REPORT.PDF

SKEDG

1. INITIAL STEPS

MILESTONES

These were the milestones in our design doc.

- March 22nd:
  Prototype for website,
  Prototype for scheduling algorithm
- March 25th:
  Prototype for event creation
- March 29th:
  Prototype for registration
  and Google Calendar linking
- April 5th:
  Link event request creation
  with optimal-time-finding algorithm
  (assume accepted request)
- April 9th:
  Link users, event ceration, and Google Calendar; Finalize prototype
- April 10th: Project prototype due
- April 12th: Finalize web user interface
- April 15th: Finalize alpha version
- April 19th - 22nd
  Create mobile apps OR
  continue finalization
- April 22nd (if time)
  Connect iCloud Calendar
- April 24th: Alpha Test
- April 27th: Finalize mobile apps
- April 30th: Finalize for beta test
- May 1st: Beta Test

PARTITION OF TASKS

None of us had any previous experience with building web apps. We split into two teams to partition the work.

Roy and Crystal were on were on Team Frontend (a.k.a. Team Bestend). Their work included:

- Using LayoutIt and Bootstrap to create a prototype of the website
- Creating a flow chart of the website structure
- Creating a basic GUI with filler text and images
- Writing the functions that calculate optimal times)

However, LayoutIt didn't give us the level of customizability that we were looking for. Also, Roy and Crystal had issues with working on the same LayoutIt template at the same time. Additionally, we designed a multi-webpage system, so Bootstrap was not ideal.

So, after setting up a flowchart and design, we decided to focus on functionality rather than interface. Roy wrote the getTimes function, and Crystal used Django to set up functionality.

Jonathan and Roy were in charge of setting up a server on AWS and learning how to format/sync Google Calendar with our website. AWS requires a very specific file structure that didn't play well with Django; Django needs a static file structure, but AWS moved the files around.

Minseung took care of the Google Calendar synchronization; a minor setback was figuring out how to use refresh tokens/authentication tokens and enabling them to survive past the default hour-long expiration date.

Things really picked up the weekend of HackPrinceton, when we met this guy named Lev who taught us how to deploy what we had through Linode.

2. ALPHA TESTING

We got to the alpha testing stage early (April 19th instead of the proposed April 24th date).
We chose an initial alpha base of fifteen friends who we determined would be most likely to schedule events with one another.

RESTRUCTURE OF FRAMEWORK

At this point, the event creator had to manually invite each invitee to the event via autocomplete (analogous to Facebook's friend-finding search bar).

However, this led to some confusion, such as

"When you invite people do you type their real names or username? Can't tell."

– Jeff Wu

In our initial milestone discussion, we didn't anticipate there being many changes between the alpha and beta testing. We planned to make minor changes to the interface and fix possible bugs, but we realized that the creator invitee parameter was a big flaw in our code for the following reasons:

1. Users could only invite other users on the system to their events. (This was actually a big deal to alpha testers. Everyone wanted a friends list to see who was on Skedg, but this added another layer of complexity.)

2. Users could see everyone else who used our website.

3. This added another level of interaction and complexity on the creator side. Additionally, what happens if the event creator invites other users during the initial event creation phase, but realizes later than he/she forgot to include someone?

To solve this problem, we decided to use an event-link system like that of Doodle or WhenIsGood.

After the creator makes an event, he/she is given an event link that he/she can then share with other users. Minseung assigned each event a random string ID, which became the new URL. After the invitees get the link, they can join the event as needed.

Additionally, the creator originally got notified each time an invitee accepted the event, then generated the potential times manually after everyone had accepted. Now, we calculate the potential times each time a new user joins or leaves the event.

Although this is more computationally expensive, we don't expect events that factor in everyone's times to take in a large number people. During the demo, someone asked if our app would slow down if we were to, for example, schedule a 333 class meeting with everyone in our class. However, that kind of scenario wouldn't be realistic because that kind of meeting wouldn't take everyone's times into account. Also, our code has been refactored to be pretty computationally sound (see internals.pdf).

Crystal restructured the Django model framework and views so that invitees were no longer added on creation, and we removed the autocomplete user search function that Jonathan made.
Roy helped fix merge conflicts when we tried committing to the group Github at the same time, and also did a pretty good job debugging issues.

2. ALPHA TESTING (CONTINUED)

INTERFACE

Interface seemed to be very important to our users as well.

"Maybe using geometric shapes (curved corner boxes) might be a better way of partitioning items in space? While I appreciate the "Hello An" as the largest part of the header, it's a relatively unimportant visual that doesn't deserve that much space. I think a logo would be fitting."

– An Chu

"Also, the words are a little hard to read, especially for when you get times. Too jumbled. Organize it better. A color scheme that looks less like a hospital check-in tablet [is needed.]"

– Jeff Wu

Additionally, many of our friends mainly browse websites on their phones, so they were massively unimpressed when they checked out the website on mobile.

To solve this, Jonathan heavily modified an online template (HTML5, Twenty) to get the interface of our current website, and added JavaScript/jQuery functions to toggle overlays, etc. JavaScript/jQuery was also used to sort events into their corresponding divs in the user homepage (past events, pending events, etc.). Also, we added mobile compatibility and narrow screen auto-resizing! Take that, Bootstrap.

ADDITIONAL FEATURES

Crystal started on two small pet projects (notification panel, message board) while she procrastinated making all the new changes to the Django models. The notifications panel initially began as a pop-up telling invitees to an event if a creator had deleted their event, but expanded to cover six cases:

- Invitees are alerted when a creator deletes an event.
- Invitees are alerted when a creator schedules an event.
- Creator is alerted when invitees join an event.
- Creator is alerted when invitees leave an event.
- Creator and invitees and notified when someone posts a message in an event.
- Message poster is alerted when the creator deletes their message in an event.

