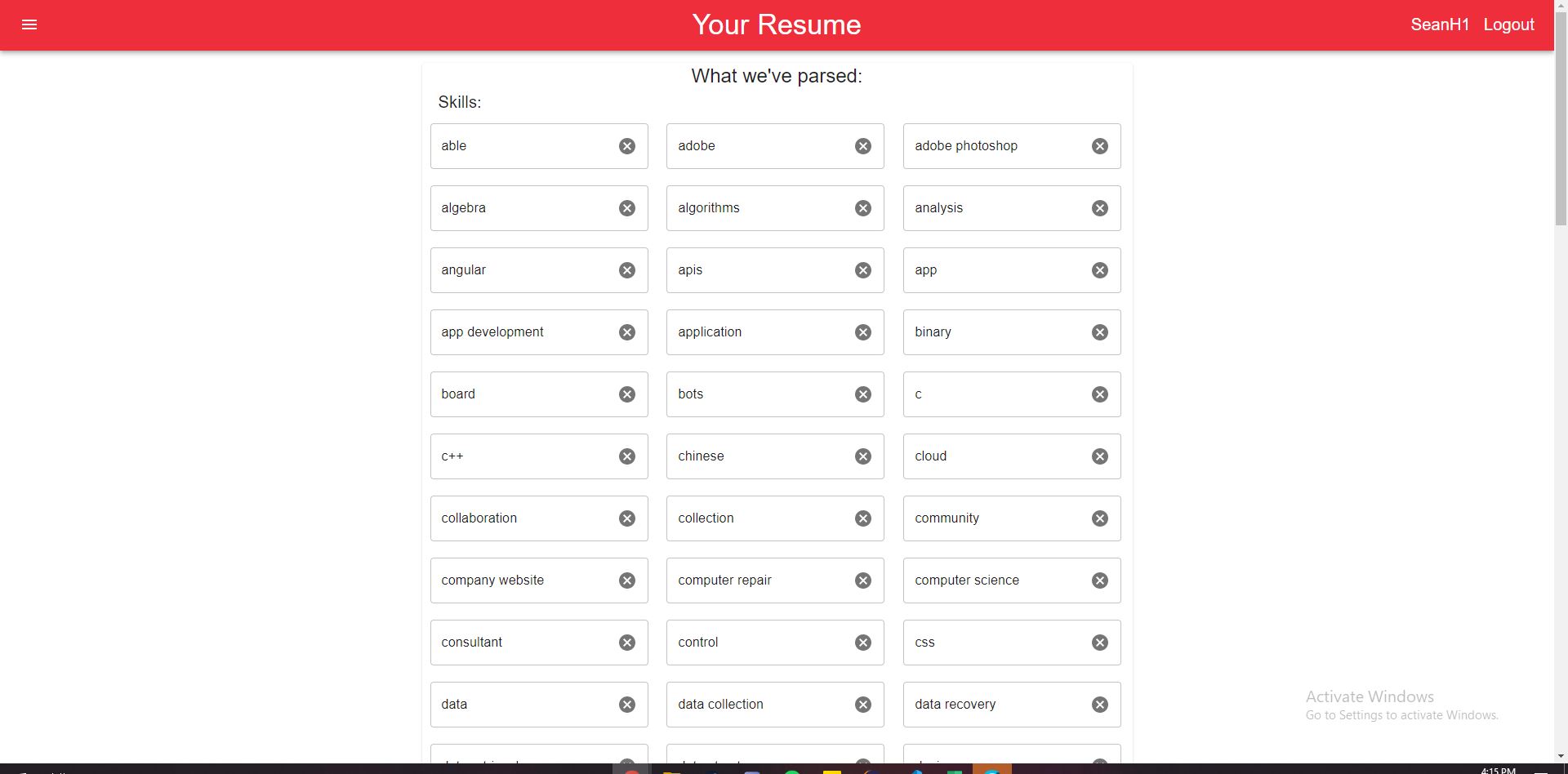


**Figure 9.5. Resume Database**

**Figure 9.6** is a reports dashboard that allows managers to get an at-a-glance overview of the prevalence of a skill or set of skills within the organization. Managers can generate ad-hoc reports using the same syntax as search queries to display the total number of resumes ingested and how many are able to match their search criteria.

