

CSCI-E599, Software Engineering Capstone

Milestone #3



Web Psychology Testing

Athittaya Kamlang-ek

David Lieberman

Jennifer Le Hégaret

Jeremy Clark

Jonathan Moon

Table of Contents

Abstract	4
Introduction	4
Demographics	6
Designing the Experiment	7
Recruiting participants	8
Reporting	10
Secure (double-blind) messaging	10
Potential threats to validity and generalizability	11
Conclusion	12
References	12
Design of the System	14
Venkman Software Design	14
Front-end Architecture	14
Back-end AWS Development	17
Middleware Used	19
Use Case Diagram	19
Component Diagram	20
Code Organization Diagram	22
Deployment Diagram	23
Additional Information	23
Testing Approaches and Results	24
Developer Testing (AWS Serverless Framework)	24
Security Testing	24
Formal Testing	24
User Login/Logout Testing	25
Experiment Creation Testing	28
UI Testing with Users	29
Load Testing	29
Development Process	29
Meeting the Requirements	29
Estimates	32
Timeline Estimates	32
Hours Estimates	32
Functional Point Accounting	32
Venkman Application Functional Point Count*	32

Functional Point Analysis	34
Lessons Learned	34
Amazon Web Services	34
Beware of New Technology	35
Communication	35
Roles	35
Get a Functional Product Fast	35
Continually Get User Feedback	36
Risks	36
Appendix	39
Appendix A: Final User Stories	39
Appendix B: System Installation/Developer Installation Manual	43
Appendix C: User Manual	43
Appendix D: Development Schedule	49
Appendix E: Journal Submission	54
Appendix F: Functional Point Accounting Estimates Details	54
Appendix G: Manual Tests Documentation	62
Appendix H: Developer README from GitHub	87

Abstract

Amazon Mechanical Turk has recently become popular among social scientists as a tool for gathering experimental data, although in many ways it is not an ideal platform for designing and conducting web-based psychological experiments. Venkman is an online web experiment platform tailored specifically to address the needs of social science researchers, providing a graphical interface (GUI) to build experiments and recruit online labor markets to participate in their studies. Critically, Venkman will also enable researchers to present precisely timed stimuli and capture and report accurately measured user response times. Based on a survey of published research, we conclude that data quality for online experiments via the Internet is adequate and reliable, making online interactive experimentation using Venkman, a tool expressly suited for this purpose, a valuable complement to laboratory studies.

Keywords: psychology, experimentation, online research, online labor markets

Introduction

Online labor markets such as Amazon Mechanical Turk (MTurk) are increasingly popular tools for experimental scientists (Buhrmester, Kwang & Gosling, 2011). With large and diverse pools of people ready to perform tasks promptly for pay, these markets present researchers with new opportunities to recruit participants for experiments. Numerous studies have been conducted using online data collection and their general conclusions are promising: classic results from psychology and economics studies can be replicated using online samples, and the data obtained online is deemed as reliable as that obtained via traditional methods (Chesney et al.; Horton et al.; Goodman et al.; Shapiro et al.).

The vast potential offered to researchers by the prospect of online, crowdsourced research is hampered by the lack of a platform that supports experiment design features crucial to the implementation of an effective study. Existing platforms do a reasonable job of organizing the participant's exposure to the range of stimulus types for which the web is an ideal delivery mechanism, including text, images, audio and video, and do an equally reasonable job of collecting responses registered as typed text or multiple choice selections. Often, however, they place limitations on the researcher's ability to control how the stimuli are presented and how response information is captured. Capabilities such as the precise timing of exposure to stimuli, accurate measurement of participant response times, controlled randomization of pre-defined sets of stimuli or the registration of a response as a mouse-click on a specified screen region either require a significant code development effort on the researcher's

part or are missing from the platforms altogether. The Venkman project seeks to address these needs and thereby redefine the model for conducting online experiments.

Venkman¹ is an online software package for developing and distributing web-based human experiments to participants across the globe. The graphical experiment-builder interface is simple and intuitive, allowing experiments to be developed quickly without the need for custom programming as shown in Figure 1.

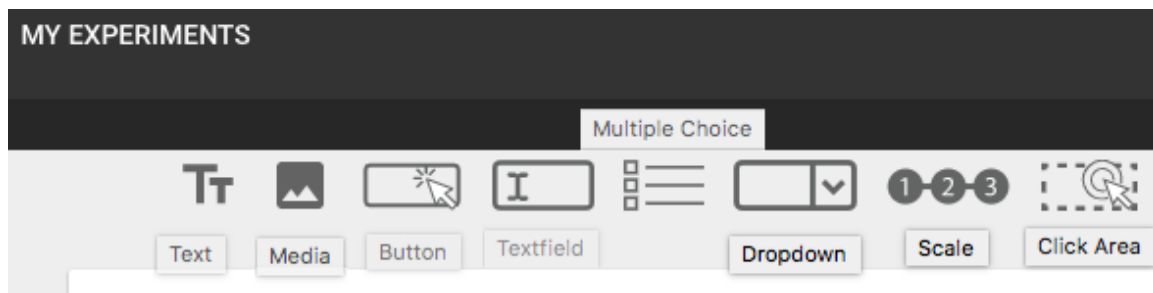


Figure 1: Components made available to researchers in Venkman’s authoring GUI include text, media, buttons, text fields, multiple choice questions, dropdown menus, scales and clickable areas.

With the ability to limit the total participation count, as illustrated in Figure 2, and to define a demographic profile for each experiment to ensure it is made available to a suitable participation pool, researchers can further curate their experiments to match their objectives.

Name
Psychological Pain and Self-cohesion

Description
By participating in this study you have the opportunity to contribute to research relating to suicide prevention and intervention.

Estimated Time In Minutes	Participant Cap ●	Incentive (in USD) ●
20	80	10

Status ●
Active ▼

Figure 2: Researchers can add a description, estimated time in minutes, participant cap and incentive amount to their experiment information.

Response data supplied to the researcher include not only the responses themselves but the timing of user responses captured with millisecond precision as well as fully anonymized relevant demographic information on each participant.

¹ The Venkman project is named in tribute to the life and work of one of the towering thinkers of our time, a man whose tireless efforts across multiple streams of endeavor to advance the cause of rationality in the face of an increasingly irrational world require no further elaboration or introduction (Reitman, 1984).

Venkman's commitment to preserving participants' anonymity may encourage them to participate fully in an experiment, as Fidler and Kleinknecht (1977) found that when participants were given guarantees of confidentiality, rates of non-response fell. There is also evidence that anonymous participants are inclined to answer less self-protectively and more accurately than participants who believe they can be identified (Bloom & Hautaluoma; Centra; Fidler & Kleinknecht; Landy & Farr; Stone et al.). On accessing the website, a registered participant receives a list of available experiments matching the participant's own self-reported demographic profile and a brief description of each experiment, including an estimated duration and the participation incentive to be received upon completion.

