**HackOHIO Team**

- November 19-20, starting at Union

- Stephen Wu, Jacob Shoaf, Stephen Pioro, Ishan Taparia

**Table of Contents**

[**Brainstorming**](#_xf9x1l2ht9d6)[**1**](#_xf9x1l2ht9d6)

[**Setup**](#_ry5lu1x52li9)[**8**](#_ry5lu1x52li9)

[**Plan**](#_w3vcwqwic4h9)[**9**](#_w3vcwqwic4h9)

[**Notes**](#_coh0vlrdkojo)[**10**](#_coh0vlrdkojo)

|  |
| --- |
| **Brainstorming** |
| **Events:**  **Web Workshop with Rockwell Automation  November 5 @ 11:15am-3pm** [**https://d1b10bmlvqabco.cloudfront.net/attach/is23vt5ua4m5zs/gslziq6grh5/iuty3ypapnyb/Rockwell\_Flyer.pdf**](https://d1b10bmlvqabco.cloudfront.net/attach/is23vt5ua4m5zs/gslziq6grh5/iuty3ypapnyb/Rockwell_Flyer.pdf)  **Wikipedia in Your Next Project**  **November 17 @ 7:00 PM // Caldwell Labs 120**  [**Wikipedia Connection**](https://en.wikipedia.org/wiki/Wikipedia:Wikipedia_Connection) **and** [**Open Source Club**](https://opensource.cse.ohio-state.edu/) **present on using Wikipedia's API, machine learning, Wikidata, and more in open source projects.** |
| **Stephen:**  **Main resources**   * [HackOHIO](http://hack.osu.edu/2016/#events) events - Attend a couple of these when able * Consider the [sponsoring companies](http://hack.osu.edu/2016/#sponsors) and their technologies; they submit challenges with additional prizes. We may attempt these challenges as part of our product or they may dictate what the product becomes. * Watch: [Facebook Live Hackathon 2016](https://www.facebook.com/zuck/videos/10103008230962831/). Note how some of these products can be really simple (like Facebook Pet pages and Messenger polls), economical (hardware to instantaneously set up Facebook live), to not really exciting (2d jumping game). * [Boilermake submissions](https://boilermake.devpost.com/submissions): This is Purdue’s hackathon. Here’s [2014’s set](https://boilermake2014.devpost.com/submissions).  [Here’s a ton more projects.](https://devpost.com/software?ref_content=hackathon-projects-link&ref_feature=projects&ref_medium=link) * [HackOhio 2015 winners](http://hack.osu.edu/2015/winners/), [2015 live page](http://hack.osu.edu/2015/live/)   + Super important to look at. The winners aren’t all innovative or super technically challenging, but they were probably well designed and well presented.   **Examples of interesting tech ideas**  So these are perfect for Hackathons because they’re not particularly difficult to implement, but they serve a unique purpose, have an impact, or are just pretty cool. Particularly, they don’t require significant infrastructure other than web servers.   * Youtube Instant: <http://feross.org/youtube-instant-media-frenzy/> * ForestApp: <http://www.forestapp.cc/> * Wunderlist: <http://wunderlist.com> * Pushbullet: <http://pushbullet.com> * Slack: <http://slack.com>   **Other tech ideas**   * These aren’t viable for us to implement, because they require more infrastructure, but they’re both well designed and address unique problems * KhanAcademy: <http://www.khanacademy.org/> * Quora: <https://www.quora.com/>   **Tools**   * Planning: consider looking at tools like Sketch (Macs): <https://www.sketchapp.com/features/>   + Combined with InvisionApp: <http://invisionapp.com> * Version Control: [GitHub](https://github.com/) and [Git Desktop](https://desktop.github.com/)   + One thing we have to decide is if we want to open-source the product. We can open-source the hackathon version and have the extended version be private (assuming we extend it later on). My preference is open-source, especially since it helps with job applications, but there are reasons to not do that. * Ohio State APIs; <https://library.osu.edu/blogs/it/webservices-everywhere/>   + We could make something OSU-related but only if it’s really cool. Dining was underwhelming and we didn’t have the manpower to turn it into something cool. A lot of people end up doing OSU related stuff. * Google APIs   + <https://developers.google.com/>   + <https://developers.google.com/maps/>   + <https://developers.google.com/youtube/>   **Languages and frameworks**   * NodeJS webapps: <https://nodejs.org/en/>   + Node is exploding in tech, find out why and whether we should use   + We would also use Heroku to deploy the app: <https://www.heroku.com/>   + MEAN stack     - <http://mean.io/>     - <https://www.quora.com/What-are-the-pros-and-cons-of-MEAN-javascript-stack-vs-LAMP-stack>   + Compare Node to LAMP/WAMP stack:     - <https://www.wikiwand.com/en/LAMP_(software_bundle)#/WAMP>     - I am really familiar with WAMP, but I’m down to try some new stuff     - MySQL v. MongoDB       * <https://www.quora.com/How-does-MongoDB-compare-to-MySQL>       * <https://www.npmjs.com/package/mssql>       * Both are database solutions; we will likely use one or the other * MySQL   + Database solution we will probably use unless Mongo makes more sense   + <http://leshazlewood.com/software-engineering/sql-style-guide/> * ReactJS   + <https://facebook.github.io/react/> * AngularJS   + <https://angularjs.org/> * Bootstrap   + <https://getbootstrap.com/>   + We will almost definitely use Bootstrap in a webapp; it’s really easily deployed with limitations but those don’t matter for a hackathon   + Ex: <http://wustep.me> <http://wustep.me/dine> <https://startbootstrap.com/> * Slack   + <https://slack.com/>   + Both as a source of inspiration and for communication * Spring MVC (Java)   + <https://spring.io/>   + Our coursework focuses on Java and we’ve worked with MVCs. Spring may be a viable option, but webapps are easier to deploy. * jQuery / jQueryUI   + <https://jquery.com/>   + <https://jqueryui.com/>   + I am personally really familiar with jQuery, but we can definitely try to work with Angular/Node   + In the webapp world, pretty much everything uses jQuery in some way unless it’s not javascript-based * [D3.js](https://d3js.org/)   **Additional concepts and resources**   * Markov Chains: <https://www.wikiwand.com/en/Markov_chain#/Applications>   + See [/r/SubredditSimulator](https://www.reddit.com/r/SubredditSimulator/comments/3g9ioz/what_is_rsubredditsimulator/), really interesting stuff * Algorithm ideas: [Quora post](https://www.quora.com/What-are-the-top-10-algorithms-every-software-engineer-should-know-by-heart/answer/Adeel-Ahmed-41?srid=Ivr5) * Machine Learning   + Example: [League of Data](https://devpost.com/software/league-of-data) * Codrops: <http://tympanus.net/codrops/> * D3.js: <https://d3js.org/>   **Ideas**   * Machine learning webapp that helps user learn about scholarships, volunteering opportunities, and events on campus, concerts, lectures, etc., offers based on major, sports, interests, tags and learns what user likes more or less   + Like [SongKick](https://www.songkick.com/metro_areas/9480-us-columbus) or [Eventful](https://itunes.apple.com/app/eventful/id284146702?mt=8) or [Eventbrite](https://www.eventbrite.com/) but personalized and geared towards academia: students, staff, and faculty, and campus events.     - Eventful is really far too commercialized (based on buying tickets) and based on entertainment. Our goal is to hit the audience of students and faculty particularly, and address issues in having way too many calendars or email lists for campus events.       * Another barrier of entry of Eventful is that you can’t personalize it without an account, and the level of personalization is lacking.   + Interacts with TeamUp, Google Calendar, and other calendars, that we can add to list that helps add sports, OUAB, and other offerings.     - Scraping calendars, converting those events into the database, is a key element of the project     - Administrators could approve events to be seen, or modify them in a queue. Ambassadors would do this for different campuses and different majors and interests.       * Create algorithm to block duplicates or very heavily similar events       * Quietly keep track of “quality” of events, but allow user to rate events as well that helps determine what events and source of events is best. For example, OUAB events are highly rated (stuff like a performer visits campus), but repeated OUAB events might be reduced (stuff like Yoga)     - Perhaps, given a page like this: <https://discovery.osu.edu/tda/data-analytics-month/>, learn how to parse it or just manually label spans to parse it     - Entertainment: parse EventBrite or Ticketmaster       * <https://developer.eventbrite.com/>     - Additionally, add functionality to export event, as text message or G Calendar, etc.     - Additionally, add things like scholarships and volunteering opportunities that do not necessarily have dates.   + User can choose view a personalized event schedule   + See: <https://github.com/wustep/stemee> and <https://teamup.com/kse89a84dcb543ed5e>     - So my goal is like the calendar above, but personalized heavily and in a feed format instead of calendar. Several filters and a search bar would be added so the user can better personalize or search for events   + Expansion: Google Maps, determine distances <https://developers.google.com/maps/documentation/distance-matrix/intro>     - This would be used to determine that an event is in distance of OSU or the campus in question   + Summarize information, but keep long text as well to be viewed     - [SMMRY](http://smmry.com/) and [autotldr](https://np.reddit.com/r/autotldr/comments/31b9fm/faq_autotldr_bot/) bot     - Allow users to vote up or down summaries, if downvoted enough, the summary is no longer used   + Maybe even attempt to tag events and categorize them in a machine-learning way, with training sets and whatnot     - Create hierarchies of categories and tags, and create relevant tags with weights, use machine learning to expand upon those categories     - Event, automatically classify tags  Example, given the description, “Programming, HTML, CSS, PHP, JavaScript, etc.” automatically learn to label it programming, cse. Use machine learning to further improve it.        * Example of weight usage and relatives:  Defined in system as:  Field: STEM > Engineering > CSE Relative of ECE, weight: 80% Relative of Data Analytics, weight: 60% Relative of Business Analytics, weight: 30% Relative of Actuarial Science, weight: 10%  Interests: Programming Relative of Computers: 80%  Maybe as a tree if not a bunch of tag relation rows (D3.js)       * So an event might be classified “CSE, CIS, Computers” and weights would be averaged in a certain way to determine if a user would be interested.         + So if I were a Data Analytics student interested in Computers, the events that would show up would be:   Data Analytics & Computers events Data Analytics events Computers events Fields related to Data Analytics: ECE, CSE, etc. events that pass a certain threshold & Fields related to Computers events that pass a certain threshold Campus events (Athletic and OUAB)     - **Research ideas: How often do people attend lectures related to their field? What if they’re faculty or staff compared to a student? What kind of lectures do they attend? Are those lectures generally from a club they often visit or various clubs or non-club events? How does that compare to OUAB attendance? How do people find out about OUAB and other on-campus events? (Facebook/Twitter/etc.?) What kinds of events do they attend?   Do people actually know of when the other free sporting events are on campus? If so, from what resource? Would they be more likely to attend if there was an easy way to find out?  Have they ever hosted or presented a lecture open to the public? If so, how do they advertise it? Do students follow many mailing lists? How many? Do they generally read them and decide to or not to go to events? Which ones? (SWE, Running Club, etc.) Do students think it’s annoying to have to follow *many* mailing lists? Are there any mailing lists students find particularly useful?  Do any students follow calendars of lots of OSU-related events? Do students use EventBrite, TicketMaster, Eventful, Spotify, etc. to find events on or around campus? (Get a list of competitors and compare them).**   + [Tags: tag\_id | tag\_name | tag\_type ] [Tag\_Type\_Names : tag\_type\_id | tag\_type\_name ] [Tag\_Relations: tag\_relation\_id | tag\_1 | tag\_2 | weight | parent ]  [Events: event\_id | event\_name | event\_who | event\_where | event\_when | event\_desc | event\_link | stars ] [Event\_Tags : event\_id | tag\_id ]  [Users: user\_id | email | password | name ]  [User\_Tag\_Preferences: user\_id | tag\_id | weight ] [User\_Starred\_Events | user\_id | event\_id ]   + Technical challenges include:     - Parsing different date formats into a single one       * Easy for calendars like Google or TeamUp, but harder for text       * Need some library to parse many formats of dates and date ranges     - Personalization algorithim * Webapp that helps you plan your majors and minors out and decide what classes to take to fulfill requirements   + Essentially a universal advising tool for colleges     - This isn’t super flashy, but if implemented well, it could be really useful   + Parse course catalog and GE lists to create database   + Given: Groups of requirements, course catalog, and pre-reqs, Generate a pre-formed draggable base schedule with options to add elective choices, etc.     - Can create and save requirements for different majors     - Like this: <https://thomascarlin.github.io/CSEBingosheet/> but way better   + jQuery, KMData, D3.js   + Create trees of course reqs, can search by course and see its pre-reqs and what it links to     - Parse pages like this: <https://kmdata.osu.edu/api/offerings/90949/api>     - Potentially link to degree audit     - Historically also see (what terms) the course was offered   + We would need to email the person in charge of the course bulletin ahead of time to get this all into a more easier readable format |