**Figure 9.6. Reports Dashboard**

**Figure 9.7** is the resume dashboard, which allows for employees to upload their resume using an interactive drag-and-drop or file picker feature. Once uploaded, the resume parser runs and displays all relevant experience which falls under categories such as “Skills” and “Your Experience” sections within the “What We’ve Parsed” table. Employees can review their parsed skills and make corrections as needed.



**Figure 9.7. Resume Dashboard**

## System Prototype

These specifications can be found in **Section 8: IS/IT Requirements**, under the non-functional requirements subsection. The specifications include how the team managed to deploy the application with the specific services. ECS handles all future installations and deployments of the system.

## Code Base

The team used GitHub as the central repository for storing code, reviewing pull requests, and managing the project plan board. Only one GitHub repository was needed for the project, which the team collectively used for the reasons listed above. Additionally, per request of ECS, the repository was to be made public and open source and could be used by anyone, including ECS. The GitHub repository can be found using this link: <https://github.com/sean-hale-dev/resume-org>.

## Test Plans

To test the project, the team used a variety of methods depending on the functionality being tested. The search feature relied on a Boolean logic parser which was tested with the laws of logic such as the Commutative law, Negative law, and DeMorgan’s law. Once the parser succeeded at all laws of logic, it was deemed complete. For other features, such as document uploading, the API endpoints were tested with many valid cases, as well as edge cases that could come up, such as an employee uploading an invalid document. Additionally, the team created a clickable URL which allowed ECS to test the application through AWS. ECS noted some issues such as the possibility of managers not knowing how to perform queries in the search bar. The team took these issues into consideration and resolved them with a training session and documenting search queries in the documentation. Once all valid documents provided by ECS were parsing successfully, and invalid documents were rejected correctly, it was deemed complete.

## Test Results

All the features in scope have been thoroughly tested by the team during code reviews made on pull requests.

# **Section 10: Cost Benefit Analysis with Risk Management**

The team created a CBA after talking with the client about the feasible present and future financials for the project scope. Numbers were attained that would reflect both the benefits and costs of implementation allowing for calculation of the Net Present Value (NPV) at a 12% discount rate. Given the numbers that were provided by the client, the CBA showed that the NPV at a risk-adjusted discount rate of 12% would be approximately $2,600,000, and the Internal Rate of Return would be 2,400%, reflecting the fact that the benefits far outweigh the costs of development.

Costs  
Costs were broken down into five main categories, as discussed with the client. These categories represent the sum total of costs that the development, hosting, and maintenance of the tool would incur.

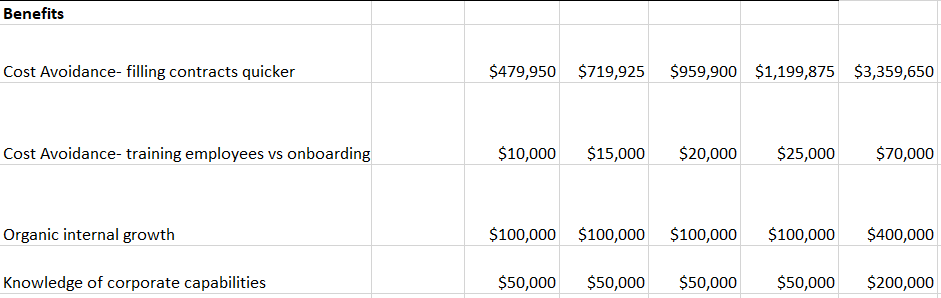
The first category is student development labor. Industry rates for web development and back-end development for contractors are around $50 per hour, and the team estimated that the tool would take a total of three hundred hours of development and testing. The second category is client labor which included the time that the client spent in meetings with the team, as well as time spent on their behalf discussing project requirements and other overhead. Third, there are the technological costs for the testing environment that was in use during the development cycle. The team used an AWS billing tool to determine that the cost would be in the range of $1,000. The fourth cost is ongoing cloud hosting, once the project is completed and turned over to the client. The team discussed with the client how many resumes would likely be in the database, and approximate numbers for the frequency with which the database would be queried, and estimated a yearly cost based on the total amount of data stored. Lastly, the team made estimates for future software maintenance by ECS. Due to the documentation that was included with the code base, the time to learn the code base is low, and thus any potential fixes are likely to be as minor as updating packages. For more precise line items, see **Figure 10.1** on the following page.