Demographics

Another notable disadvantage of MTurk as a platform for human subject testing is the demographic skewing of the user population. It has been reported that the potential subject population for studies conducted via MTurk available in the United States skews more female, with lower self-reported incomes and higher education than the general population (Gosling et al., 2004). Venkman seeks to ameliorate this problem by empowering researchers to define the demographic characteristics of the user population eligible to participate in a given experiment, as in Figure 3, and also to set a limit on the maximum number of users who will be accepted as participants.

Allow participants who match all the following (similar rules are OR'ed together) ⓘ:

Gender	=	Male	🗑️
Highest Level of Education	<	Associate degree	🗑️
Color Blindness	!=	Complete color blindness	🗑️

[+ Add Rule](#)

Audience Summary:
 (gender = "male") AND
 (educationLevel < "3") AND
 (colorBlind != "complete blindness")

Figure 3: Researchers can define the characteristics of the participant eligible to participate in an experiment. Here, the researcher has defined participation rules requiring respondents who are male, with an education level lower than an Associate degree (e.g. Some college, High school, etc.) and are not completely color blind. Only participants who meet all of these criteria will see this experiment in the list of experiments they are eligible to complete.

Additionally, researchers may use Venkman for convenience sampling purposes before deploying their experiment to a larger or more specific population. This purposeful sampling practice can help maximize the efficiency and validity of a study, regardless of

whether the methodology employed is qualitative or quantitative (Morse, 2016). Researchers can invite members of a sample population to take an experiment by sending them a link to the Venkman website and having them sign-up as participants on the Venkman platform as shown in Figure 4. Once these participants have registered with Venkman, they may be given access to the experiment by direct URL.

Figure 4: Signing up for an account

Designing the Experiment

Researchers benefit from the Venkman platform's services and functionality at various stages of the research process. When designing the experiment, researchers can choose from the common rudimentary survey platform or short-answer questionnaires. They can also import images, audio clips or video clips to serve as additional experimental stimuli and can control the exposure time of these elements to the participant as in Figure 5. These millisecond exposure time data are collected automatically and made available to researchers via download for further analysis.

X	Y	WIDTH	HEIGHT
167.92	-43.067	627.84	497.25

Figure 5: Researchers can control the exposure time of a stimulus by Venkman's Auto-Advance mechanism, which accepts a delay value in seconds. Here, the Auto-Advance is set to 5 seconds.

Venkman also allows the researcher to define groups of stimuli and to randomize the order of their presentation to the participant within the context of the individual group.

Recruiting participants

Participant recruitment through profile matching is key to the Venkman model. Upon creating a new account, each participant will have an opportunity to enrich his or her profile with additional demographic information which will be applied to the registry of published experiments to assemble matches (see Figure 6). The participant is not required to supply any additional profile information to Venkman, but the richer the participant's personal profile, the greater the number of experiments he or she may be eligible to complete. The researcher creating an experiment may choose to specify no participant profile constraint on the experiment, making the experiment available to every registered Venkman user.

For any given demographic attribute, the researcher may specify a single matching value, in which case only participants matching the specified value for that attribute will be eligible to complete the experiment. Alternatively, the researcher may specify a range of values or define a set of discrete acceptable matches; a participant's eligibility would then require a value that falls within the specified range or belongs to the defined set. Finally, the researcher may specify simply that a given demographic attribute must have some value, without declaring a preference as to what that value might be. Only a participant whose profile matches all of the attribute rules required for an experiment is eligible to complete it.

Optional Profile Details

Completing the rest of your profile information is optional. By filling out more of your profile, you may be eligible to participate in more experiments.

Note: The data below will be included in your responses when you participate in an experiment.

Date of Birth
August ▼ Day 31 ▼ Year 1973 ▼

Gender
Male ▼

Which race/ethnicity best describes you?
Asian / Pacific Islander ▼

Highest Level of Education
Graduate degree ▼

Household Income
\$500,000

Marriage Status
Married ▼

Address City State Zip
Massachusetts ▼ 02134

Color Blindness

- ✓ Not color blind
- Red-green color blindness
- Blue-yellow color blindness
- Complete color blindness
- Other

Figure 6: Example of a participant filling out profile information. Participants are not required to supply any profile information to Venkman, but the richer the participant's profile, the greater the number of experiments he or she may be eligible to complete.

Because user privacy is a critically important to the Venkman team, no participant's Personally Identifiable Information (PII) is ever shared with any other Venkman user, whether a researcher or a fellow participant. Only the values the

participant has provided for demographic data points specified by the sponsoring researcher will be included in result reports, as discussed in further detail below.

Reporting

In Venkman, the primary medium for supplying researchers with the results of their experiments will be downloadable comma-separated-values (CSV) or JavaScript Object Notation (JSON) files. Each participant's response to each test will include the response itself as well as the elapsed time in milliseconds between the exposure of the stimulus to the participant and the time at which the participant responded. No participant's PII will be included in the downloaded data as CSV or JSON. If, however, the researcher has declared a demographic profile to define the experiment's participant pool, the relevant demographic data points for each participant will be included in the downloaded file. As an additional privacy safeguard, participants' demographic data points in categories the researcher has not specified for the experiment will not be included in the downloaded file. To account for the possibility that participants may update some demographic profile information over time, the relevant participant profile values will be reported to the researcher as of the time the individual participant's response was collected, rather than as of the time the researcher downloads the experiment results.

Secure (double-blind) messaging

Upon reviewing the results of an experiment, a researcher may wish to make contact with a participant whose responses are of particular interest. Venkman does not share PII in the downloaded report, so no contact details are available to the researcher; instead, each participant's results will include an encrypted token that can be redeemed to send a double-blind message via the Venkman interface. Figure 7 shows the encrypted token included in response data downloaded as a JSON file. Although Venkman will by default provide no PII in the generated message to the participant, the researcher is free to share his or her contact information in the sent message form, as shown in Figure 8. The decision to respond to this overture and engage in further dialog with the researcher is left entirely to the participant.

```

{
  "row[1]response.Finish/New Button/expectedValue": null,
  "occurrenceIndex": 1,
  "row[1]response.Intro/Begin/expectedValue": null,
  "eventIndex": 1,
  "eventType": "Group Change",
  "row[1]response.Intro/Begin": "Begin",
  "row[1]response.Finish/New Button": "Finish",
  "timeSinceGroup": 3,
  "timeSinceLastAction": 3,
  "path": "/Experiment",
  "timeSinceStart": 3,
  "presentationScreenIndex": 1,
  "timeSinceScreen": 3,
  "timestamp": 1524968217404,
  "group": "/Experiment",
  "isCorrect": null,
  "participantId": "us-east-1:34f7c4b4-f02e-4458-bea6-66a2467139b1",
  "experimentId": "52dd3ed7-2052-44c8-812e-5d5b0616de40",
  "messageToken": "fbe33d983b2acb64f2265742842958cc4a593867687a7dc6cbca203750c1177a585d4caada09d50f14c77c3eace7e1f67E6Qlcdl",
  "deviceScreenWidth": 1920,
  "os": "Windows",
  "osVersion": "10",
  "viewportWidth": 1216,
  "browser": "Chrome",
  "browserVersion": "65.0.3325.181",
  "deviceScreenHeight": 1080,
  "viewportHeight": 949
},