|  |
| --- |
| **Setup** |
| **Tools**  We are for sure using:   * **GitHub, GitHub Desktop**   + [**http://github.com**](http://github.com)   + [**https://desktop.github.com/**](https://desktop.github.com/)   + **Get a GitHub account, download GitHub desktop** * **Slack**   + [**https://ohioh.slack.com/messages/general/**](https://ohioh.slack.com/messages/general/)   + **Windows:** [**https://slack.com/downloads/windows**](https://slack.com/downloads/windows)   + **Mac:** [**https://slack.com/downloads/mac**](https://slack.com/downloads/mac) * **Wunderlist**   + [**http://wunderlist.com**](http://wunderlist.com) * Google Drive   + For planning and documents * Namecheap server hosting + [Heroku](http://herokuapp.com)   + To have a live product that we can actually put on our resumes. * MySQL (MariaDB)   + <https://mariadb.com/>     - **Windows client:** [**Mysql Workbench**](https://dev.mysql.com/downloads/workbench/)     - **Windows:**       * **npm install --global --production windows-build-tools**       * **in Administrative CMD** * **NodeJS**   + [**http://www.npmjs.com**](http://www.npmjs.com)   + Watch: <https://www.pluralsight.com/courses/node-intro>   + <https://docs.npmjs.com/files/package.json>   + Packages     - <https://www.npmjs.com/package/http-server>     - React     - <http://docs.sequelizejs.com/en/v3/docs/getting-started/> * React   + <https://github.com/enaqx/awesome-react>   + <https://github.com/facebookincubator/create-react-app/blob/master/packages/react-scripts/template/README.md#table-of-contents>   + <https://facebook.github.io/react/docs/installation.html> * Material-UI   + <http://www.material-ui.com/#/>     - <https://material.io/icons/> or maybe <http://fontawesome.io/#icons-new>   Post-hackathon   * LESS or SASS   + <http://sass-lang.com/>   + Adding this might be a little overkill for a hackthon, especially if we use Bootstrap, so probably not   + Will definitely be useful later on. * Spring MVC Framework?   + <http://spring.io>   **Read**  Programming   * <http://udacity.com> * [**https://www.smashingmagazine.com/2016/03/server-side-rendering-react-node-express/**](https://www.smashingmagazine.com/2016/03/server-side-rendering-react-node-express/)   React   * <http://reactfordesigners.com/labs/reactjs-introduction-for-people-who-know-just-enough-jquery-to-get-by/> (for Stephen mainly) |

