6.170 Final Project Design Document

Maddie Dawson, Heeyoon Kim, Lynda Tang, Jennifer Wu

Project Cintern

S1. Overview *and Motivation?*

System Description

Problem: There are two groups that have a hard time with summer internships: students and employers. For students, applying to internships is time-consuming and confusing. It takes forever to navigate to many different application sites and repetitively fill out the same information for each application. After applying, it's hard to keep track of which applications you have already submitted, are working on, etc. For employers, it's not always easy to advertise, especially for smaller, lesser-known companies.

Solution: Cintern offers great things for both parties. For students, it offers a centralized place to apply and search for internships, greatly reducing the amount of time and number of tabs. For employers, it helps give an audience for internships, and a quick and easy way to screen student applicants.

Key Purposes

- **Make applying to internships less complicated:** Cintern provides a centralized location for students to search for applications, reducing the need to navigate to multiple company websites.
- **Reduce time in applying:** Students can fill out a Common Application section one time, and submit it to each company, removing the need to repetitively fill out the same information.
- **Organize Applications:** Students can easily track application statuses in one place, and remember which ones they have applied to.
- **Advertise Internship Positions:** Cintern helps companies advertise job positions by providing a large audience of potential applicants.
- **Screen Applicants:** Employers can filter through applicants by school, major, and other factors. Employers can also mark applicants that they especially are interested in by starring applicants.

Existing Solutions

Indeed: One major competitor is Indeed. Although Indeed allows users easily search for and filter job postings, the app actually redirects applicants to the company's own application site. This extra step differs from Cintern's centralized and streamlined approach to job application.

LinkedIn: LinkedIn is another app that connects students to employers, but (1) students connect to employers on a personal, not company level, and (2) users cannot apply to jobs on LinkedIn.

Monster: Monster is very similar to Cintern in that Monster allows applicants to apply to companies directly through their app. The Monster signup process is straightforward but gets necessary information from the user to pass along to employers. However, Monster does not let employers create custom forms for users to fill out when they apply.

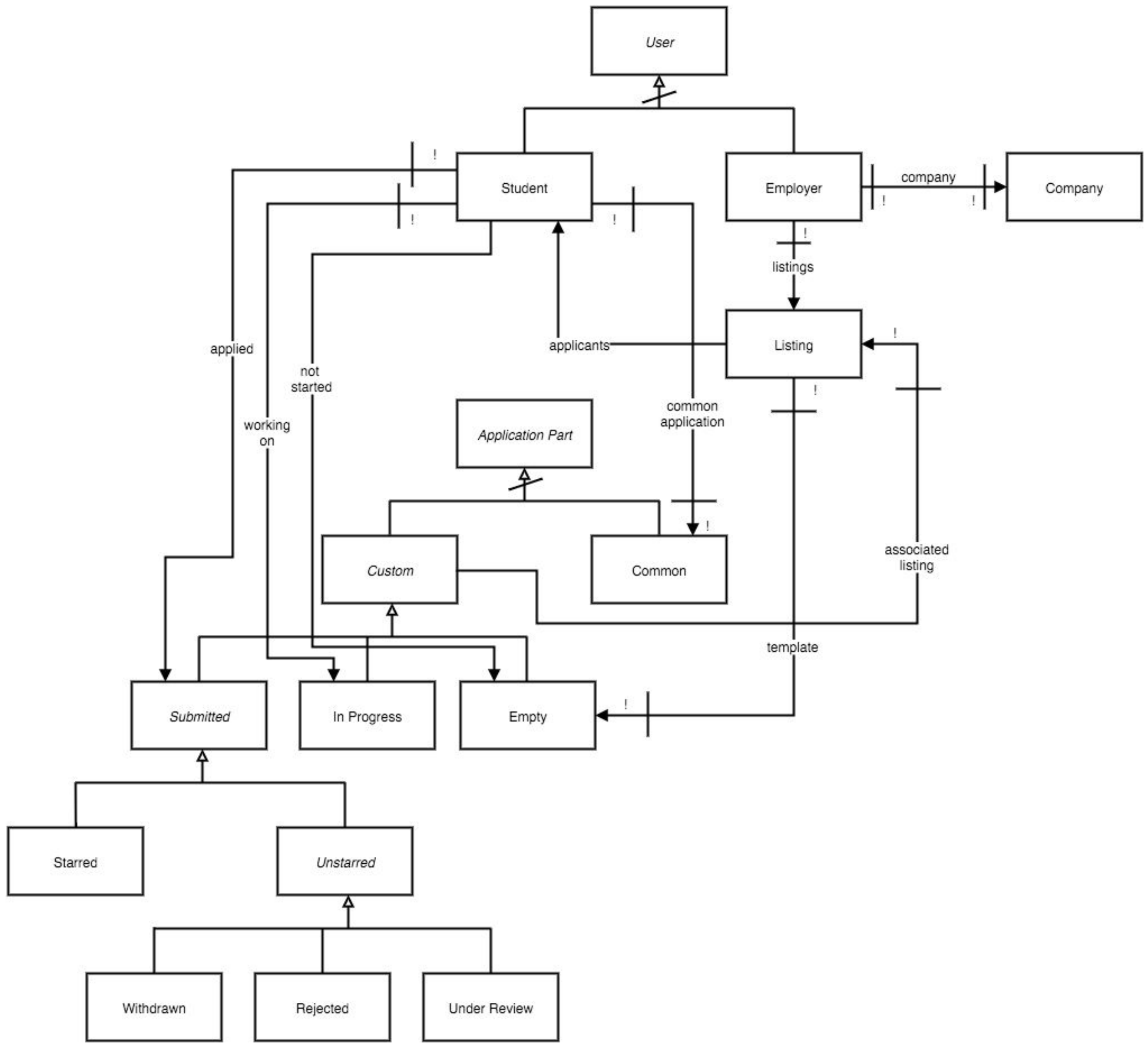
Angellist: Like Monster, Angellist allows users to apply to jobs directly through their app. However, Angellist has a couple significant drawbacks: (1) again, employers cannot create custom forms for applicants to complete, and (2) the signup process is extremely lengthy and time-consuming to the point of deterring possible users.