```

Figure 7: Message token in JSON results data

Contact A Participant

Participant Message Token
 fbe33d983b2acb64f2265742842958cc4a593867687a7dc6cbca203750c1177a585d4caada09d50f14c77c3eace7e1f67E6Qlcdl

Subject
 Experiment follow-up

Message
 Hello, I'm interesting in some follow up analysis of your results. If you would be willing, please contact me at winston.smith@1984.com

Reply to email

SEND

Figure 8: Secure message form to contact participants.

Potential threats to validity and generalizability

There are two fundamental challenges to the validity of online experiments. The first of these challenges reflects the concern that participant pools are insufficiently representative of the general population. As discussed above, Venkman addresses this risk by empowering researchers to define the characteristics of an experiment's participant pool. Should the researcher choose to publish an experiment with no

defined demographic rules, however, we project that the demographic profile of the resulting participant pool would be comparable to that of a study implemented using other online labor market platforms such as MTurk. In our estimation, then, Venkman will provide data gathered from participant pools that are slightly more demographically diverse than standard Internet samples and are significantly more diverse than the standard subject pools at American colleges (Buhrmester et al., 2011).

The second challenge concerns the overall quality of the data that participants provide, which can be affected on several fronts. For example, the incentive to complete an individual experiment published via Venkman is not substantial, as is also the case for MTurk, which may in turn raise questions about individual participants' level of commitment. It is also possible that the platform's emphasis on ensuring user anonymity may encourage individual participants to create "dummy" accounts in order to reap the incentives of giving multiple responses to the same study. To date, however, there is little evidence to suggest that data collected online are of poorer quality than data collected from subject pools in the physical lab (Buhrmester et al.; Gosling et al.; McCredie & Morey). Overall, we therefore project that Venkman will prove to be at least as reliable as a source of experimental data as MTurk, and in certain aspects may significantly improve upon it.

Conclusion

The foregoing discussion together with published research suggest that researchers should consider Venkman preferable to currently available online tools for collecting experimental data. Venkman's intuitive graphical builder interface will enable researchers to create experiments without programming knowledge and without the need for clumsy integrations with online survey tools such as *SurveyMonkey* or *Qualtrics*. The software's online labor market gathering capability informed by robust profile-matching algorithms will allow researchers to define the characteristics of their target populations, while result reports including millisecond-precise response timings will offer them opportunities to deepen their analyses. In sum, the practical advantage of a comprehensive online experiment delivery tool built expressly to meet the needs of the researcher community should make Venkman the obvious choice for conducting online experiments in the twenty-first century.

References

Bloom, A. J., & Hautaluoma, J. E. (1987). Effects of message valence, communicator credibility, and source anonymity on reactions to peer feedback. *The Journal of Social Psychology*, 127(4), 329-338.

Buhrmester, M., Kwang, T., & Gosling, S. D. (2011). Amazon's Mechanical Turk: A new

source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1), 3-5.

Centra, J. A. (1979). Determining Faculty Effectiveness. Assessing Teaching, Research, and Service for Personnel Decisions and Improvement.

Chesney, T., Chuah, S. H., & Hoffmann, R. (2009). Virtual world experimentation: An exploratory study. *Journal of Economic Behavior & Organization*, 72(1), 618-635.

Fidler, D. S., & Kleinknecht, R. E. (1977). Randomized response versus direct questioning: Two data-collection methods for sensitive information. *Psychological Bulletin*, 84(5), 1045.

Goodman, J. K., Cryder, C. E., & Cheema, A. (2013). Data collection in a flat world: The strengths and weaknesses of Mechanical Turk samples. *Journal of Behavioral Decision Making*, 26(3), 213-224.

Gosling, S. D., Vazire, S., Srivastava, S., & John, O. P. (2004). Should we trust web-based studies? A comparative analysis of six preconceptions about internet questionnaires. *American psychologist*, 59(2), 93.

Horton, J. J., Rand, D. G., & Zeckhauser, R. J. (2011). The online laboratory: Conducting experiments in a real labor market. *Experimental economics*, 14(3), 399-425.

Landy, F. J., & Farr, J. L. (1980). Performance rating. *Psychological Bulletin*, 87(1), 72.

McCredie, M. N., & Morey, L. C. (2018). Who Are the Turkers? A Characterization of MTurk Workers Using the Personality Assessment Inventory. *Assessment*, 1073191118760709.

Morse, J. M. (2016). *Mixed method design: Principles and procedures* (Vol. 4). Routledge.

Reitman, I. (1984). *Ghostbusters*. Columbia Pictures.

Shapiro, D. N., Chandler, J., & Mueller, P. A. (2013). Using Mechanical Turk to study clinical populations. *Clinical Psychological Science*, 1(2), 213-220.

Stone, E. F., Spool, M. D., & Rabinowitz, S. (1977). Effects of anonymity and retaliatory potential on student evaluations of faculty performance. *Research in Higher Education*, 6(4), 313-325.

Design of the System

Venkman Software Design

Venkman provides a web-based, easy-to-use environment for computerized experiment design, data collection, and participant recruitment. The software also allows researchers to collect response data with millisecond-precision timing for their analyses.

