// Notice the length of the report and the conciseness

CS 130: Software Engineering

Professor: Miryung Kim

TA: [redacted]

---

jobSpot

Part A Report

Team Members

[redacted]

October 13, 2016

Project URL:

1 Motivation

// Clearly states the problem the application tackles, where and how it would be used

As computer science students ourselves, all of us go through the recruitment cycle every year. Since the response rate for submitting applications online is fairly low, many applicants will send off their resumes to a lot of companies at once. Searching for and applying to companies or startups can be a very time consuming task. Also, at a certain point, when juggling interviews, followups, and offers from numerous companies, managing and scheduling all of them becomes unfeasible to handle through email alone. As of now, there are no good application where job seekers can keep track of all their applications and interviews. A few might resort to online planners or Excel spreadsheets, but this method is very cumbersome/decentralized and can be lacking in organizational and specialized features such as reminders or calendar views of upcoming interviews and deadlines.

Our proposed product, jobSpot, would solve this issue for job applicants. Our vision is to develop a centralized and feature-rich web application that gathers all of a user's job hunting status, interviews, recruiter's email, and other information into one personalized dashboard. We also plan to help the user save time with job hunting by having a curated list of popular job opportunities, with the application link provided. All it would take is one click to add that application to their personal pool to be tracked. Our proposed browser extension can also save the user time by having them fill out commonly asked information only once, instead of every time they fill out a new application.

2 Feature Description and Requirements

//You need to define your features more clearly. For instance, you could mention any possible application status and how it is changed when different actions are taken

Our product is composed of three features: a personal application dashboard, an available positions list, and an auto-filling browser extension. We will now describe each of these components in greater detail.

2.1 Personal Application Dashboard

// Thorough description of usability with multiple applications

This dashboard allows a user to see all of his or her current job or internship applications that are in progress. Each application includes a status, such as "Interview Scheduled," a date and time for the interview, offer information, and recruiter email(s). This will also include a calendar view of all dates entered by the user, including upcoming interviews, application deadlines, and career fairs. Each application will also be color coded based on its status, and can be sorted and filtered based on user queries.

The user interacts with this dashboard by either adding an application for a specific company, or adding a generic event such as a career fair or tech talk. The user supplies needed information about an application, such as its status, relevant dates, and contact information. The user will also provide needed information about career fairs or tech talks, including a date and location. Applications and events added to a user's dashboard can easily be moni-tored, modified, or deleted as new information pops up.

When a user starts receiving offers, they can track their compensation packages and response deadlines with the app. The user will be able to easily sort and compare compensation between various companies to help them make their decision.

(Feature Extension) The dashboard can also send out automated alerts through email or SMS when a deadline/scheduled event is approaching, if the user wishes.

2.2 Available Positions List

This list provides a job- or internship-searcher with a list of commonly applied-to positions at major companies and startups, along with direct links to their online applications.

A job seeker interacts with this list by finding an available position through sorting or searching the list, and making one click to easily add this position to their personal application dashboard.

A company representative interacts with this list by signing in to verify their legitimacy with their company and then adding, removing, and editing available position information. This user must provide information about the position, such as a title and description, along with a direct link to the position's online application.

2.3 (Project Extension) Autofilling Browser Extension

This extension will help with automatically filling commonly-requested information for job applications, such as personal information (name, address, etc.), resume, GitHub / LinkedIn / personal website links, past employer / experience, etc. All fields are optional.

A user will only have to provide information once to the extension (and update information if needed). Then, whenever a user is applying to a job, he or she will press the extension's icon in the browser. The extension will then attempt to autofill fields on the current page using the user's saved information.

2.4 Usage Scenarios // A good variety of scenarios are discussed

//The usage scenarios should a statement of purpose of that particular scenario. As well as assumptions about equipment or software. Scenarios, which are describing some of user interaction with the proposed system, should be specified as series of steps.

An example user of our app would be a college student looking for a summer internship. There is little cost in applying, so this student goes on the career fair website and applies to every company that he or she has heard of. A week later, this student would like to apply to more companies but has completely forgotten which companies he or she has applied to. As well, this student has been getting many interview schedules and has lost track of when his or her availabilities are.

Another scenario would be a very strong student who has many upcoming offer deadlines. This student would often lose track of when the deadlines are. In addition, when recruiters ask the student about competing offers, he has to dig up all the old emails to compile a list of competing offers he has. With our app, these functionalities would all be automated

under the “Deadlines” tab.