  
 **Figure 10.1. Cost Estimations**

Benefits

The benefits of the tool were determined after lengthy discussions with the client surrounding current internal work flows and potential for improvement. The team determined that there were four main areas where the client would benefit after completion of the tool.

The first area is cost avoidance; currently when ECS receives a contract proposal, it takes 57 days on average to find employees that fill the contract requirements. The client estimated that this loses the company $959 per day that the contract goes unfilled, and that the tool will allow for contracts to be filled significantly faster. The second area also covers cost avoidance, in a different way; currently ECS is occasionally driven to hire new employees to fill contracts when current employees cannot. This incurs additional cost in the form of onboarding the new employee, and the tool can help to fill those instances where ECS was unable to find a current employee that met the requirements, thus saving on cost. Third, an ECS employee mentioned there was an intangible value to being able to bid on organic growth for the company, enabling more data-based promotions and other internal optimizations. Lastly, there is the intangible value of being able to quantify the capabilities of the corporation, the estimate was created based on the value of an individual user’s information using estimates from advertising agencies multiplied by the number of users that will interact with the tool. For more precise line items, see **Figure 10.2** below.



**Figure 10.2. Benefit Estimations**

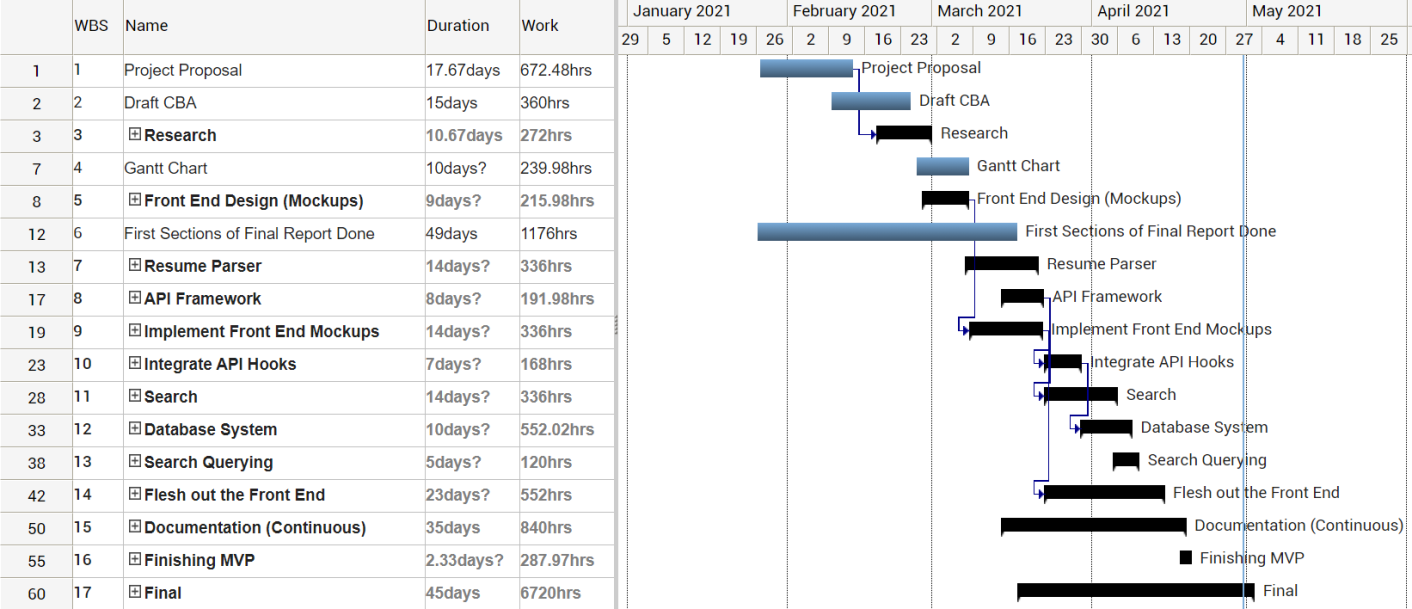
Risks

Project risks were evaluated in three main categories: technological, resource, and implementation. Risks include budget troubles, loss of important stakeholders, and time or requirement issues.   