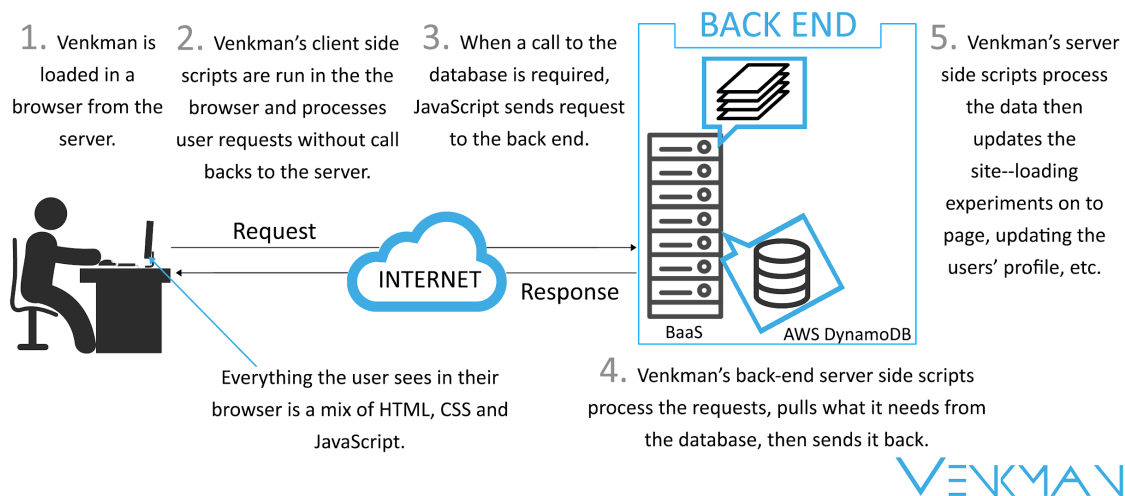


Figure 9: Overview of Venkman's architecture

The front-end architecture is developed using the React JavaScript library. The back-end server side (shown in the blue box in Figure 9) manages all requests generated by user activity. Front-end scripts send these requests to the server to be processed upon the completion of an experiment by a participant. This emphasis on the client experience allows response timings to be captured accurately and consistently by eliminating the threat to precision presented by discrepancies in network speeds and latency.

Front-end Architecture

Venkman's success as a viable tool for the efficient development of large-scale research projects depends on a user interface that is intuitive, easy-to-use, powerful and interactive. The complexity of the demands on the user interface requires a framework that supports reusable, easily composable, and stateful components. Given its widespread adoption by a large, committed developer community, its support for the lightweight construction of reusable components and its impressive performance, React was selected as the framework for Venkman's user interface. In conjunction with React, Material UI Next, a library of components based on the Google Material Design Specification, is used throughout Venkman for the buttons, text fields, dialogs, etc. to create a consistent, professional look.

The creation of an experiment is a core activity in the Venkman application. The code excerpt in Figure 10 illustrates the basic structure of a sample experiment. The structure is recursive, consisting of a collection of objects each of which represents either an individual screen or group of screens, which may in turn have a collection of children of the same two types. All screens can have user interface elements such as buttons, text fields, images, and so on, as dictated by the researcher's experiment design. Groups may also be associated with datasets, a custom data structure that supports looping over a repeatable experience, a feature also shown in Figure 10.

```

1 {
2   "screens": {
3     "useDataSet": true,
4     "id": "Experiment",
5     "isGroup": true,
6     "dataSetId": "Dataset #1",
7     "children": [
8       {
9         "id": "Intro",
10        "elements": [
11          {
12            "x": 144,
13            "width": 200,
14            "y": 126,
15            "id": "New Image",
16            "type": "image",
17            "url": "{row.url}",
18            "height": 200
19          },
20          {
21            "type": "button",
22            "id": "New Button",
23            "x": 194,
24            "y": 426,
25            "width": 100,
26            "height": 50,
27            "text": "Continue",
28            "onClick": "{next}"
29          }
30        ]
31      }
32    ]
33  },
34  "dataSets": [
35    {
36      "rows": [
37        [
38          "id",
39          "column_1",
40          "url"
41        ],
42        [
43          "1",
44          "Image 1",
45          "http://images.com/test.jpg"
46        ]
47      ]
48    }
49  ]
50 }

```

Figure 10: Code excerpt showing an example of an experiment structure.

Back-end AWS Development

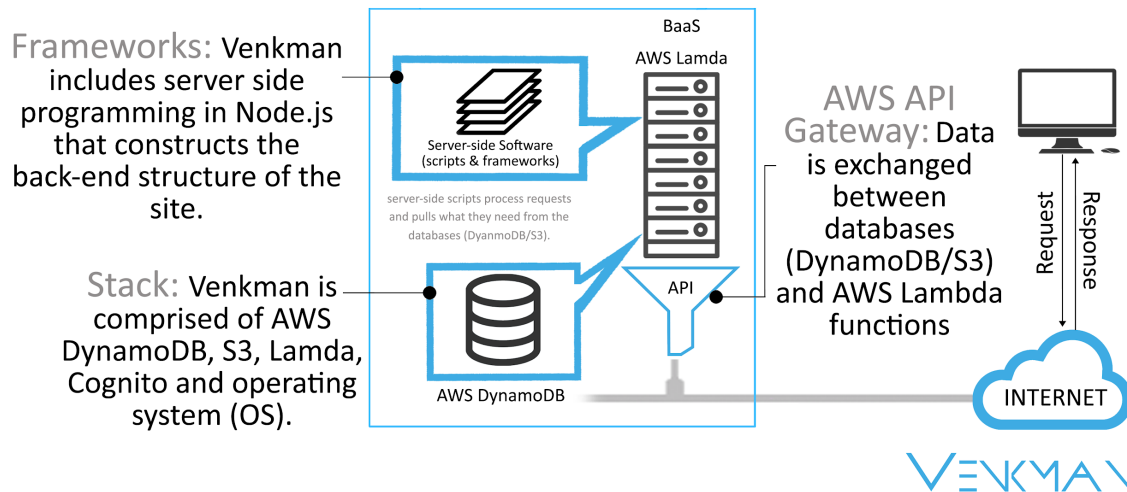


Figure 11: Overview of Venkman's back-end architecture

Figure 11 above illustrates Venkman's back-end architecture, which follows the Amazon Web Services 'serverless' architecture pattern, making extensive use of AWS components, including AWS Lambda, DynamoDB database tables, S3 file storage buckets, API gateway, and Cognito. Each of these components makes a vital contribution to the Venkman implementation:

- **AWS Lambda:** a serverless, event-driven computing platform that runs stateless, self-contained blocks of code in response to events and automatically manages the compute resources required by that code. AWS Lambda provided Venkman with a resilient, scalable platform for executable code while at the same time encouraging separation of concerns by isolating logic, removing the need to manage server uptime and maintenance and reducing the application's overall code execution cost.
- **DynamoDB:** a NoSQL database service that supports data retrieval by key-value query. DynamoDB's highly flexible approach to the persistence of structured data enabled the team to adopt a rapidly iterative development process, characterized by frequent and occasionally dramatic changes to core data models at a pace that would have been nearly unthinkable in a system founded on a more traditional relational database.
- **S3:** a reliable, highly available and scalable cloud storage solution. Venkman relies on S3 buckets for the storage and management of media files, including

images, audio and video, uploaded by researchers for use in their experiments, and for the storage and delivery of the client-side application code. S3 was also critical to the development process as the staging area for code deployments via AWS Serverless Architecture Model (SAM) tools.