Another example user of our app would be a college student looking for a part-time job. This student needs some quick cash and is willing to take on any job, so he/she wants to mass apply jobs near her. Unfortunately, many of the job applications have pages upon pages of necessary information, so this student can only apply to 2 jobs per hour. With our autofilling browser extension, students can now apply to jobs with a much higher efficiency.

Employers can also use our app to post an available position onto our database. This allows them to advertise their openings to more students, and gives them a way to directly update and contact job applicants on the status of their applications.

2.5 Diagrams

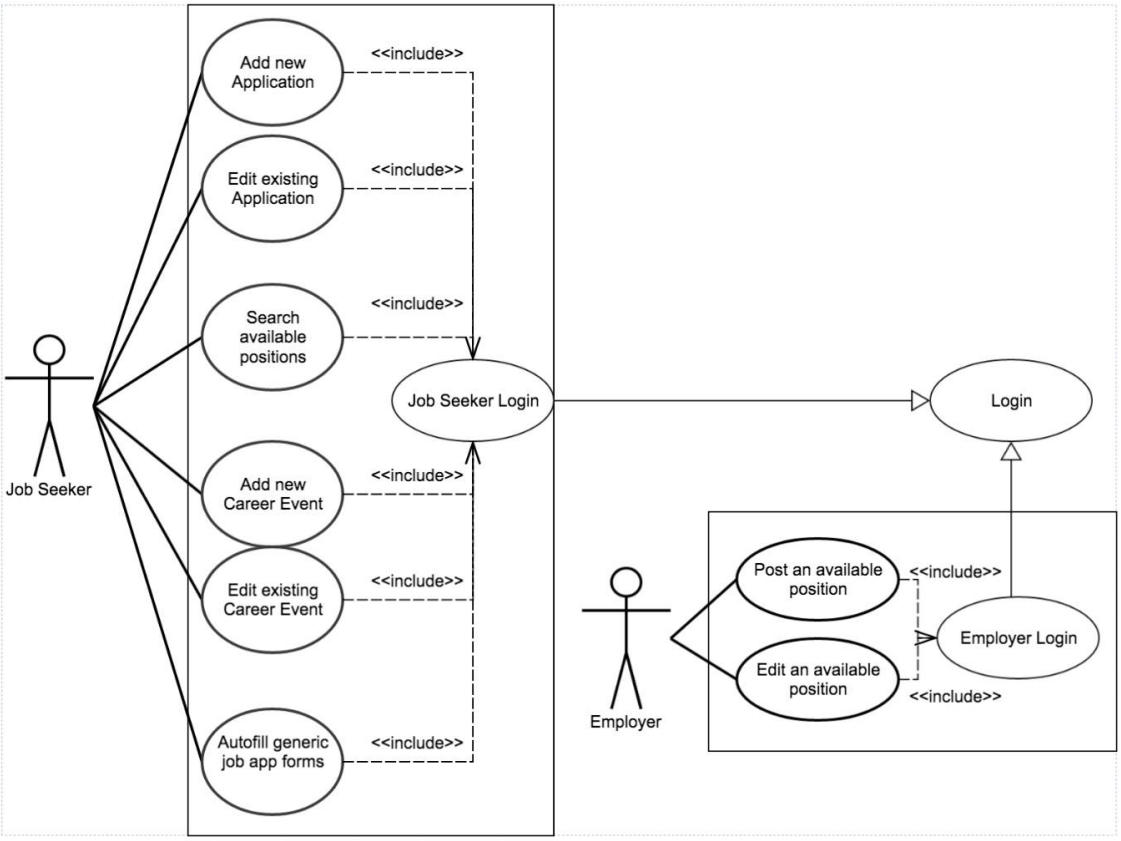

Figure 1: Use case diagram for our application. As we can see from the figure above, there are two possible types of users: job seekers and employers.