S2. Design Essence

Concepts

- **Common Application**
    - **Purpose**: To reduce the need of students repetitively filling out the same information for applications.
    - **Definition**: A form containing the student's info (i.e. name, school, graduation date, major, etc.) that can be submitted to multiple companies.
    - **Operational Principle**: After a student fills out the common application, the information from the form will be forwarded along and visible to any employers whose listings the student applies to.
- **Custom Application**
    - **Purpose**: To allow employers to request specific information from students
    - **Definition**: A form that the employer creates for each internship listing that details other information that the employer would like from students. Students applying to the listing are required to submit the custom application associated.
    - **Operational Principle**: If an employer would like to get certain information from students that is not on the common application section, the employer may create a custom application for the specific position and require students to fill it out before submitting. When a student submits a custom application, the employer associated with the custom application can view the information from the form.
- **Listing**
    - **Purpose**: To allow employers to advertise open internship positions
    - **Definition**: An internship posting detailing the position at the particular company, skills desired, deadline, and other information.
    - **Operational Principle**: When an employer posts a listing, the listing becomes visible to all students, who can then apply to that listing.

Data Model



Textual Constraints
For any (l,a) in applicants, where l is a Listing and a is a Student, there must be a Submitted Custom Application Part c, such that (a,c) in applied and (c,l) in associated listing.

Any Empty Custom Application Part e where (a,e) in not started has the property that there is a Listing l, such that (l,e) in template.

For a given Student a and any two Custom Application Parts c and k, where (a,c) and (a,k) can be in the set {applied, working on, or empty}, if (c,l) is in associated listing, then (k,l) cannot be in associated listing. (Essentially, this is saying that a Student cannot have more than one Custom Application with the same associated listing.)

If Listing l and Empty Custom Application Part e has the (l,e) template relation, then (e,l) is in associated listing.

If a Starred Submitted Custom Application Part s, where (a,s) in applied for a given Student a has the (s,l) associated listing relation, then (l,a) in applicants.

If Listing l and Empty Custom Application Part e has the (l,e) template relation, then for any Student a, (a,e) cannot have the applied or working on relation.

For any Rejected Unstarred Submitted Custom Application Part r, if (a,r) in applied for an Student a and (r,l) in associated listing for a Listing l, then (a,l) not in applicants.

For any Withdrawn Unstarred Submitted Custom Application Part w, if (a,w) in applied for an Student a and (w,l) in associated listing for a Listing l, then (a,l) not in applicants.

Explanation of Data Model
If a Submitted Custom Application Part is Starred, this means that an Employer has starred (aka favorited) the particular Student that owns this Application Part.

Security Concerns

Threat model:

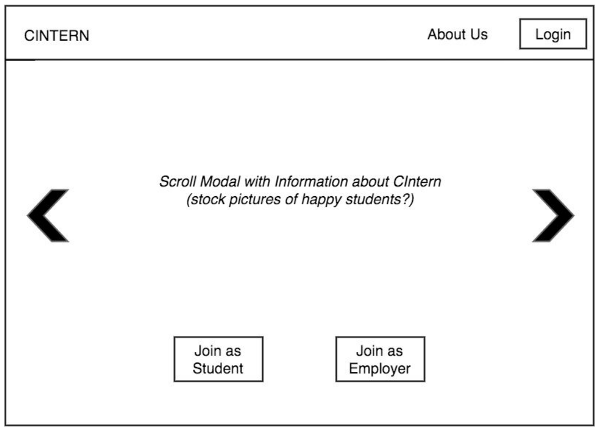
- An attacker does not have physical access to the servers or machine with another user's logged-in account.
- An attacker does have access to the network between a user and the server and can eavesdrop.
- An attacker may have some information about legitimate a user's full name, email address, and/or username.
- An attacker may try to submit counterfeit forms the way a legitimate logged-in user would submit an application form.
- An attacker may try to inject code or embed a script when submitting a form.
- An attacker may replay a request (e.g. when submitting a form) to flood the server.

- **Code injection** (e.g. SQL/NoSQL injection)
  - Mitigation: Sanitize all user input--that is, remove anything from the input that could be interpreted as code.
  - Motivation: A malicious user may want to retrieve user information from the database by way of code injection through a form.
- **Cross-site scripting** (XSS)
  - Mitigation: Escape all text so that HTML tags are interpreted not as HTML but strings.