- **API gateway:** a middleware layer that allows RESTful web service endpoints to pass events to other AWS components. Venkman uses Amazon API Gateway to invoke AWS Lambda functions, providing users with the lowest possible latency for API requests and reducing code execution costs against the maintenance of a monolithic application server, since API costs are calculated per execution, rather than as a function of server running time.
- **Cognito:** a user account management and authentication solution. All Venkman users, both researchers and participants, must be registered users bearing authentication credentials and system-verified email addresses. Cognito supplies an out-of-the-box process that lets users easily sign-up and sign-in, as well as a powerful mechanism for handling both coarse-grained authorization based on user role and fine-grained authorization based on the user's relationship to individual rows of data stored in DynamoDB tables.
 - **IAM (Identity and Access Management) Roles:** Used in conjunction with Cognito, IAM roles provide subtle, fine-grained mechanisms for handling and observing authorization permissions that all but eliminate the need for custom authorization checks in Lambda execution code.

Middleware Used

Venkman's AWS Middleware act as a “glue” between the application and its network. It controls the flow of information between the application and the Lambda functions, DynamoDB, and S3.

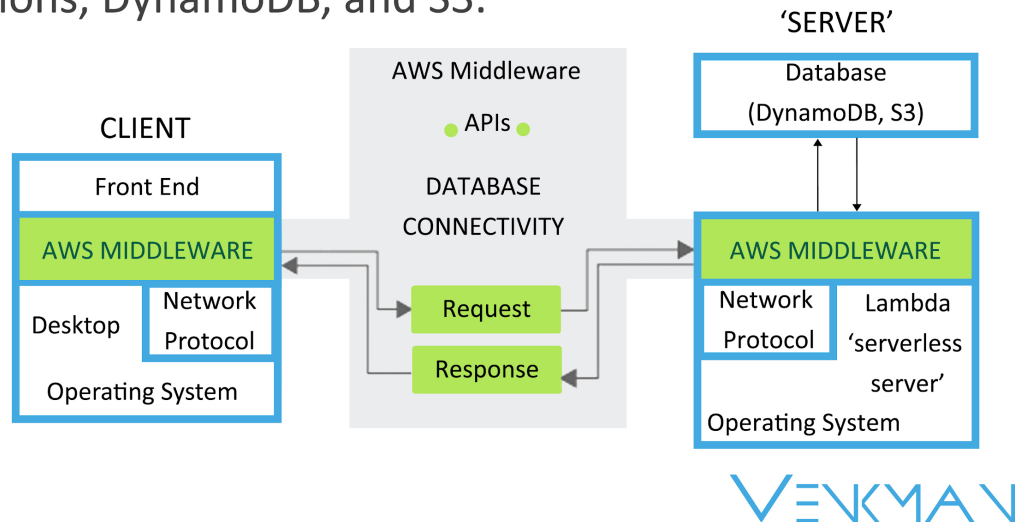


Figure 12: Overview of Venkman's middleware architecture

In essence, our middleware layer is managed by AWS integrations as shown in Figure 12 above. The integrations among API Gateway, AWS Cognito, AWS Lambda and AWS DynamoDB fill the roles typically assumed by common middleware components such as Tomcat, Weblogic, Express, Spring Boot, JDBC drivers, etc. In some ways the AWS SDK used in Lambda functions can be considered middleware of a sort, but the heavy lifting is done through configurations of the various AWS components.

Use Case Diagram

Use Cases: There are three distinct user roles defined within Venkman: the Researcher, Participant and a Super Admin. Figure 13 below shows different use cases for Venkman.

Researcher	creates experiments, gathers data, and may also take experiments
Participant	takes experiments
Super Admin	enables behind-the-scenes functionality