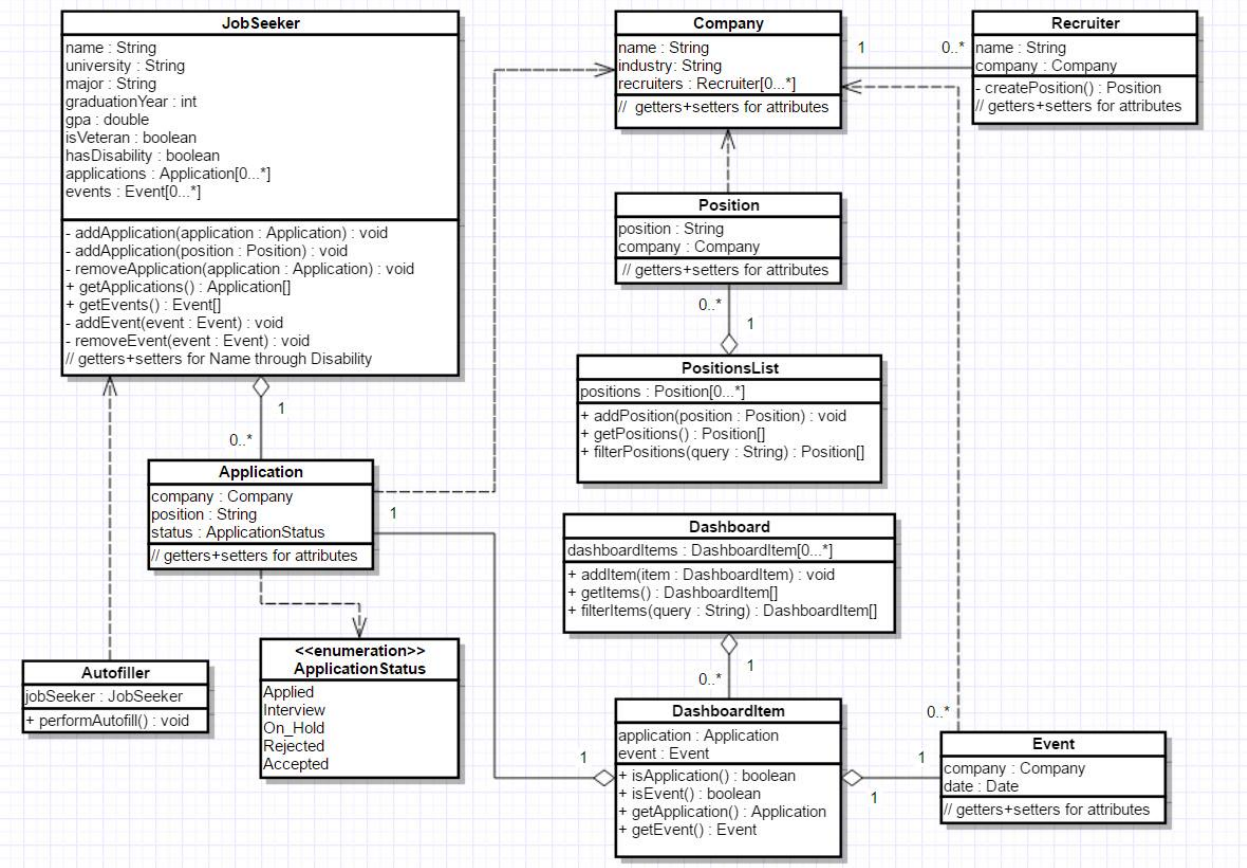

Figure 2: Class diagram for our application.

/* Use case Diagram: “extends” keyword is needed for login use cases inheritance.
Class Diagram: You can use inheritence for handling different kind of DashboardItem. With your current design, one of the DashboardItem fields remains null whether it’s an event or application. Each job seeker should own a dashboard (composition). Also, the job seeker should access the application through the dashboard and dashboardItems (leaving the responsibility of filtering and sorting applications to dashboard, as you designed). Recruiter and Company relationship should change to composition (If a company is removed from the database, all of the recruiter objects should be removed as well). */