The risks were evaluated based on their impact on the project- losing a key developer is high impact, as it would jeopardize the team’s ability to complete development, while budget issues were lower impact, as the costs for the project are low. Many of the risks were low impact; if the project ran over budget, the client has stated they would provide additional budget along with a “stern warning.” Based on these risks, the team determined that the relative risk of the project is quite low, and the majority of risks could be mitigated with regular diligence and communication. By industry standards of risk index, this means that banks would loan money to the project at an approximate rate of 12% interest, which is the rate used for NPV calculations.

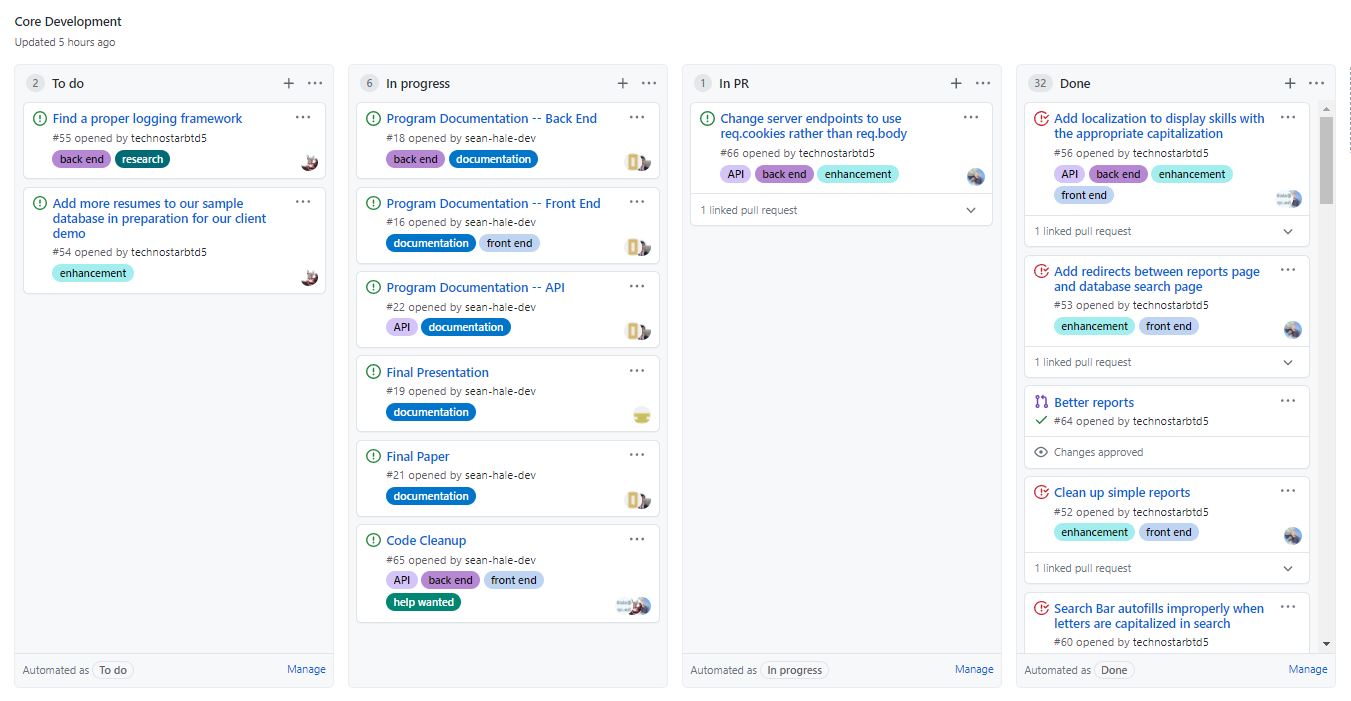
Mitigation  
While many minor risks can be avoided or prevented, a portion of the risks cannot be properly mitigated. For example, the highest risk is the loss of an important stakeholder. This is largely out of the team’s control, as the risk scenarios include client employees potentially leaving ECS or a team member having a life-altering event. Risks that can be somewhat mitigated include a team member failing to complete work involved, requirement miscommunications, and project complexity.   
To mitigate risks, the team developed a number of strategies. To avoid a team member being incapable of completing their work, the project manager regularly checked in on how each member was doing, ensuring that the workload was accomplishable. To prevent requirement miscommunication, client meetings were regularly held to discuss and update based on development progress. To prevent the project from being too complex to complete in the allotted time, the team was careful to avoid scope creep, stick to the initial set of requirements, and only focus on add-ons after the minimum viable product was reached.