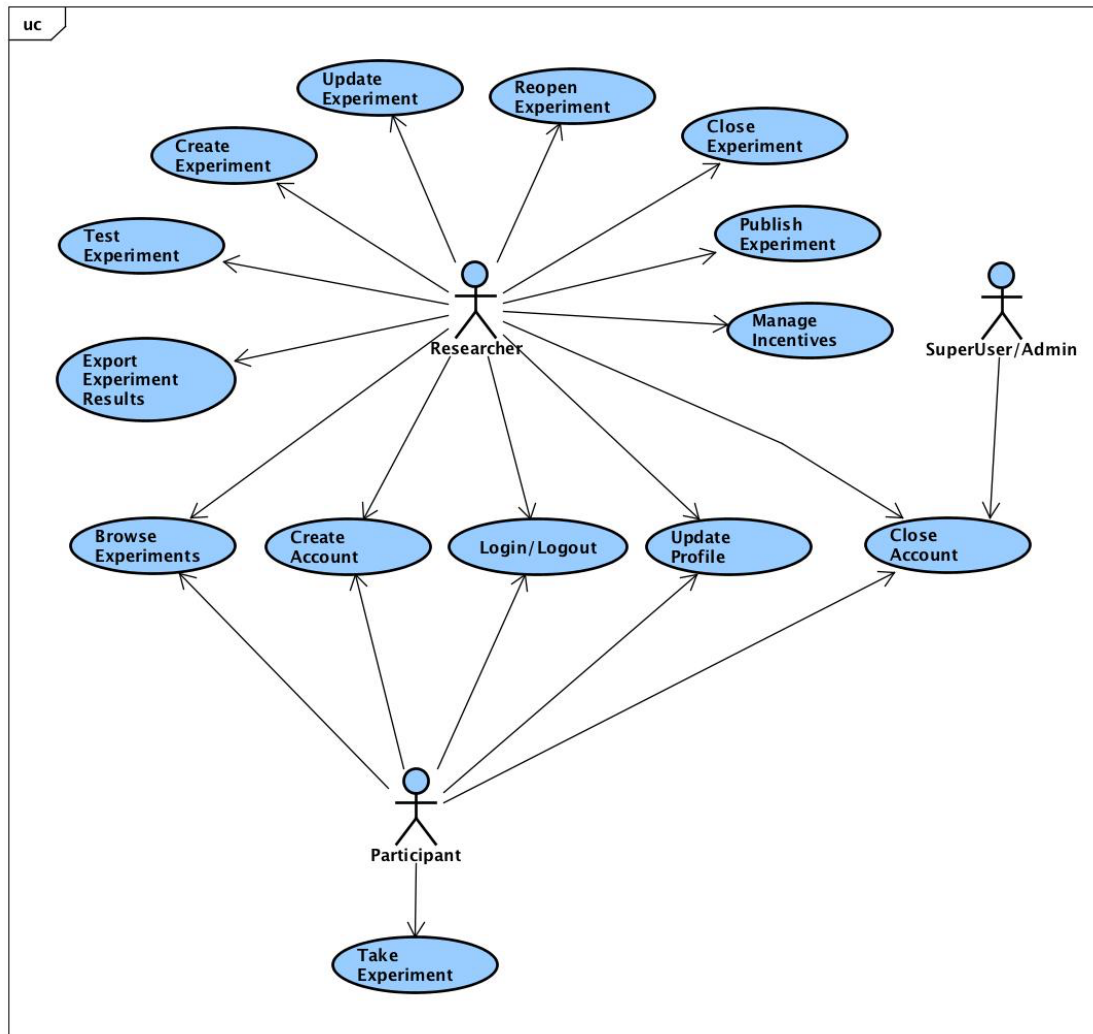


Figure 13: Venkman's UML Use Case Diagram

Component Diagram

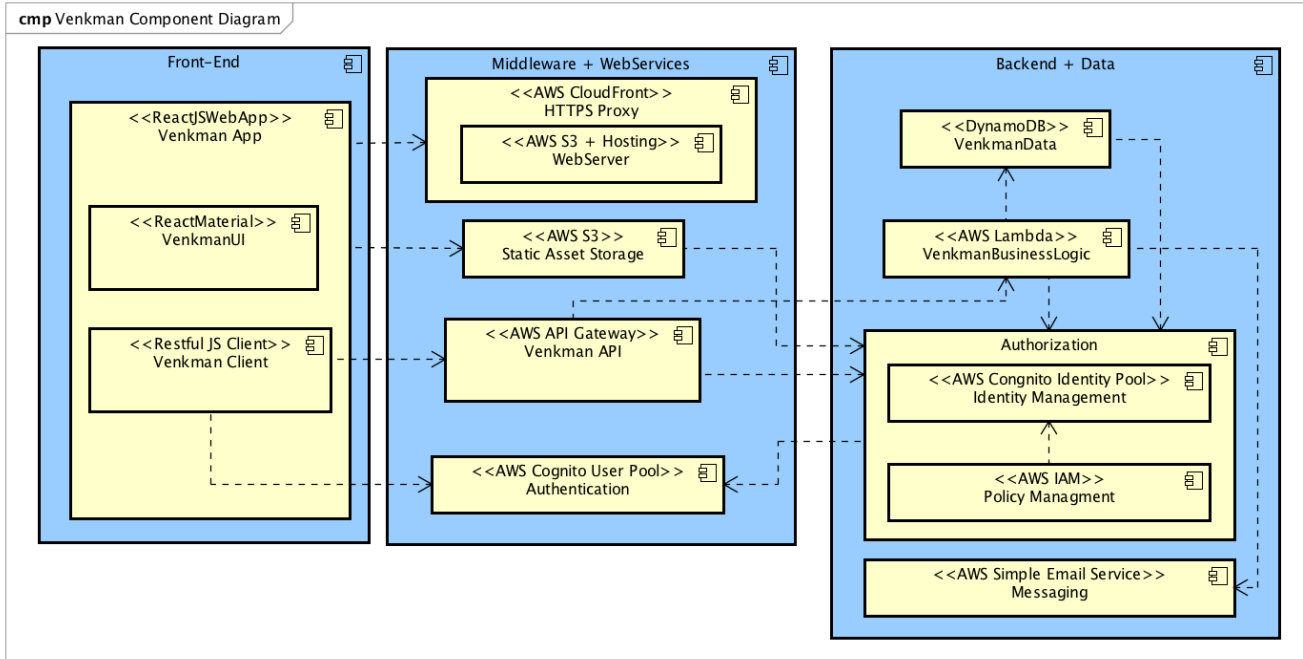


Figure 14: Venkman’s UML Component Diagram.

Figure 14 above is a diagram in Unified Modeling Language (UML) format providing an overview of the components that make up the Venkman application. The user accesses the application using a web browser pointed to the user-friendly URL:

<http://venkmanscience.com>

As is consistent with standard practice in the design of “stateless” web applications, the user’s session state is maintained entirely on the Front-End tier. The Middleware tier represents the AWS components with which the user interacts via the client. As noted above, AWS Cognito handles user sign-up, sign-in, logout as well as account deletion requests; AWS S3 serves up both the Client tier’s code and various uploaded media files researchers may add to their experiments; and AWS API Gateway exposes the RESTful API endpoints invoked by calls from the Client. The Data tier contains components that are fully shielded from direct user access: the AWS Lambda component where all server-side code runs, the AWS DynamoDB database schema which houses all application-specific persisted data, and the AWS Simple Email Service (SES), used to support double-blind messaging between researchers and participants.