// Interpretable and understandable mockup

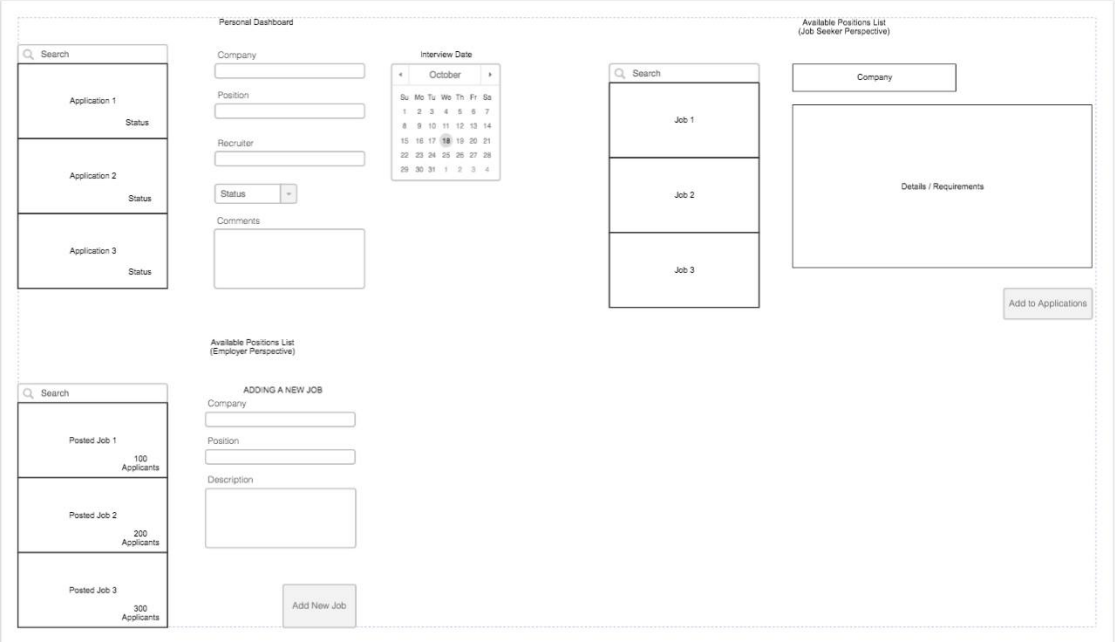

Figure 3: Design mockup for our application. On the upper left is the mockup of the personal application dashboard. The upper right contains the mockup diagram for the available positions list from a job seeker's point of view. Finally, the bottom left contains the mockup diagram for the available positions list from an employer's point of view.

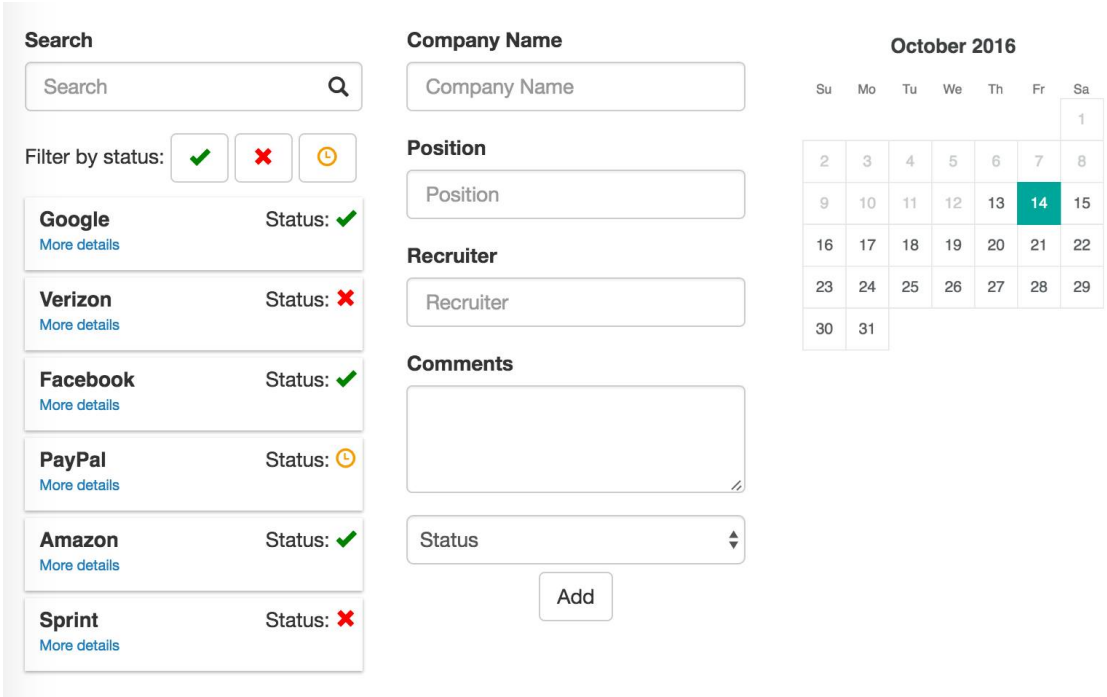

Figure 4: Detailed design mockup for the personal application dashboard.