# **Section 11: Project Plan/Schedule**

The team followed a hybrid of Agile and Waterfall methodologies when creating the application. The team used Gantter, a waterfall cloud-based project management software, to keep track of deliverables and tasks which were assigned to each team member. Additionally, the team also made use of an agile Kanban board on GitHub as a centralized project board to make live changes. As such, there were many main categories the team used to organize the project plan and to create milestones. The main phases as outlined in the Gantt chart are as follows: Research, Front End Design, Resume Parser and so on. In each of the bolded sections in **Figure 11.1**, there are subsections which outline in detail what needs to be accomplished and by when.



**Figure 11.1. Gantter Project Board**



**Figure 11.2. Kanban Board**

As soon as the team was created by the MITR professor, the project started with creating a project proposal for the first 17 days. This phase consisted of finding a prospective client and determining the project to be worked on for the semester. The team began by gathering potential clients found through mutual connections or online postings. Ultimately, the team decided to pursue ECS, who offered a project that was enticing and would offer a beneficial learning experience. Once ECS agreed to allow the team to undertake their project, the team set up a meeting with ECS to talk about the logistics of the project. In this meeting, the team talked about four sections: the client organization, project team, problem statement, and the information system and information technology solution. These sections would officially kick off the team’s final report draft which would be handed in as an assignment for the class and to ECS during the conclusion of the project. Additionally, the team created a CBA to help visualize the different types of costs, benefitis, and risks posed by creating the application.

Once the project was set in stone, the team began to research how to best implement the solution using the technologies chosen, including the front-end, back-end, and cloud tools. This task took about 10 days to complete and helped the team to understand the underlying infrastructure of the application and how each section within the project worked individually and together. The team also focused on fleshing out the Gantt chart once all research was concluded and it was confident about the proposed deliverables, such as the front-end design mockups, and the software development tasks.

After the Gantt chart was completed, the team began to design mockups to visualize how the website would look. Once the mockups were finished, the team presented them to ECS where approval was given to begin the developing portion of the project. The team began development on the front-end and back-end, which can be seen in the Gantt chart starting with the Resume Parser task and ending on the Fleshing out the front-end task. Each of the developers set up their coding environment and began working on their assigned tasks as indicated in the Gantt Table and Kanban Board. The front-end developer worked on creating web pages that best aligned with what was designed in the mockups. The back-end developers worked on many of the tasks described in the Gantt table, such as integrating the API hooks, working on search querying, or integrating the database.

As the development portion of the project was coming to an end, the technical writers began to write up the documentation provided with the application. The documentation covered both the front-end and back-end aspects of the website, including how to interact with the website and performing the search queries in the search bar.

The final phases of the project were to finish the MVP, which included tidying up the application and handing it over to ECS. The team also completed the final report, which was handed in as an assignment for the MITR class, and also given to ECS.