Messaging also turned out to be a really successful idea, as our testers could post event-specific comments within the scope of the app (such as location of event, reminders, etc.). Jonathan helped with the CSS to make the notification panel and message board attractive, and Crystal embedded the form functionality in the HTML and wrote the Python code.

3. BETA TESTING

Because of the massive design change on both the front-end and back-end, we kicked off beta-testing one day after anticipated (5/2/15). The feedback this time was much more positive regarding interface.

CHANGES

Since the demo, we've made quite a few user-friendly changes. Roy added a "change password" feature to the "My Account" and Minseung added a feature to the events page such that invitees now know which events specifically conflict with their events. (Originally, only the creator could see which events everyone could make. Now, the invitee can see which events he/she can make.) Jonathan has updated the tutorial and Crystal has added a manual "refresh times" functionality on the creator-end, in case calendars have been changed since initial event creation. Also, users can now change which of their Google Calendars they'd like to pull events from.

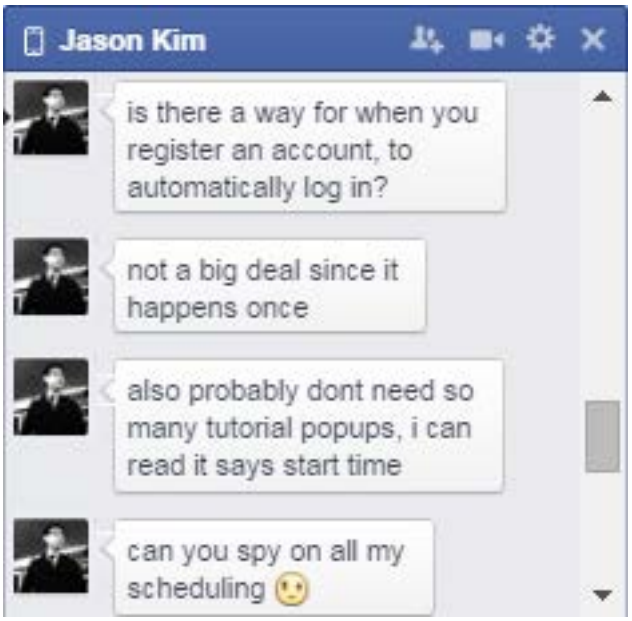
PRIVACY CONCERNS

We've also taken a few precautions for user privacy. I wish we could take credit for encrypting user passwords, but Django automatically does that. So, admins can't see user passwords. Additionally, we add a level of security through e-mail account verification. The randomized eventID as opposed to the default ID provided by the database count reduces the chance of malicious users joining the creator's event (of course, the creator can boot invitees as well), and there's a check in the manageCreator function (in views.py, which handles input from the creator end) to make sure that the user deleting any event is actually the creator (in case anyone wants to sneakily mess with our forms).

4. NEXT STEPS

There are a few features we'd like to implement to further develop our app, though, including:

- A feature to reschedule past events.
- A feature to schedule events at regular occurrences (for example, schedule lunch with Joe once a week).
- Smart adding options: for example, a dropdown that lets you pick specific events such as "lunch" and "dinner" (so the program would find optimal intervals within conventional lunch and dinner times) and also a dropdown that lets you pick times such as "today," "within the week," "over this past weekend," etc.
- Login through Facebook and Google Plus (although who uses Google Plus, really?) for ease of access.
- Sync with iCalendar and Outlook in addition to Google Calendar.
- Priority weights given to certain members of an event. (For example, if Brian Kernighan is a V.I.P. for a COS 333 Meeting, the algorithm would insure that he could make each time.)
- An opt-in e-mail notification system that not only allows you to get the notifications discussed earlier, but also send the event link to specific e-mails to invite them to your event.
- More specific added Google events. Currently, the event added to the Google Calendar only shows the title of the event. We'd like to add an optional location (with Google Maps integration) and include the invitee e-mails in the submitted event.
- Revenue generation so that we can afford matching team t-shirts.*
- Static pages explaining our privacy system to address concerns.



- Jason Kim, voicing privacy concerns

Developing Android/iOS app versions is completely necessary given that we already have a mobile-friendly website.

* TEAM MEETING POLICY

In addition to working on our own time and attending weekly team touch-base meetings internally and with Amy, we met as a group twice a week for three hours each to work on the website. For each minute that a team member was late to the meeting, he/she would have to contribute $1 to the team fund. We planned to use this money to pay for a group meal at the end of the semester, but we were really good about showing up on time, so we may only be able to do group ice cream. (And by that, I mean we could probably buy one ice cream cone and take turns eating it.) It turns out that we needed a lot more time to work on the website than was planned, however, so we also held many weekend meetings.

SKEDG

5. REFLECTIONS

If we had carefully mapped out the workflow and anticipated the alpha testers' distaste for the creator invite system, we could have saved a lot of time rewriting the structure of the website. However, I think this was a really valuable learning experience. In the future, we might target a focus group and see what users would want in terms of functionality before building the website.

Our team dynamic was great. Although we got off to a rough start in that no one had previous experience with developing web apps, we all found specific parts of the project to work on and played an equal role in Skedg's success.

CREDITS

Credits go to Amy Tai for being an awesome TA and to Brian Kernighan. Also, we outsourced logo creation to Amy Xiong, and got some great alpha/beta testing feedback, particularly from

- Michelangelo X Ball Van Zee
- An Chu
- Annie Chu
- Artur Filipowitz
- Kristin Gjika
- Chris Habermann
- Alana Jaskir
- David Mazumder
- Laura Srivichitranond
- Jason Kim
- Jeff Wu

And many, many others. Also, shout-out to Princeton Engineering for featuring us on their Facebook page.

SKEDG