    - Motivation: A malicious user may embed a script tag in their application submission so that the server or another unsuspecting user executes an evil script.
- **Cross-site request forgery** (CSRF)
    - Mitigation: Use form tokens to make sure only legitimate form submissions are accepted. The node.js module csurf can be used to create and check for form tokens.
    - Motivation: A malicious user may want to sabotage another user's application by submitting a fake form their behalf.
- **Brute-force login**
    - Mitigation: Rate-limit login attempts to make it harder for malicious users to brute-force guess usernames and passwords. The node.js module ratelimiter can be used for account lockout if too many login attempts are made within a certain time period.
    - Motivation: A malicious user may want to try to log-in as another user by guessing their username and password.
- **Network eavesdropping**
    - Mitigation: Use SSL/TLS encryption to make sure that people eavesdropping on the network cannot decipher the data being transferred.
    - Motivation: A malicious user may want to eavesdrop on the network after a user submits an application to retrieve sensitive data (e.g. their email) or copy their responses (i.e. plagiarism).

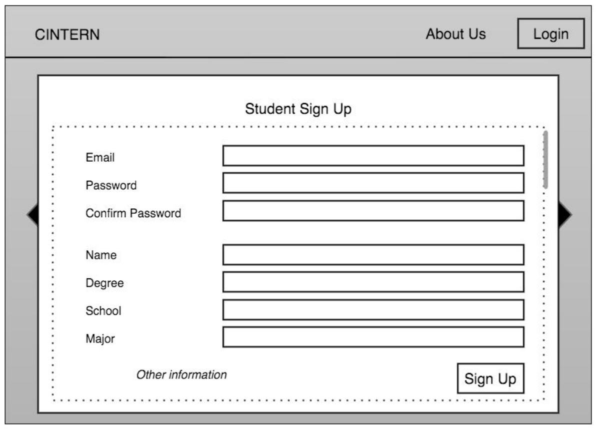
User Interface

Home Page

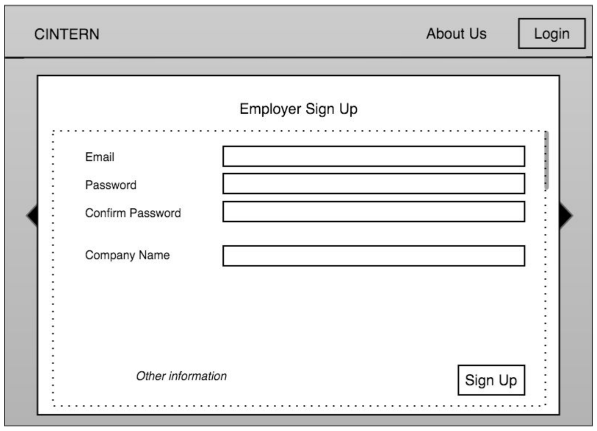


Scroll Modal with Information about CIntern
(stock pictures of happy students?)

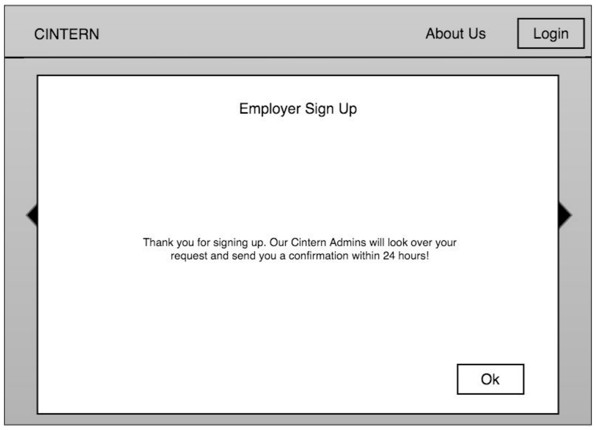
Student Sign Up Display



Employer Sign Up Display

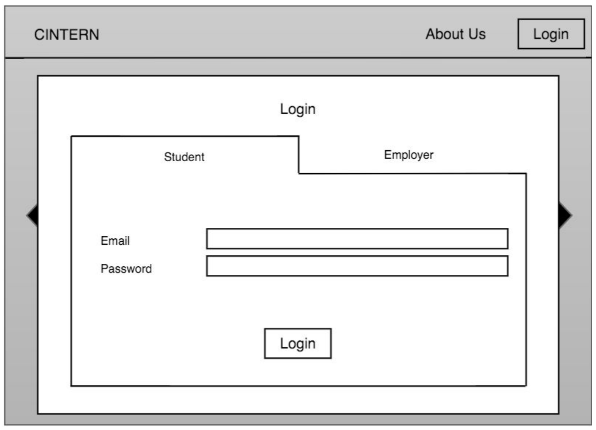


Employer Sign Up Wait Page Display

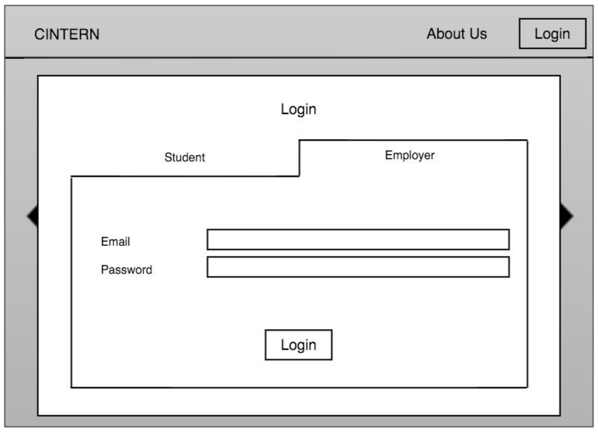


Thank you for signing up. Our Cintern Admins will look over your request and send you a confirmation within 24 hours!

