Selfie with Strangers

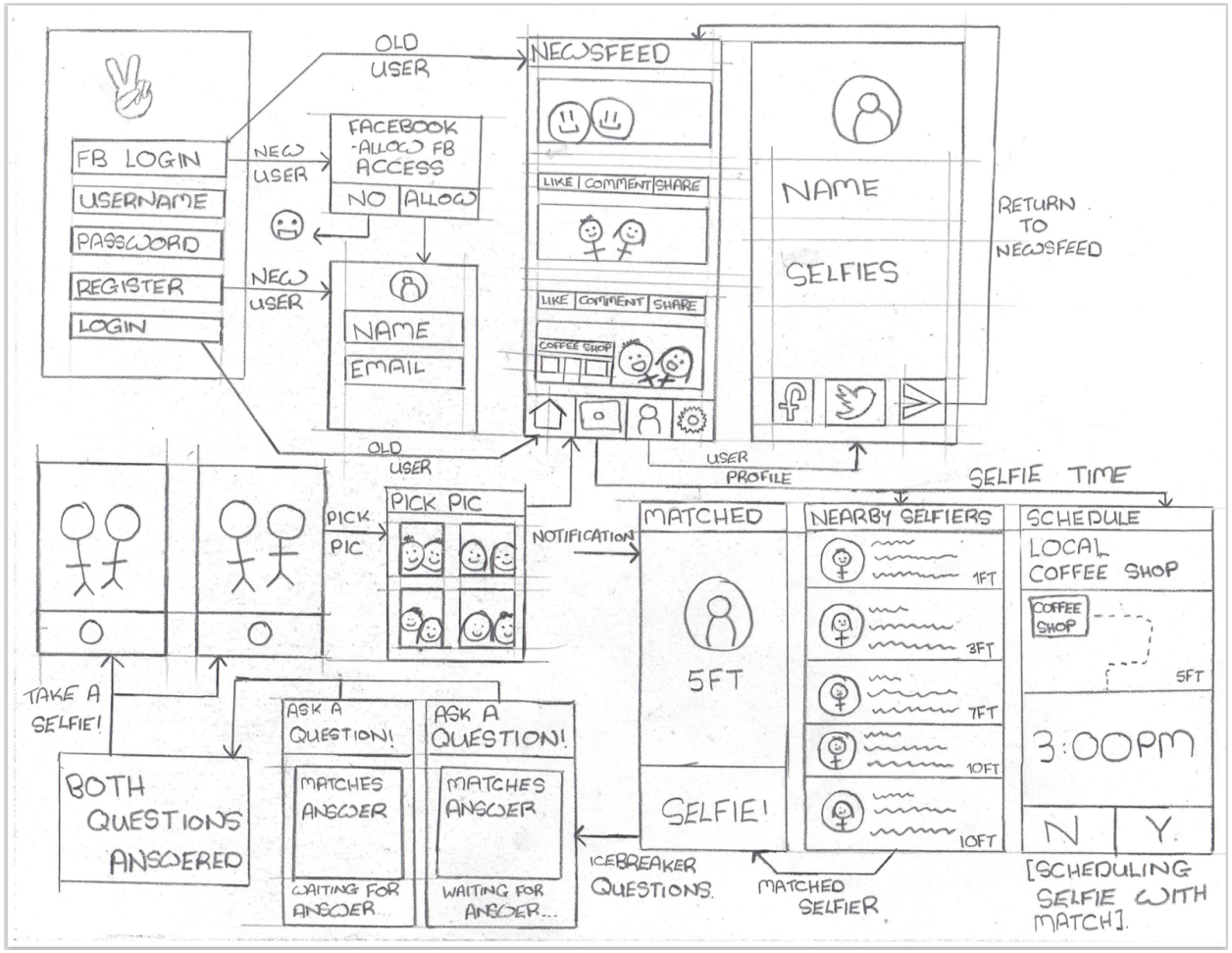
Design Specification

9 April 2015

Summary of the Project Idea

Selfie with Strangers is a low-pressure, low-risk way to meet people around you. Users find other users by being notified when they are close to each other, or by scheduling a meeting for a future time. Once users meet, they tell the other person about themselves and take an awesome selfie together.