Code Organization Diagram

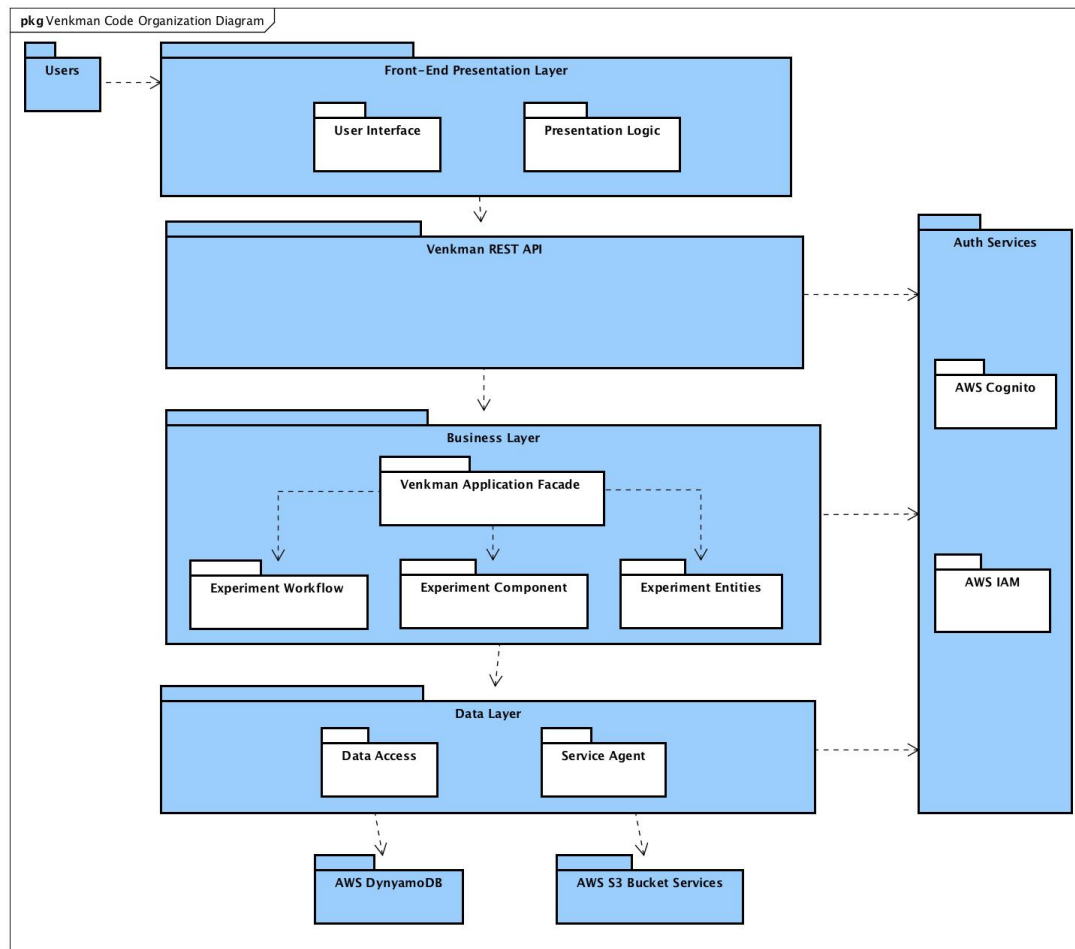


Figure 15: Code Organization Diagram

The UML diagram shown in Figure 15 illustrates the conceptual organization of the Venkman application codebase. Of note here is that both the RESTful API calls themselves, through configured AWS component integrations, as well as explicit code integrations deployed in the Business Layer, interact with the Auth Services layer that encapsulates AWS Cognito and AWS IAM. This allows the application to make transparent use of AWS authorization features where possible, while still having the ability to perform custom checks against those features where required by application logic.

Deployment Diagram

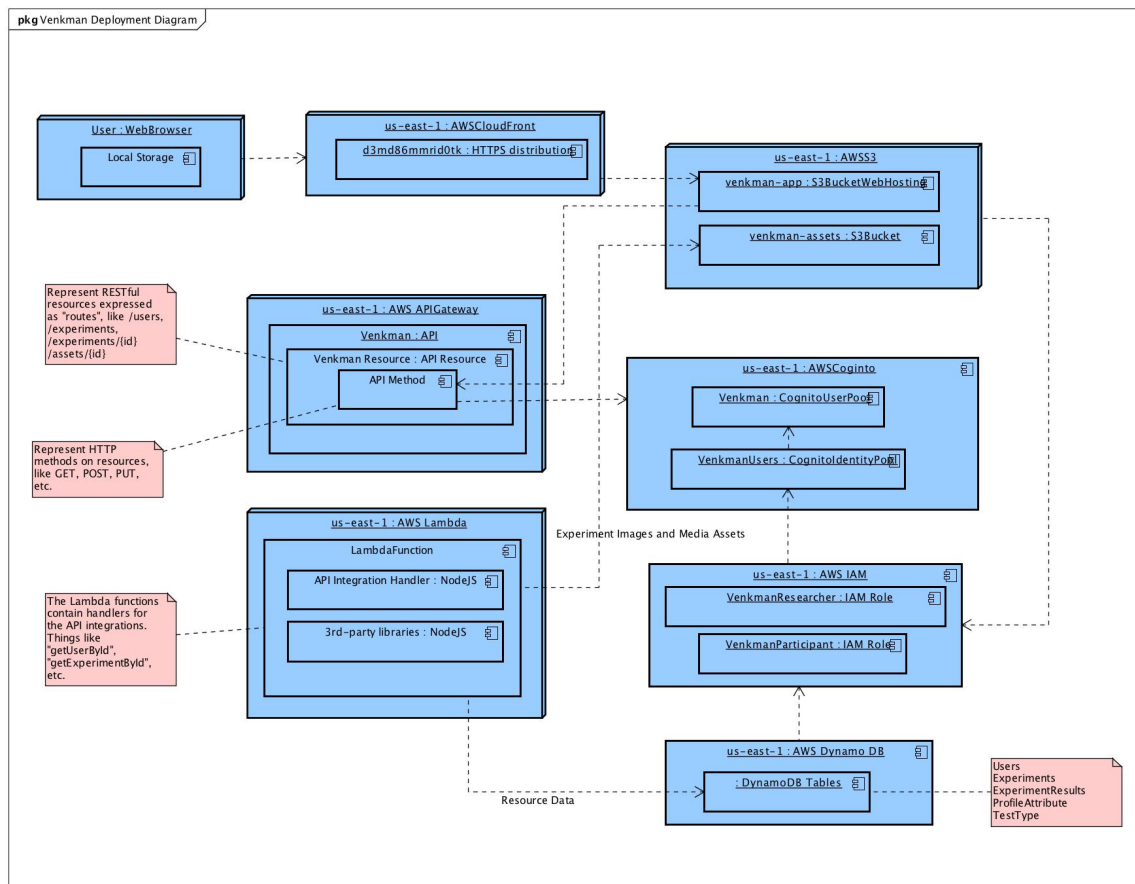


Figure 16: Deployment Diagram for Participant Taking an Experiment. A larger version of this diagram is included in Appendix H.

The UML diagram in Figure 16 above is a deployment diagram showing the physical architecture of Venkman as it deploys to various AWS components. All of the components are currently hosted in the AWS “us-east-1” region, so our high-availability posture will need some adjustment should we choose to scale Venkman to other regions in the world. Regarding the API gateway and AWS Lambda nodes, it should be noted that we have many more API Resources and Lambda Functions, but in order to conserve space, this diagram does not contain the instance nodes of each API Resource and Lambda function. The same is true for the DynamoDB tables. A note in the left-hand corner lists the tables used by Venkman; we have DynamoDB tables for Users, Experiments, and Experiment Results.

Additional Information

Additional information about the Venkman system is available in the Appendix H “README” section.

Testing Approaches and Results

Developer Testing (AWS Serverless Framework)

To develop and validate our Lambda functions, Team Venkman relied heavily on ad-hoc testing using Postman against local processes running under the AWS Serverless Application Model (AWS SAM). This approach supported rapid development of Lambda functions integrated with cloud-based application components such as our DynamoDB schema and S3 file storage solution, without requiring deployments to the cloud for each code adjustment.