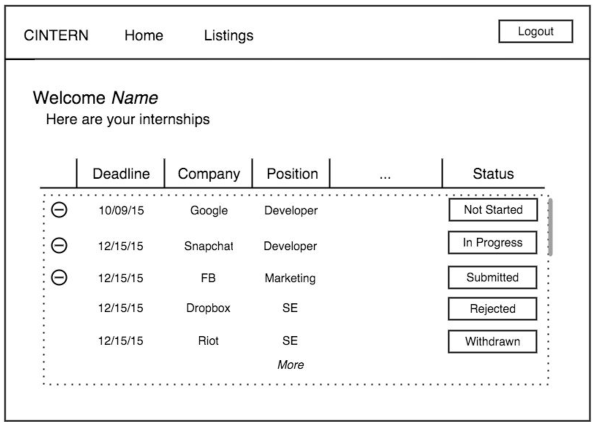
Student Login Display



Employer Login Display



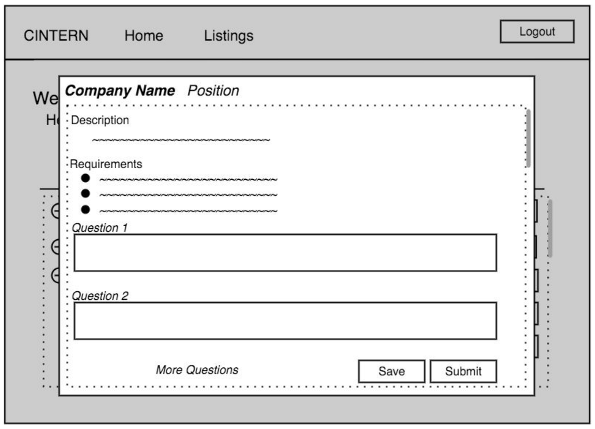
Student Dash Page



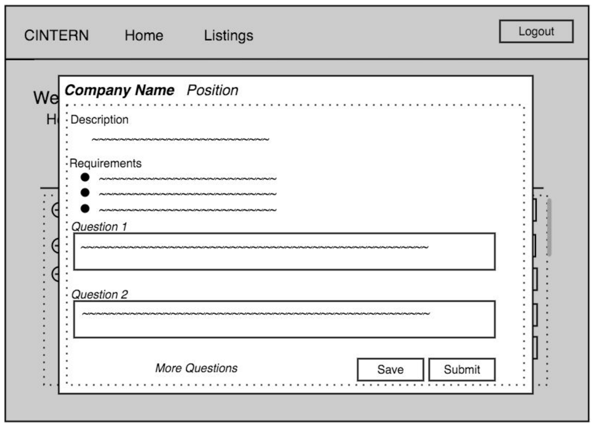
	Deadline	Company	Position	...	Status
⊖	10/09/15	Google	Developer		Not Started
⊖	12/15/15	Snapchat	Developer		In Progress
⊖	12/15/15	FB	Marketing		Submitted
	12/15/15	Dropbox	SE		Rejected
	12/15/15	Riot	SE		Withdrawn