Detailed Application Flow



Login:

- Receive information from form and query facebook for authorization
- If need to: Create a new user and save the user to the database
- Loads user session
- User is taken to Newsfeed

General Non-Login Views:

- Should have at least return to Newsfeed button

Newsfeed:

- Maintains a list of X most recent selfies taken in a region (or hold selfies from the previous week/day?)
- Can receive new Selfie to add to list
- Has buttons for View Profile, Search for Selfies, and Settings

User Profile:

- User is able to view all his/her selfies
- Contains basic user information

Search For Selfie:

- Gets location of user
- User gets option of taking a selfie with someone nearby, or someone at a future time and undetermined (but relatively nearby) location
    - Switch between Nearby or Future Views by swiping?
- Nearby:
    - App displays location and image of a nearby user.
    - Displays red border if that user is the one for Future Selfie ???
- Future:
    - Displays location near two people for Selfie and a time
    - Users must both agree for it to be scheduled
    - Decline by clicking NO or swiping?
- Receive decision and then either initiate nearby or future scheduling

Taking Selfie:

- Answer Question:
    - Display question from database to each user to answer
    - Save answer to database
- When Users Meet:

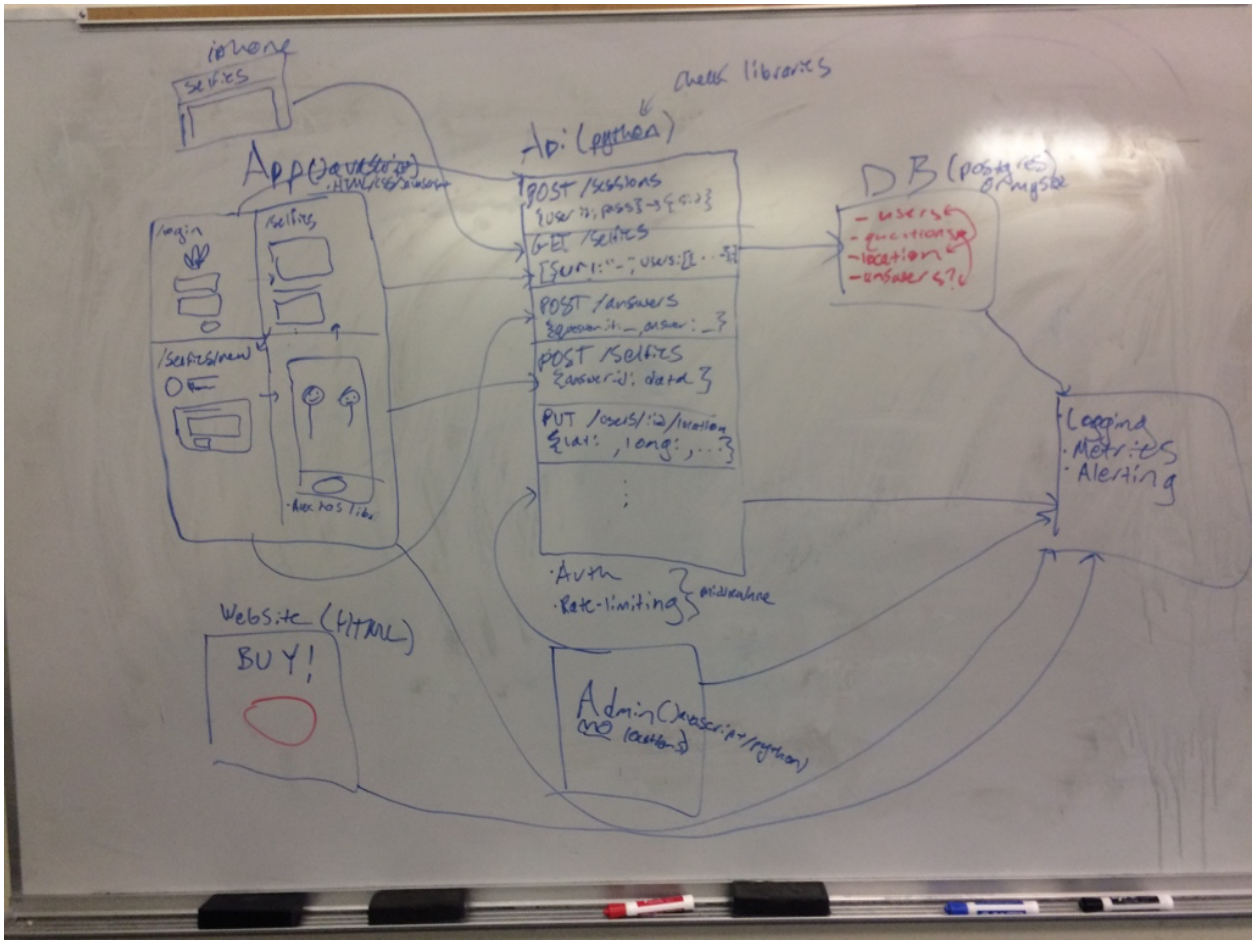
    - Generate meeting code for each user
    - Each user will be required to enter the code that the other person has, in order to verify that they have met the correct person
- Take Picture (s):
    - Pictures taken by either user are temporarily stored
- Pick Picture:
    - Display pictures taken by both users
    - If both users allow pictures to be posted to newsfeed, then the selected pictures is posted to the newsfeed
    - The answer to the question is associated with the picture, as well as an optional caption

