Rush Tracker: Team Work Plan

By Random Engineers (random-engineers@mit.edu)

Ben Shaibu
Dennis Smiley
Louis Descioli
Matt Kerr

Stakeholders

Rush Tracker will most directly affect all who are associated with college fraternities. Our target audience is fraternity brothers, who will use the application to keep track of potential rushees for their fraternity. Within the subset of brothers using Rush Tracker, we also intent for a fraternity's Rush Chairman to use our application. A Rush Chairman is a brother who is elected to manage a fraternity's rush. The we envision Rush Chairmen to have Administrator accounts in Rush Tracker.

The rushees themselves will not use our application. The only effect rush tracker has on a rushee is that fraternities he is rushing will be able to know more about him.

In addition to the direct effects on brothers and rushees, our application will have indirect effects on the campus as a whole. One indirect effect might be that there is a smaller fraction of non-affiliated students at schools. There are probably other second-order effects.

Resources

Technologies Required

- Development
    - Rush Tracker will be developed on team member's personal computers using Ruby on Rails, Javascript, Git for version control, Ruby Gems, and a third-party CSS web framework called Bootstrap.
- Production
    - Rush Tracker will be deployed on to Heroku for its MVP and production releases.

Computational, cost and time constraints

- Hosting Static Files / Images
    - Best practice would involve decoupling static files such as Rushee images from our production Heroku server by placing them on an external server such as S3 or CloudFront due to storage constraints. Setting up an external server is not something we will do for the scope of the 6.170 class since we don't want to pay for that. Instead we will host static images on our production server, however this limits our ability to scale. This is not a problem since we do not intend to begin scaling the application within the timeframe of 6.170.
- Postgres Database and Filtering
    - We will be using Postgres for our database. We plan to filter the Rushees by multiple fields and combinations of the fields. If we decide to do search using ActiveRecord, we will need to structure our queries to take advantage of the optimizations in the underlying database.
- Integration with social media
    - Integrating our application with social media applications (for example Facebook) would require additional time cost since no team member is familiar with any external API.
- HTML / CSS views
    - No member of our team is especially knowledgeable of CSS. This will consume time.

Tasks

Due Date	Name	Expected Effort	Assigned
11/6	**Team Pitch**	2 HR	All
11/6	**Teamwork Contract**	1 HR	All
11/10	**Concept Critique**	1.5 HR	All
11/12	**Preliminary Design**	7 HR	All
11/12	**Teamwork Plan**	1.5 HR	All
11/17	**Implement Brother, Admin, Brother Verification MVC**		Dennis
	11/12 - Have Model Done		
	11/14 - Have Controller and Basic View		
	11/16 - Have Finalized View		
11/17	**Implement Rushee MC, Action M**		Matt
	11/12 - Have Action Model Done		
	11/14 - Have Rushee Model Done		
	11/16 - Have Rushee Controller		
11/17	**Implement Rushee V, Join Table M**		Louis
	11/12 - Have Model Done		
	11/14 - Have Controller and Basic View		
	11/16 - Have Finalized View		
11/17	**Implement Presentation Mode**		Ben
	11/15 - Have Basic View and Models Incorporated		
	11/17 - Have Finalized View		
11/17	**MVP Implementation**		All
11/18	**MVP Demo**		All

11/20	**Design Critique**		All
11/24	**Revised Design Document**		All
12/2	**Code Critique**		All
12/8	**Voting**		Louis
12/8	**Reminders**		Dennis
12/8	**Social Media Integration**		Ben
12/8	**Events**		Matt
12/8	***Polish*** *(time permitting)*		Everyone
12/8	***Profile Filtering and Search*** *(time permitting)*		Everyone
12/8	***Mobile Version*** *(time permitting)*		Everyone
12/8	**Final Implementation**		All
12/9	**Final Demo**		

Risks

Implementation Risks

These items are risks because no team member has experience with them:

- Full Screen View/Presentation View
- Mobile View is unknown territory
- Concurrency issues - multiple people editing a profile/voting
- Modal views
- Scaling associations
- Filtering
- Integrating APIs

Design Risks

These items are potential design risks:

- Trust
    - Issues:
        - Brothers may claim to be in a fraternity that they are not
        - A Fraternity may claim to represent a fraternity on a campus when it does not.
    - Mitigations:
        - Administrators must verify new brother accounts before they become activated.
- Users / Brothers maliciously or accidentally modify rushee information
    - Issues:
        - Brothers with strong feelings about a Rushee may attempt to lower or increase their standings by messing with their information.
        - Inexperienced users may delete things.
    - Mitigations:
        - We have mitigated this by not allowing normal brothers to delete rushees.
- Rushees without Social Media accounts
    - Issues:
        - A rushee may not have a social media account
    - Mitigations:
        - We structure our application so that social media integration is not an integral part of our application.

MVP

The main goal of our product is to create a system that makes it easy for Fraternities to keep track of the potential pledges (rushees) that they are interacting with during Rush. One of the key issues with current methods is that after collecting all this information in multiple applications (such as multiple GoogleDocs, phones, and people's memories) there was no easy way to output it in a nice presentation format. Our MVP will test just that idea; getting all the Rushee information and easily generating a slide show.

Subset of features

During our MVP, we will focus on these core features that make up our application:

- Admin Account
- Brother Account
- Rushee Profile
- Presentation Mode
- In the MVP, anyone can edit rushee profiles (including deletion)

Postponed Issues

While these features add to the overall Rush Tracker experience, they are not critical to our MVP. We will be postponing:

- Events
- Action Reminders
- Voting
- Profile Filtering and Search
- Social Media Integration
- The Mobile Version