More

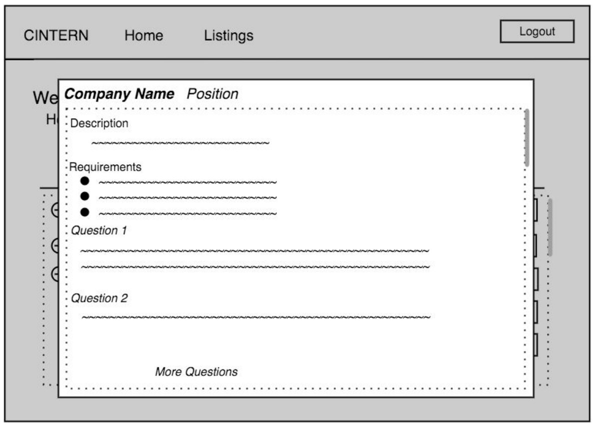
Student Application Not Started Display



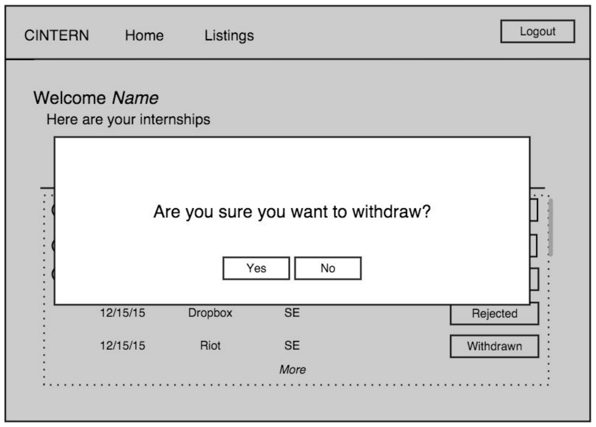
Student Application In Progress Display



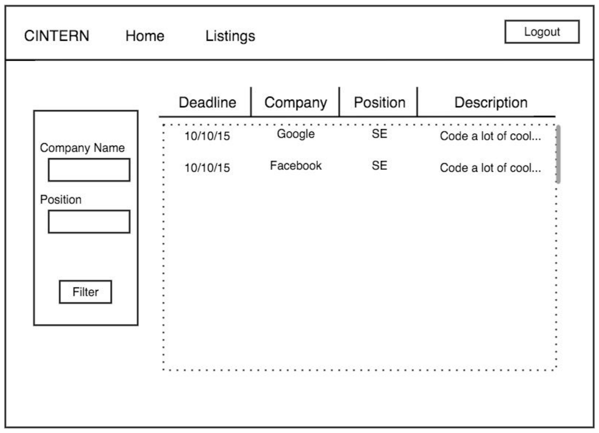
Student Submitted Application Display



Student Withdraw Application Display

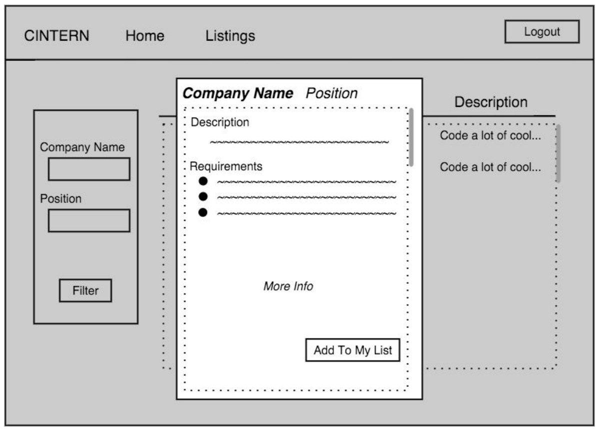

CINTERN
Home
Listings
Logout
Welcome *Name*
Here are your internships
Are you sure you want to withdraw?
Yes
No
12/15/15 Dropbox SE Rejected
12/15/15 Riot SE Withdrawn
More

Student View Internship Listings Page

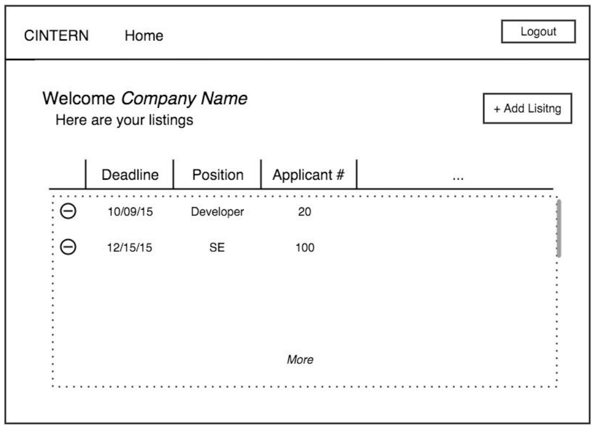


Deadline	Company	Position	Description
10/10/15	Google	SE	Code a lot of cool...
10/10/15	Facebook	SE	Code a lot of cool...