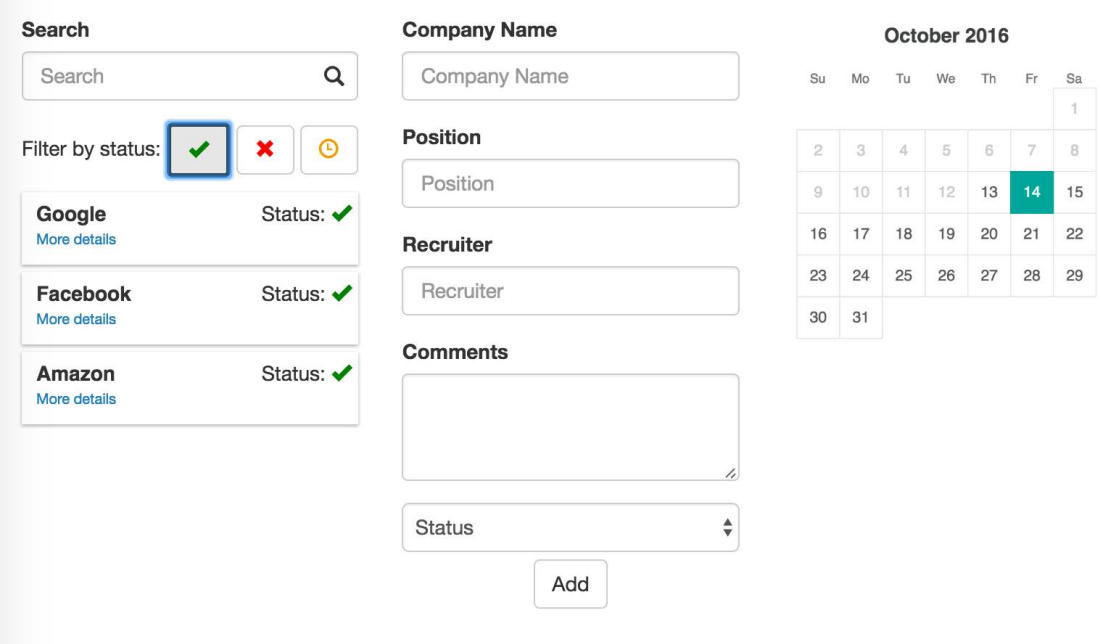

Figure 5: Detailed design mockup for the personal application dashboard, with filters turned on.

3 Feasibility // Feasibility also includes future extensions

This project is feasible to implement because, at its core, it is a standard web application involving a backend of data storage providing information for a standard responsive frontend. In addition, we've decided to use JavaScript for all parts of this project, front and back end, to simplify connecting everything together. Finally, we've divided the project into three distinct components: a personal application dashboard, an available positions list, and an auto filling browser extension (to be implemented if time allows). This division allows us to separate work between team members and not require each person to be able to contribute to everything.

4 Capability // Well described backgrounds and well-defined roles

Our team will be able to implement this project due to a diverse background of members including some very knowledgeable in web development along with a mix of experience between front and back end work.

[redacted] practiced with full stack development when he worked on mobile apps in his spare time, and has done backend work while interning with Hulu and Google. He will be learning JavaScript and Node.js for this project for creating the server side of this application, in addition to experimenting with the browser extension. For part A of the project, [redacted] did half the write-up with [redacted] and added details to the UML diagrams.

[redacted] has developed the backend for numerous projects in node.js, and has done full-stack work while interning at Esgyn as well. Some of the node.js projects he has worked on include FitRPG, a browser based RPG game that gamifies exercise and healthy living by rewarding players automatically for steps they recorded on their FitBit, and Loops – an innovative way for college students to trade textbooks and save money through algorithmically found trade circles. For this project, [redacted] will be working on the backend and helping out with the front-end as well. For part A of the project, [redacted] helped write this report and typeset it with LaTeX.

[redacted] has worked as a full-stack web development intern at Ticketmaster. He has worked on redesigning and implementing the Java-based backend infrastructure for ticket recommendations, and used React to develop the new front-end for the ticket selection and checkout process. [redacted] is also very well-versed in JQuery and NodeJS as he has utilized them in many personal web app side projects such as a Movie Recommendation Engine and Task Management software for his high-school. For this project, [redacted] will be one of the lead full stack engineers as he will contribute to both the front-end skeleton and back-end infrastructure of jobSpot. For part A of the project, [redacted] implemented the preliminary designs for both the general use case diagram and wireframes of the UI.

[redacted] has developed web applications with Daily Bruin, as well as worked on front-end design with Angular and database/cloud integration during his time as a software intern at Qualcomm. He has built various web/mobile(iOS) apps in his spare time, so is familiar with JS and other basic web development tools. For this project [redacted] will be working on the backend and helping out with the front-end as well. For part A of the project, [redacted] wrote potential use cases, and made the presentation powerpoint.

[redacted] has experience with web development and had recently worked on web scraping using PhantomJS, a javascript framework, during his internship at Symantec. [redacted] also has experience building desktop applications and knowledge of Android app development. [redacted] will be working on both front-end and back-end in this project. For part A of the project, [redacted] helped with the UML class diagrams and the powerpoint presentation.

[redacted] has worked as a web developer for the Daily Bruin. Most recently he helped rewrite bruinwalk.com. He also did full-stack development as a software engineering intern at Factual Inc. He will be helping with the backend and frontend. For part A of the project, [redacted] helped with mockups and writing requirements.