# **Section 12: Post-Turnover Plan**

The team met with ECS on April 30th to handover the application and to discuss the post-turnover plan. The team discussed all necessary steps the future developers and managers should know of when interacting with the application. As such, it was established that the application was easy to edit and had a minimal learning curve once handed over. This was aided with the documentation created for each of the sections in the repository which detailed functionality and the specifications of which they were built off. Part of the documentation covered how to properly install and run the source code on ECS’ machines. Other parts covered how to interact with the application interface, essentially teaching employees how to use and navigate the interface. Since there were two primary users for the application, the administrator/manager, and the employee, each have different outlined training material in the documentation. The team also demoed the application and provided a brief training seminar to senior management and to the employees who would be using the application.

The team plans to keep in contact with ECS after the application handover if there are any bugs discovered. If a user discovers a bug, ECS has the ability to contact one of the team members and coordinate a time to discuss why the bug occurred and steps to combat it. This ensures that the handover is smooth and allows ECS to properly go over the documentation to test the application and make sure that both the administrators and employees know how to use it. The team agreed to offer support to ECS for 1 month after handing over the application.

# **Section 13: Results and Client Feedback**

The team completed the project well before the application hand off date of April 29th. The core deliverables agreed upon during the first couple of meetings were completed as well as some optional features discussed throughout the semester. Some of the main features include being able to query the resume database based on keywords and Boolean operators, viewing the queried resumes in an ordered list or by individual results, and filtering or sorting based on relevance. An optional feature that was added was a front-end that supported pages for administrators. While ECS did not require a front-end for the application, the team found that it would be easier to navigate all the features the application offered in a simpler manner by implementing a minimalistic design. ECS’ color scheme was also used in the development process to align with the company’s various software products.

Both the back-end and front-end were written with the ability to continue development after the project, something ECS recommended and appreciated during the application handover. Additionally, during each weekly team meeting, ECS gave the team feedback on any updates or problems that were brought up. The team's live application demos allowed ECS to provide more specific feedback. At the conclusion of the project, ECS acknowledged the team’s dedication and work throughout the semester and were thankful for taking on this project. ECS was grateful for the team’s work and have provided the team an opportunity to stay in contact and potentially work for the company in the future.

# **Section 14: Conclusions**

This project served as a great learning experience for all members of the team and allowed for personal and professional growth. The team’s willingness to take on a project where some of the required technologies were unfamiliar showcased determination and the ability to quickly adapt, which ECS acknowledged.

The team completed the application in a timely manner thanks to the weekly team and ECS meetings. There were a few aspects of the project that worked well, including weekly meetings and researching certain services and technologies before implementation. The team meetings allowed each team member to clearly communicate their updates, problems, and future deliverables. This was useful because it added a sense of accountability for each team member to accomplish their weekly project deliverables. Team members who were assigned new technologies that they were not familiar with researched its viability for the application. As a result, the bad services and technologies were weeded out in the beginning and allowed the team to choose the best ones for the project and ECS’ endeavors.

The team also faced some challenges throughout the course of the semester and had to adjust accordingly to adopt a schedule that would meet both the team’s and ECS’ likings. For one, the team created an ambitious schedule for the project during the beginning of the semester. However, some of the deliverables on the schedule fell short due to the deadlines coming up too quickly due to unforeseen circumstances such as team members completing assignments in other classes or doctor’s appointments. This led to some team member’s pushing off deliverables for later which affected others’ work on the project since they were dependent on those deliverables. However, after thorough communication and leadership in the weekly team meetings, deliverables were being met on the deadlines and the team carried through with the rest of the project on time.

Given the nature of the project and the short amount of time to complete the project, one thing that the team agreed would be good to achieve next time would be to get a head start on work whenever there was free time. More times than not, some aspects of the project were left untouched until needed to be accomplished, which piled up work for some team members. If more deliverables and work on the project were completed before the unforeseen circumstances listed above occurred, then there would be a lower probability of the team not meeting deadlines and adjusting the original schedule.

Despite the rollercoaster of successes and challenges, the team persevered throughout the semester to complete ECS’ application on time.