Student Internship Listing Description Display

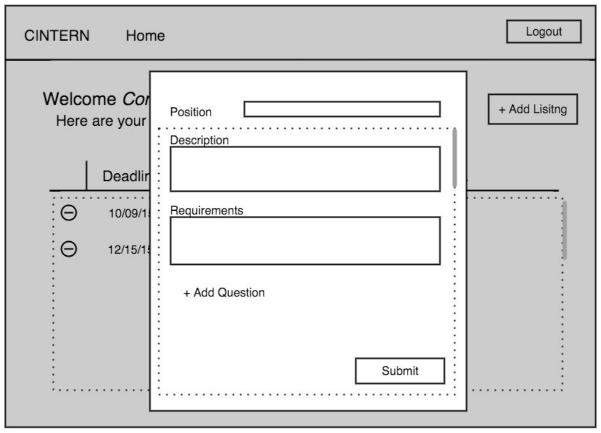


Employer Dash Page

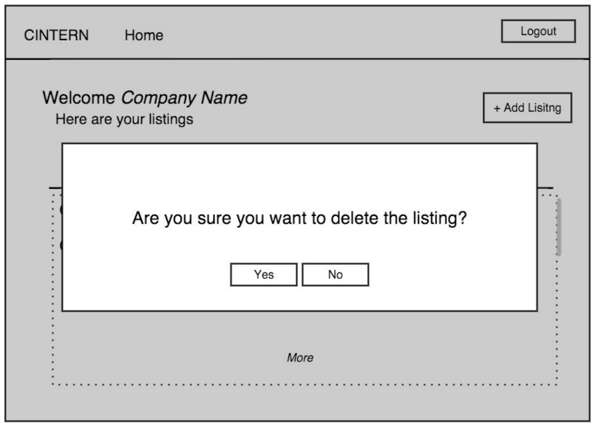


	Deadline	Position	Applicant #	...
⊖	10/09/15	Developer	20	
⊖	12/15/15	SE	100	

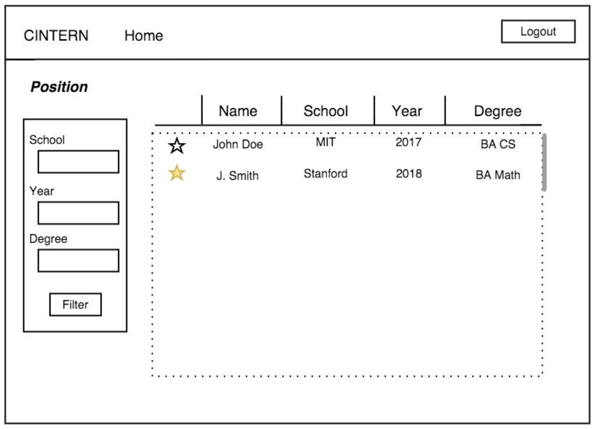
Employer Create Listing Display



Employer Delete Listing Display

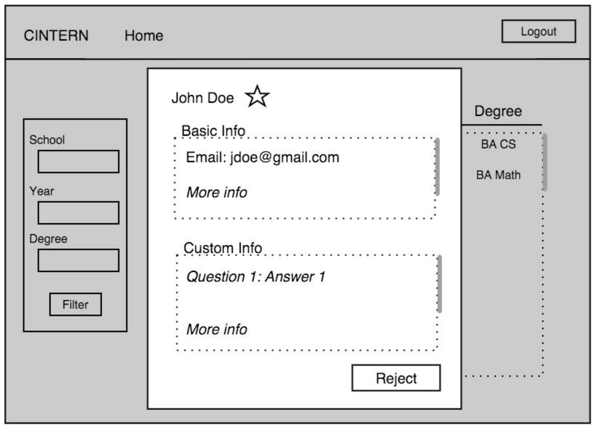


Employer Listing Applicants Page



	Name	School	Year	Degree
☆	John Doe	MIT	2017	BA CS
★	J. Smith	Stanford	2018	BA Math

Employer Applicant Display



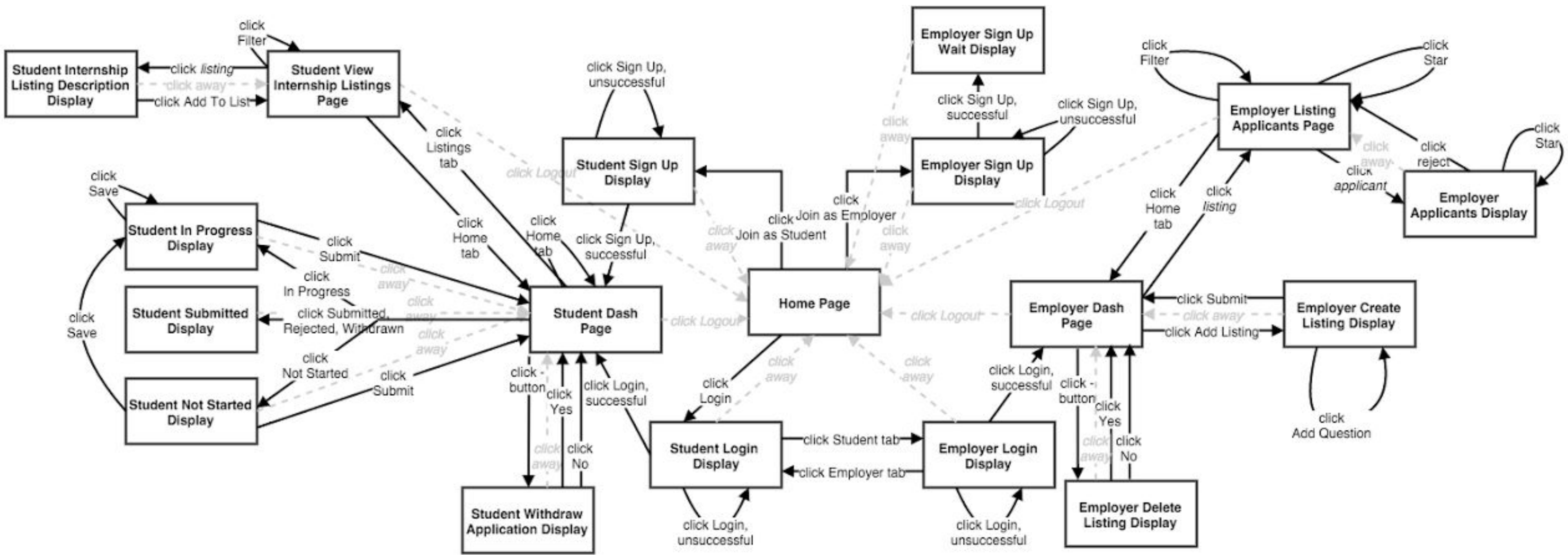
Email: jdoe@gmail.com

More info

Question 1: Answer 1

More info

Wire Frame



S3. Challenges

User Experience Challenges

Should we limit number of postings a user can apply to in a year?

