Senior Project

Steps

User creates account (to save resume and job searches)

Users upload resume / Companies upload job descriptions

Parse resume for keywords / parse job descriptions for keywords

Compare keywords

Return matching results to user

//

Allow users to apply (or just view) jobs.

After applying (or viewing) tell the user why he/she isn’t qualified for the position

Questions

Job comparison comes from local website? User entered website? Or the most popular job websites (i.e. monster.com, craigslist, etc).

Additional work

--Mobile application

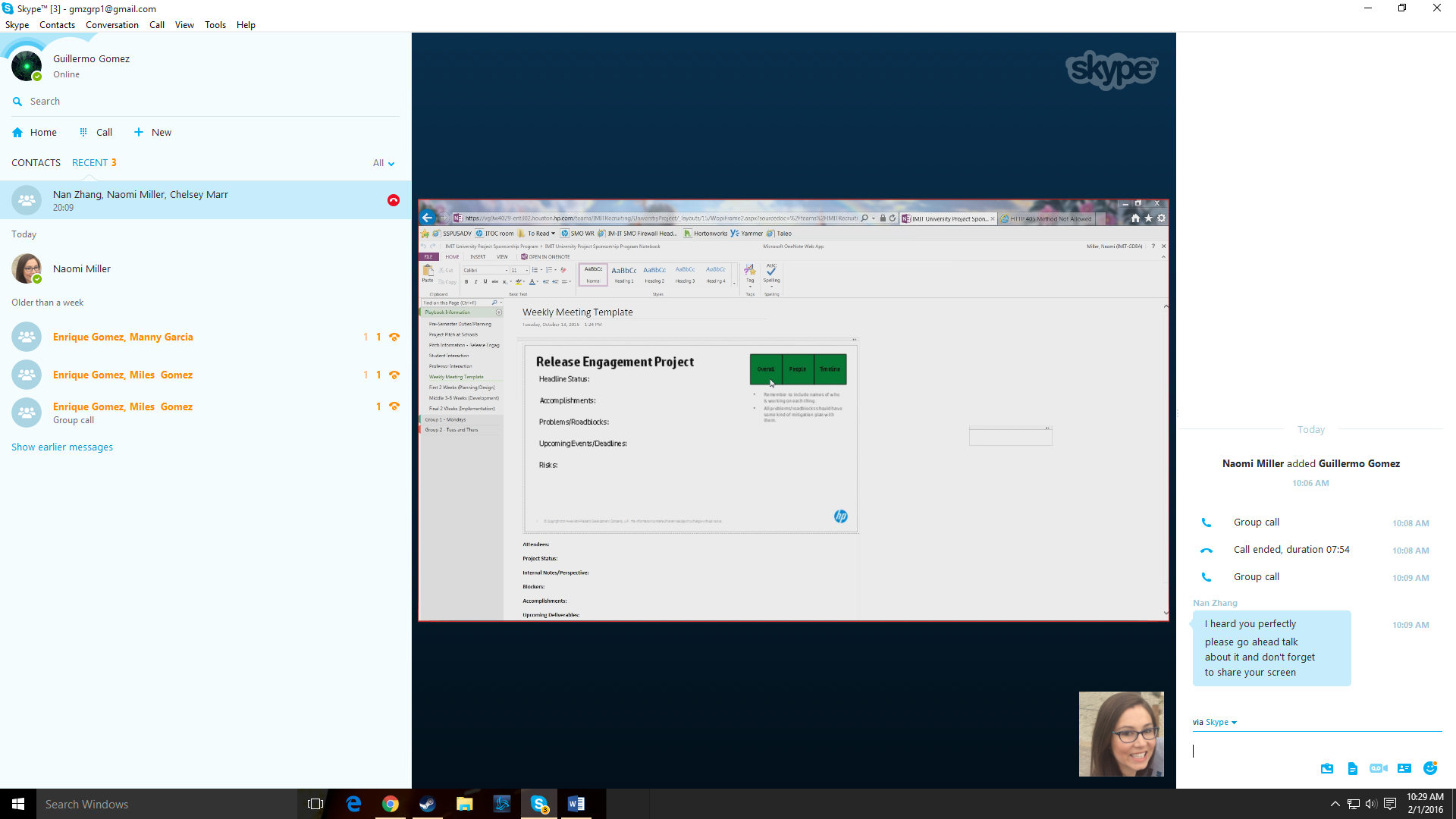
--Job data gathering from other websites. (pluggin maybe)

HP meeting

Parsing algorithm main component of the project

* Employer side
* Master login for each aspect of a company

Agile development time sheet.



Red - bad

Yellow - behind

Green – good

Status slide

1 at each meeting

Friday @ 9 a.m.

Proposal, sprints, status slide

2/5/2016

* Break up Grantt chart more
* User stories
* 2 week sprints
* HPE haven
* Gems for parsing word/paragraphs
* HPE haven for reading files
* https://www.havenondemand.com/

## ****Ruby****

* [Video](https://drive.google.com/open?id=0BwEWIi69phSsc0tNVXQ5cHFvMG8)
* [Github](https://github.com/HPE-Haven-OnDemand/up-and-running-with-hod/blob/master/index.rb)
* First what you want to do is open up your terminal, create a new directory, and 'cd' into it. We’ll call it ‘example\_app’.
* mkdir example\_app  
  cd example\_app
* Next thing to do is look at the gem on [Github](https://github.com/HPE-Haven-OnDemand/havenondemand-ruby" \t "_blank). Then, install the Haven OnDemand gem by running
* gem install havenondemand
* Next, create an index.rb file
* touch index.rb
* and now you can begin coding! Open up the index.rb file and require the gem and initialize it
* require 'havenondemand'
* client = HODClient.new('API\_KEY', 'v1')
* where you can find your API key [here](https://www.havenondemand.com/account/api-keys.html); the second parameter specifies the API version and is optional - defaults to 'v1'.
* Then, let’s create some text and store it in a hash which will be POSTed to the [Sentiment Analysis API](https://www.havenondemand.com/developer/apis/analyzesentiment#overview). Note - the gem accepts the parameters as a key value pair in a hash.
* text = "I love puppies"
* data = {text: text}
* Next, let’s call the [Sentiment Analysis API](https://www.havenondemand.com/developer/apis/analyzesentiment#overview) and log the text, sentiment, and score to the console
* r = client.post('analyzesentiment', data)
* sentiment = r.response.body["aggregate"]["sentiment"]
* score = r.response.body["aggregate"]["score"]
* puts "#{text} | #{sentiment} | #{score}"
* Now that we have everything set, let’s go back to our terminal and run the code
* ruby index.rb
* The text that was POSTed to the [Sentiment Analysis API](https://www.havenondemand.com/developer/apis/analyzesentiment#overview), the sentiment of it, and its score will be logged to the terminal
* I love puppies | positive | 0.805340693105

2/9/2016

-singular resume

-employer side can view multiple resume.

-employer side, matching resumes on completeness top 5.

- scale for users, i.e. 8/10, 4/10.

- matches calculated in real time? User side? File stored on server?

- file storage tree.