|  |
| --- |
| **Plan** |
| **Tech** NodeJS v6.9.1  "express": "^4.14.0",  "material-ui": "^0.16.4",  "mysql": "^2.12.0",  "react": "^15.4.0",  "react-dom": "^15.4.0",  "sequelize": "^3.26.0"  **Stephen W** SQL structure Database populator, remove duplicates Approval process Get events method & algorithm Tag relations  **Ishan** Front-end: sign-in page Events page Material-UI integration and Font-awesome  **Ishan + Jacob** Figure out promises to fetch events from database without refreshing  **Jacob** Interest control panel / React component Database populator for emails Tag relations, interests  **Stephen P** Wireframe mockups Presentation & 1-minute pitch Addressing questions and tech Future expansion Market research What exists? How can we improve on it? How is our product special? Answer questions above in bold, through market research or searching  **Way later** Scholarships / fellowships |

|  |
| --- |
| **Notes** |
| SQL Structure   * events   + event\_id (primary\_key, bigint)   + name   + location   + description   + retrieved     - where the event was retrieved from (original calendar link or whatever)   + time\_start   + time\_end   + importance     - 5 stars = really important event, show to pretty much anyone who doesn’t hate the tags     - 1 star = less important, show to people who are really interested in that tag   + approved (int)     - 0 = not approved, 10 = approved * events\_tags   + event\_id   + tag * tags   + tag\_id (primary\_key, bigint)   + name * tags\_relations   + parent   + child   + weight   + (This is the list of all parents). * users   + user\_id   + ~~template (boolean)~~   + ~~template\_name\_1~~   + ~~template\_name\_2~~   + email   + password   + access     - (This is really for future ideas, but can be discussed in presentation)       * 0 for regular user, 10 for email confirmed, 50 for regular user/submitter, 100 for ambassador, 500 for administrator for now.       * 10 - once confirmed, user can vote up and down events and report them       * 50 - users must get approved to submit events or feedback       * 100 - ambassadors, in the future, can approve events but another ambassadors has to also approve it       * 500 - administrators can immediately approve, delete events, modify weights, etc. * users\_tag\_preferences   + tag\_id   + user\_id   + weight     - 0 = don’t want to see anything here     - 1 = default     - 2 = starred * users\_events\_votes   + tag\_id   + user\_id   + vote * users\_events\_reports   Tags   * **Academic [.25]**   + **STEM [.50]**     - **Engineering [.75]**       * **Computer Science**       * **Mechanical**     - **Math**     - **Medical**     - **Science [.75]**       * Chemistry       * Physics       * Other   + **Art [.50]**     - Graphic Design     - Fine Art   + **Business** * **Athletics [.60]**   + **Sporting Events [.75]**     - **Football**     - Soccer     - Volleyball   + Intramural   + Club   + On-Campus Athletics     - (Stuff like Yoga, Nike stuff, etc.) * **Social [.5]**   + **Greek Life**   + **Movies**   + **Music [.7]**     - Alternative Rock     - Classical     - Comedy     - Country and Folk     - Dance/Electronic     - Festivals     - Hard Rock/Metal     - Jazz and Blues     - Latin     - New Age and Spiritual     - R&B/Urban Soul     - Rap and Hip-Hop     - Rock and Pop     - World Music     - (From ticketmaster)     - Other   + **Other** * Scholarships / Fellowships [0]   + Undergraduate [0]     - 1;st year     - 2nd year     - 3rd year     - 4th+ year   + Graduate [0]     - Masters     - PhD * **User [0]**   + **Undergraduate Student**   + **Graduate Student**   + **Staff / Faculty**   + **Other**   + (By default, select all of these tags for events) |
| Misc   * We aren’t encrypting passwords for now, will do later * Let the user pick X for do not want these events, ? for maybe, V (check) for yes * When events are classified, we want them to be classified with the deepest tags (leafs) possible. These are the only things in the database, essentially.   + Let’s say we have a CSE event. We check the CS box, and Engineering, STEM, and Academic are all classified as well.   + In the database, we only need CS checked though? * Things to scrape to start:   + Teamup   + OUAB   + Google Calendars   Default weights   * I am a [undergraduate student / graduate student / other / staff or faculty] near [Columbus, Ohio] interested in the field of [<Academics > Academic Tree OR NONE>]   + Field     - If entire academic tree selected, V all of those academics     - If none academic selected, X all of those academics     - If STEM selected, V all the leafs of STEM   + Type of person     - If undergraduate student, V all of Undergraduate (1st, 2nd, 3rd, 4th years), X all of graduate scholarships       * V everything in social but Music, V everything in Athletics     - If graduate, V all of graduate (Masters, PhD), V all of undergraduate     - If other or staff/faculty, X all of undergraduate and all of graduate     - V the correct user type tag   + ? for every other tag   Algorithm   * v1: find everything whose tags were selected by the user   + Join the tag preferences table and the user preference. If V, show event. * v2:   + Let S = (# of stars on event / 4 + .20)     - 1 star = .55       * Stuff like Yoga or very specific lectures, only show to people really interested in subject matter     - 2 star = .70       * More specific lectures but good speaker, enough for very similar fields, smaller concerts     - 3 star = .95       * Regular lectures, Schottenstein concerts     - 4 star = 1.20       * Stuff like bigger lectures like Science Sundays, bigger on-campus events, big sporting events     - 5 star = 1.45       * Reserved for famous people, campus concerts that are free, hackathons, football games, etc.   + Join all the tags of the event with the tags of the user   + If any tags are at 0%, prevent the event from showing up   + If any tags are at 100%, show the event for sure, unless we have a relationship of 0 (see below)   + Otherwise, if the event’s tags are all ?s     - 1. Find all the user’s tags that are checks, join that list with the tag relations of those.     - 2. Join the tags of the event’s relations with those tag relations on parent tags.     - 2.5. If any of these relations are 0, then don’t show the event and end here.     - 3. Multiply S by the highest weight of that table, or 1 if there is no relations.     - 4. If we have a score of atleast 72, show the event.     - Academic example:       * So then we have a table looking like         + Chem | Science | Physics | .75 | .75         + Chem | STEM | Physics | .50 | .50         + Chem | Academics | Physics | .25 | .25         + Multiply S by the highest weight from that table.         + So tags with shared parents (.75) would be shown if it’s at least 3 stars         + Tags with shared grandparents (.50) would be shown if at least 5 stars     - Scholarship example:       * Let’s say we have a scholarship for 1st and 2nd years and STEM majors.       * User selected 3rd and 4th+ years and the correct STEM major.         + We’ll have         + Chem | STEM | Chem         + Chem | Academic | Chem         + Chem | Science | Chem         + 3rd | Undergraduate | 1st | 0         + 3rd | Undergraduate | 2nd | 0         + 4th | Undergraduate | 1st | 0         + 4th | Undergraduate | 2nd | 0         + 4th …..         + Resulting in the event not showing up. |