A Scalable, Portable Web Application for a Community Project Marketplace

http://yale.projectsboard.io

Author:
Rafi Khan, B.S. Computer Science, expected May 2015.
rafi.khan@yale.edu

Adviser:
Daniel Abadi, Associate Professor of Computer Science, Yale University. dna@cs.yale.edu

December 9, 2014 // Yale University
1 Abstract

The purpose of the Yale Projects Board is to help students work together on projects at Yale. Those with project ideas can post them to the Projects Board and create a set of roles (called openings) to be filled. Then those who are looking for projects to work on—either to earn extra income, to bolster their resumes, or simply for fun—may browse the postings and get in touch with the leaders of interesting projects.

Our priorities for The Projects Board were not just to scale to many users—our current implementation should scale to tens, if not hundreds of thousands of users—but to easily port to different communities. We believe such a website will be most effective if it takes advantage of the trust of an exclusive group of people: in this case, Yale University. The Projects Board is built as an API protected by CAS authentication, but that authentication scheme can easily be replaced by any other to create a similar community at other universities or even entire neighborhoods. In the months following our launch, we can easily imagine harvard.projectsboard.io, or newhaven.projectsboard.io.

We’ve also included a administrator-review feature, which allows site administrators to control what can be posted to site, thereby keeping the community safe and productive.

Because of our particular feature set and access to the Yale community, the Projects Board has the potential to connect individuals and groups unlike ever before, creating an open space for collaboration in a diverse forum of possible projects. The Board is currently live at yale.projectsboard.io.
2 Features

The Projects Board offers the following set of features:

To All Users  
*Anyone on the internet*

- The ability to search through available openings
- Sort and filter search results

We made this feature available to the public as a demo. Those who might not have CAS login or want to get a preview before logging in can still get a sense of the site and what it has to offer.

To the Yale Community  
*Anyone with CAS credentials*

- Create profiles about themselves, including affiliation with Yale, personal bio, relevant skills, LinkedIn/Github URLs
- Contact the project leaders of openings in which they are interested
- Keep a favorites list of attractive openings
- Post a project (described below)
- Post openings (described below)
- Contact other members on the site via email

Site Administrators  
*Those maintaining the live website*

- Approve projects before they appear in search results (described below)
- Delete unwanted openings or projects

2.1 Posting a Project

Users can create projects, which represent the higher-level goal they are trying to achieve. Example projects might be titled “Yale Daily News Website”, or “Discovering a New Mapping Algorithm”. To deter “trolls”, or people posting vulgar or fake projects, each project must be approved by site administrators before its openings show up in search results.
2.2 Posting an Opening

Projects themselves do not show up in search results, openings do: openings are the low-level, specific roles that project leaders are looking to fill. Example openings might include things like “Frontend developer”, or “Research assistant”. Openings include a name, description, pay amount, pay type (lump sum, hourly, volunteer), and timeframe (term-time, summertime, full-time).

3 Technology

The Yale Projects Board was built using Ruby on Rails with a PostgreSQL database. We employed the use of the pg_search gem for search, the rubycas-client gem for CAS login, the paperclip gem for resume uploads, and the thin gem for our webserver. For testing, we used rspec-rails and factory_girl_rails gems. The front-end was coded using the Angular.js Javascript framework, and SASS for CSS.

3.1 The Data Model

Our data is represented with 5 main models: User, Project, Opening, Skill, and Favorite.

User
The User model holds all of each user’s data, as well as his or her relationship to projects and openings. Users have a many-to-many relationship with projects, which are all the projects the user is leading (projects may also have multiple leaders). Thus, users also have many-to-many relationships with openings, through projects.

Project
The Project model holds each project’s name and description, as well as a reference to each leader. The Project class has a one-to-many relationship with the Opening class.

Opening
The Opening model holds information about each opening, and belongs to a project.

Skill
The Skill model is a representation of all the possible skills that a user could have, or an opening could require. We have begun with a set list, and will grow that list as more skills are necessary to represent the talents of our users. Skills have a many-to-many polymorphic association with a Skillable, which can either be an opening or a user. In the case of an opening, the skills represent the necessary talents required to complete the job. In the case
of a user, they represent the talents the user has to offer.

**Favorite**
The **Favorite** model is a way for users to mark certain openings as favorites, to come back to later. This model has no properties, and essentially acts as a join table between the **User** and **Opening** models.

### 3.2 Search

The search feature makes extensive use of PostgreSQL’s fulltext search feature. By default, user queries search through an opening’s name, description, associated project, as well as its associated skills. Similarly, a query for users searches the user’s name, bio, and associated skills. The gem **pg_search**, by Casecommons, facilitated the creation of this feature.
3.3 User Interface

A large portion of the user interface was completed by my colleague Bobby Dresser, and is outlined in his report.

3.4 Testing

Ruby on Rails makes test-driven-development easy, and using the RSpec testing framework, we have written tests for all of our backend functionality. The testing framework makes sure of model validity—for example, all openings must have a name—and behavior: upon creation, all openings set their expiration date one month into the future. More importantly, the tests check for authentication (only administrators and project leaders may edit a project or its openings) and the correct behavior of the API (calls to URLs with certain parameters return the expected results).

3.5 Email

Email is a necessary, but sometimes hard-to-build, feature of a web application. Using the Sendgrid API and Rails’ ActionMailer class, we built in HTML email notifications for contacting users and approving projects. As the application grows and gets more traffic, managing email correspondence will become increasingly important.

4 Scalability

Because the Yale Projects Board is intended to operate within a relatively small community, we created it to efficiently scale to thousands of users, though given the strength of the Ruby on Rails framework, there is no reason it should not scale to tens, if not hundreds of thousands of users. Instead of scaling to many users, however, it’s more important the project will scale to many communities, which could each have a custom login and their own database for users, projects, and openings. Because our entire application is an API, porting to a different community would only require creating a new authentication scheme, a new database, and minor front-end tweaks. CAS, like many authentication schemes, is cookie-based, so if the Projects Board were ever to move to another community (harvard.projectsboard.io, perhaps?), it would only require replacing two or three methods to fully customize the site.

5 Security

As the administrator of any website knows, the internet, even within a community such as Yale, is not void of malicious people. All activity that involves inputting data or viewing
user information is protected by CAS login, which serves to prevent unwanted activity from the internet at large.

5.1 Project Approval

Furthermore, we felt it was important that each project require administrator approval, and administrators retain the power to edit and remove any postings to keep spam low and the quality of the website high. Administrators cannot, however, edit user profiles. We made this choice knowing that only the users themselves could edit their own profile, and users should have the freedom to portray themselves as they liked on the site.

5.2 Expiring Openings

We also needed a way to remove openings that had already been filled. One way is for users to manually mark an opening as filled. However, projects leaders might not always be diligent about doing this, so all openings automatically expire after 1 month. After that period, projects leaders have the option to renew that opening if the position is still open. This allows us to keep the site relatively empty of old openings, without too much inconveniencing project leaders.

6 Launch

http://yale.projectsboard.io had its soft launch to friends and close colleagues in December 2014. In the coming months, we’ll work to onboard more leaders of organizations to create postings. After a strong community of project leaders has joined the site, we’ll begin marketing to users.

7 Acknowledgments

I would like to thank my collaborator, Bobby Dresser, for countless hours of work and motivation, as well as the Project Board’s adviser, Daniel Abadi. Special thanks also to the friends who helped us test the initial prototypes, and gave invaluable feedback.