Settings:

- User is able to update user information and update security preferences
- User is able to specify whether or not selfies will be posted to the newsfeed

Aleksandr Burkatovskiy - 3/27/15 and Caleb Larson - 3/29/15

Architecture Diagram



Bruce Spang - 4/9/2015

Architecture Description

The Selfie with Strangers stack is roughly structured into two main services: an app which handles the UI, an api which handles the business logic and persistence.

The app service is responsible for making api calls to the api service to collect data, and then using that data to render the user interface. The app service is written in node.js using express. The app service has no persistence of its own, and can only make requests to the api service and send html to the browser.

The api service is a JSON/REST api which is responsible for all Selfie with Strangers business logic and data persistence. The api will have endpoints for each required object and behavior for Selfie with Strangers. In addition to having endpoints for all the necessary Selfie with Strangers

functionality, the api will also handle things like user sessions, user permission checking, and rate limiting

To understand this architecture a little better, let's examine how a selfie would be uploaded and included in the newsfeed First, a user would take a selfie using the app, and pick the one they wanted to upload. Once they picked the selfie, their browser would upload the selfie to the app's `POST /selfies` endpoint, which in turn would upload it to the api's `POST /selfies` endpoint, which would create the selfie and store it in the database with any associated metadata. When a user wanted to view the newsfeed, their browser would make a request to the app's `GET /selfies` endpoint, which would make a request to the api's `GET /selfies` endpoint, which would return a json-encoded list of selfie objects. The app would then take this list of objects and use them to render the html for the newsfeed.

Bruce Spang - 4/20/2015

Software Components

(include the relevant libraries for each component description)

Login/Users component

- Facebook api/Javascript SDK to log-in with facebook
- Use oauthlib for authentication with the api
- Passport.js for authentication in the app
- Must use bcrypt for storing password hashes

Newsfeed

- The newsfeed will be a list of selfies from all users. This can be easily implemented with flask/express.js

Search for Selfies

- HTML5 geolocation library: http://www.w3schools.com/html/html5_geolocation.asp
- Geolocation via ip address: urlopen from urllib, this gives us the options of using something like http://freegeoip.net/json/<IP> to get a JSON object with the geolocation data of that IP
    - Example use here: http://stackoverflow.com/revisions/26165487/1
- Making a unique ID for this match in python (this makes it easy to assign a question to this match and for finding pictures)
    - UUID: https://docs.python.org/2/library/uuid.html

Taking a Selfie

- getUserMedia API for taking a picture
    - http://davidwalsh.name/browser-camera
- express has an api for uploading files
    - http://howtonode.org/really-simple-file-uploads

- Flask has an API for uploading files
    - Found here: http://flask.pocoo.org/docs/0.10/patterns/fileuploads/
- Upload to Facebook
    - Facebook Javascript SDK for posting a link: https://developers.facebook.com/docs/javascript/quickstart/v2.3#dialogs
    - Facebook Javascript SDK for actually uploading it to Facebook https://developers.facebook.com/docs/javascript/quickstart/v2.3#graphapi
    - Worth looking into if we are truly dedicated to Python:
        - https://github.com/pythonforfacebook/facebook-sdk
        - http://facebook-sdk.readthedocs.org/en/latest/api.html

Machine Learning

- Scikit-Learn/Scipy/Numpy - These libraries will be used to build statistical models for matching users to each other, suggesting ice-breaker questions, and possibly for suggesting times and locations.

Ops Details

- Vagrant for the dev environment
- Ansible for provisioning/deployment
- DigitalOcean for hosting
- Travis for CI
- Scales/graphite/nagios for metrics

Zac May - 3/31/15, Oskar Singer - 4/9/15, Wesley Fung - 4/1/15, Bruce Spang - 4/10/15

Database Schema

Please see the initial schema migration in *src/api/migrations/versions/32d34fe1baa0_.py*

Aleksandr Burkatovskiy - 3/27/15, Oskar Singer - 3/30/15, Bruce Spang - 3/30/15

Revision History

- Added high-level architecture summary - 4/20/15
- Initial Submission - 4/10/15