- Option 1: Yes; cap the number of applications a user can submit
  - Pros:
    - Puts a cap on the amount of server space needed to store applications (for a fixed number of users)
    - Potentially saves a lot of space if malicious users try to flood the server with fake applications
    - Applicants will be incentivized to submit applications only to job postings that they are interested in
  - Cons:
    - Any limit on the number of applications will be arbitrary
    - Some users may want to apply to many companies during their internship search with good intentions, this may turn away many users
- Option 2: No; there will be no limit to the number of applications a user can submit
  - Pros:
    - Users can apply to as many job postings as they wish
  - Cons:
    - Depending on number of users, allowing unlimited applications could potentially take up too much space on the server
    - Allows for spam applications that clog up both the employer applicants listing and our own server space
- Decision: Yes; cap the number of applications a user can submit. However, this cap will be high enough (maybe 40 applications) that most legitimate users will be to apply to all of their desired companies.

How do we verify employer accounts?

- Option 1: No verification; anyone can create an employer account
  - Pros:
    - Easy to implement
    - Employer account creation is fast
  - Cons:
    - Easy for users to masquerade as companies and create fake employer accounts and post fake listings, which can ruin applicant experience
- Option 2: Employers register an account using a company email with a verified domain (e.g. google.com)
  - Pros:
    - Employers can register an account quickly and without hassle
    - Requiring an email with a verified company domain prevents outsiders from posing as the company
  - Cons:
    - Duplicate employer accounts may be created if two people from the same company both register accounts

        - Difficult to implement - requires finding defining some database of verified domains
- Option 3: Cintern admins verify employer emails and send a key 24 hours after employer sign up
    - Pros:
        - Prevents the creation of duplicate employer accounts
        - Further verifies the user's ties to the company s/he is representing
    - Cons:
        - Verifying an account is more time-consuming compared to the scenario where employers make their own accounts
        - Depends heavily upon admins doing their job (aka us)
- Option 4: Cintern admins send out an email with a special sign up keys to employers (not at request)
    - Pros:
        - Allows us to reach out to as many employers while being relatively sure of their identity
        - Easy initial implementation and testing
    - Cons:
        - Email can be interpreted by spam
        - Difficult to determine which emails to send sign up keys to
        - May miss potential employers (which leads us back to Option 3)
        - Requires work from admins to figure out where to send emails to
- Decision: Cintern admins verify employer emails and send a key 24 hours after employer sign up. Although this approach takes longer than the other options that we considered, it does not bar any legitimate employers from creating accounts while still verifying employers and preventing duplicate account creation.

Should we implement job posting deadlines?

- Option 1: Yes; allow employers to set application deadlines for job postings and allow them to edit them
    - Pros:
        - Enforces application deadlines for employers that use them
        - Allows employers to extend deadlines without creating a new listing
    - Cons:
        - Editing deadlines can be complicated to implement (i.e. notifying applicants of the change)
- Option 2: Yes, but don't allow employers to change the deadline after the job posting
    - Pros:
        - Easy to implement, as in applicants cannot submit an application after that application deadline
    - Cons:
        - Employers may want to change their deadline (biggest problem is if an employer wants to move a deadline up)
- Option 3: No; employers cannot set application deadlines for job postings
    - Pros:

        - Keeps the user interface and experience simple
        - Easy to implement
    - Cons:
        - Employers would have to use some other method to inform applicants of application deadlines.
        - This causes work on the employer end, in which they either have to delete their listing or set another status so that they don't get more applications after a certain deadline
- Decision: Yes, but don't allow employers to change the deadline after the job posting. This approach gives employers the option to specify an application deadline, and not allowing deadline changes simplifies both the implementation and the interface.

Should common applications be editable?

- Option 1: Yes; the applicant can edit their common application after submitting it, and we notify relevant employers of the change
    - Pros:
        - If the user makes a mistake, s/he can go back and edit the application to fix the mistake
    - Cons:
        - Can become potentially difficult to implement
        - Applicants could potentially use the notification system to try and force employers to view their application additional times
- Option 2: Yes; the applicant can edit their common application after submitting it, and we do not notify relevant employers of the change
    - Pros:
        - No need for an application-change notification system
        - Provides more flexibility to the applicant
    - Cons:
        - Some applicants may make important changes to their common applications that no employers will end up seeing
        - Could be frustrating for an employer to see an applicant's information change (could make the site seem less credible)
- Option 3: No; the applicant may not resubmit the common application
    - Pros:
        - No need for an application-change notification system
        - The applicant info that an employer sees never changes, so an employer never needs to revisit the applicant page once they've looked at it
    - Cons:
        - If an applicant makes a mistake on the form or some other change occurs (e.g. student changes major), s/he cannot edit the application
- Decision: No; the applicant may not resubmit the application. Although the user will not be able to fix mistakes or change information, this approach makes not only the implementation but also the interface simpler for both users (applicants and employers) to understand.

Implementation Challenges

How do we represent an application and its status (not started, in progress, submitted, rejected)?

- Option 1: Each application object holds references to both its subparts and its status
  - Pros:
    - Given an application, its status can be retrieved by a simple lookup
    - Changing the status of an application is easy
  - Cons:
    - Completion level is only relevant on the applicant side, but employers only care about whether an application is submitted or rejected
- Option 2: Each application object holds references to only its subparts; each user holds references to lists of applications for each level of completion
  - Pros:
    - Retrieving all applications of a certain level of completion is simple
  - Cons:
    - Changing an application's status requires moving the application between lists or deleting the old one and creating a duplicate in the new list
- Decision: Each application object holds references to both its subparts and its status. To check or edit an application status, users just have to complete a simple lookup; employers don't have to look through each of the user's completion-level lists.

How do we represent starred applicants for a particular listing?

- Option 1: Each application object has an attribute that indicates the application is either starred or unstarred
  - Pros:
    - Given an application, its starred status can be retrieved by a simple lookup
    - Changing the starred status of an application is easy
  - Cons:
    - For a given applicant, there is no continuity from one application to another as employers star applications, not applicants
- Option 2: Each listing holds a reference to a list of starred applicants
  - Pros:
    - It is easy to get all starred applicants for a listing
  - Cons:
    - Changing the starred status of an applicant requires moving the applicant between lists or removing from one list to add to another list
    - When an applicant withdraws their application to a listing, it must be dissociated from the list of starred applicants
- Decision: Each application object has an attribute that indicates the application is either starred or unstarred. This approach is simple to implement and avoid some of the complications of application withdrawal.

What happens when an application is withdrawn?

- Option 1: The application is deleted and is no longer visible to either the employer or the applicant
  - Pros:
    - This approach saves storage space if the user is uninterested in reviewing their withdrawn applications
    - Applicants may resubmit an application, so it is somewhat forgiving to the applicant
  - Cons:
    - The applicant cannot review the withdrawn application
    - New application no longer carry over their starred/unstarred status from the previous, this can be confusing on the employer end
- Option 2: The application is not deleted but can only be accessed by the applicant, and the applicant cannot submit again to that application job listing
  - Pros:
    - The applicant can keep track of all of their applications, even withdrawn ones
    - Employers can be sure that there will be no changes to the applications that they currently have. This is easy on the design model
    - There can only be one withdrawn application for job listing, so this saves server space per listing
  - Cons:
    - Applicants cannot allowed to make any mistakes in their applications
    - Applicants must be very sure that they want to withdraw an application, this is very unforgiving
    - Applicants may accidentally withdraw an application and then not be able to resubmit (however, this is a small problem since we have an “Are you sure” prompt)
- Option 3: The application is not deleted but can only be accessed by the applicant, and the applicant can submit to that job listing again
  - Pros:
    - The applicant can keep track of all versions of their applications, so they may look back on what they previously wrote
    - Allows for user error and is very forgiving
  - Cons:
    - May take up server space in keeping track of all the deleted applications
    - May be complicated as job listings will always have to keep track of the most current version
    - Will clog up user dash if they are constantly withdrawing applications and resubmitting them/using this previous applications as commits
    - Confusing on the employer end if users constantly withdraw their applications to make small changes
- Decision: The application is not deleted but can only be accessed by the applicant, and the applicant cannot submit again to that application job listing. This design is similar to

how most job search websites handle withdrawal and will hopefully be intuitive to most users.

What happens to a rejected application?

- Option 1: The application is deleted and is no longer visible to either the employer or the applicant
    - Pros:
        - This approach saves storage space if the user and employer are uninterested in reviewing rejected applications
    - Cons:
        - This approach would require us to implement another feature that notifies the user of a rejected application (instead of just assigning a “rejected” tag to the application in the applicant’s list)
- Option 2: The application can still be accessed by both the employer and the applicant but is marked as rejected
    - Pros:
        - Both the applicant and the employer are able to review rejected applications; this could be useful for application comparison
    - Cons:
        - May hurt the applicant’s feelings :(
        - Clutters employers’ lists of applicants and applicants’ lists of applications
- Option 3: The application can still be accessed by the applicant but not the employer
    - Pros:
        - Since employers are uninterested in rejected applicants, rejected applicants don’t clutter their list of applicants
    - Cons:
        - Employers would be unable to review rejected applicants
- Decision: The application can still be accessed by the applicant but not the employer. This way, applicants can review their rejected applications, but employers can remove unwanted applicants from their list.

Should we support applications and listings that carry over to the next year?

- Option 1: Save all job postings and applications forever
    - Pros:
        - Applicants and employers will always be able to access previous applications and postings
    - Cons:
        - After several years, the accumulation of old postings and applications will take up a significant amount of space in the database
- Option 2: Delete all job postings and applications once per year
    - Pros:
        - Since applicants can’t reuse applications, deleting old applications declutters applicants’ lists of applications

        - Employers’ job openings usually change from year to year as companies grow and reorganize, so stale job openings from previous years will be irrelevant
    - Cons:
        - Students and employers will be unable to review past applications and postings, respectively
- Decision: Delete all job postings and applications once per year. Since Cintern is focused on helping students find summer internships, once companies are done hiring for one summer, all stale applications and postings are deleted before the next hiring season.