Security Testing

Since Venkman mainly uses Amazon Web Services' (AWS) platform for our Back-end as a Service (BaaS) and data storage, the software adopts the AWS Shared Responsibility Model for its data security solution. The Shared Responsibility Model lets AWS handle the security of the cloud, while Venkman handles security in the cloud. To satisfy our obligation to our users, we have employed security measures (e.g. Cognito, IAM) to ensure that their information is secure. For example, all interactions with tables in the application's DynamoDB schema are handled with database connections defined against the credentials of the user currently in session, rather than against the application-scoped credentials characteristic of more traditional models; this ensures that only the user identified as the owner of a particular data point may modify it. Our testing strategy for this layer of the application was largely informal, consisting of assuming user identifies and confirming that behaviors were either available or not available as expected.

Formal Testing

Team Venkman uses Manual, Cypress and Postman to test the functionality of the requirements communicated by our client. Our initial focus was entirely on functional testing; although we were (and remain) confident in the AWS environment's ability to perform effectively under user load, we determined comparatively late in the process that AWS component configuration can have significant effect on the system's response to load, discussed in further detail below under "Load Testing."

While we have some automated testing in place, most of our testing was done manually. Here is the breakdown of our testing strategies:

Test Strategy	Focus	# Tests	% PASS	% Fail
Postman	Direct API testing, security.	55	98% (54/55)	2%*

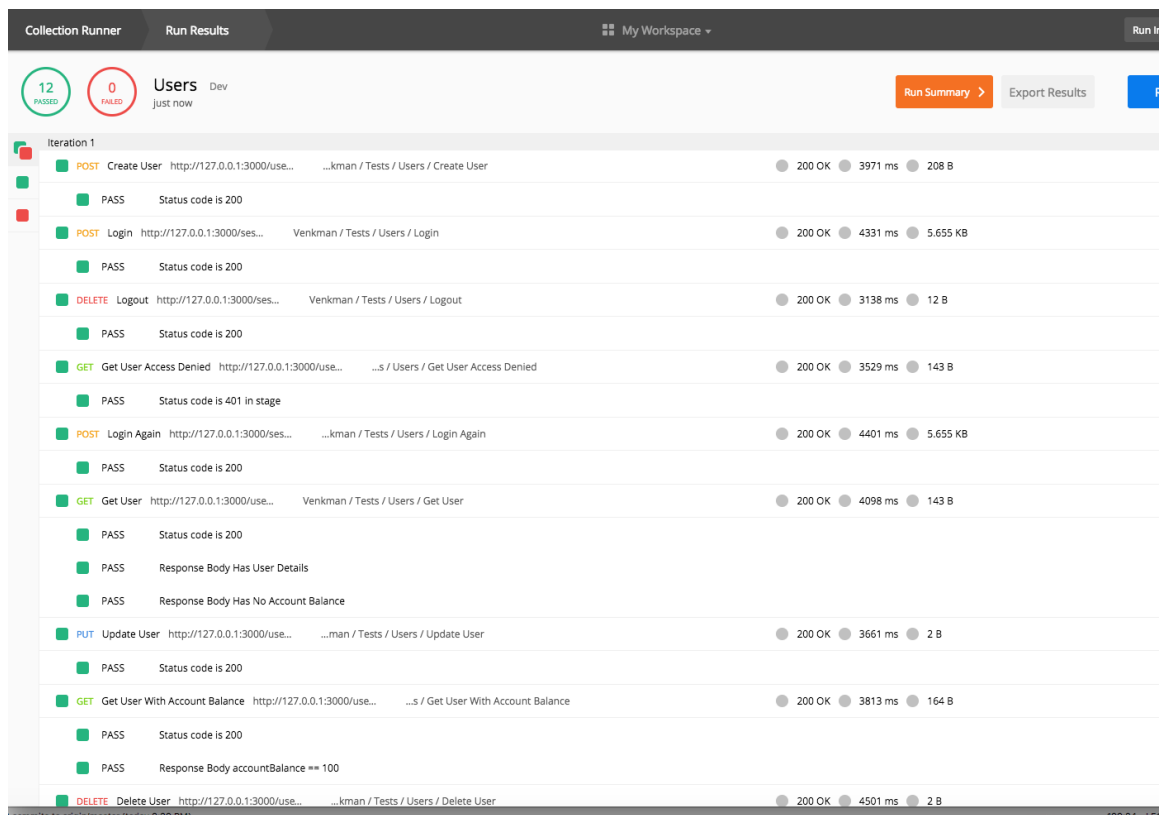
(automated)				
Cypress (automated)	Account management (9/42 reqs)	9	100%	0%
Manual Human Eye	All V1 requirements	42	100%	0%
Manual Human Eye	13 additional features (in-app visualization, researcher contact, correct vs. incorrect responses, etc.)	13	100%	0%

The Manual Tests are fully documented in detail in Appendix G titled “Manual Tests Documentation.” Cypress was chosen to do mostly automated front-end testing (e.g. logging-in/logging-out) because of its familiar Javascript syntax/format and its capability of running real time tests in the browser. Figure 17 below shows an in-browser test using Cypress. In addition to Cypress, Postman is also used to test API suites. Postman was chosen for its practical ‘Postman Collection Runner’ that allows the team to run and test all of our requests quickly in controlled sequence mimicking real usage.

User Login/Logout Testing

The screenshot displays a Cypress test runner interface. On the left, a list of test steps is visible, including 'VISIT /', 'GET a', and several assertions for the 'SIGN UP AS A PARTICIPANT' flow. The main area shows a web browser with a 'Please check your email' message, indicating that an email has been sent to a test address and a link to 'Verify Email' is provided.

Figure 17: Screenshot running the requirement “As a user, I can create either a researcher account or participant account.” on Cypress



The screenshot displays the Postman Collection Runner interface for a collection named 'Users'. It shows a summary of 12 passed and 0 failed tests. Below the summary, a table lists individual test results for 'Iteration 1', including the test name, HTTP method, URL, status code, response time, and size.

Test Name	Method	URL	Status	Response Time	Size
POST Create User	POST	http://127.0.0.1:3000/use...	200 OK	3971 ms	208 B
PASS Status code is 200	PASS				
POST Login	POST	http://127.0.0.1:3000/ses...	200 OK	4331 ms	5.655 KB
PASS Status code is 200	PASS				
DELETE Logout	DELETE	http://127.0.0.1:3000/ses...	200 OK	3138 ms	12 B
PASS Status code is 200	PASS				
GET Get User Access Denied	GET	http://127.0.0.1:3000/use...	200 OK	3529 ms	143 B
PASS Status code is 401 in stage	PASS				
POST Login Again	POST	http://127.0.0.1:3000/ses...	200 OK	4401 ms	5.655 KB
PASS Status code is 200	PASS				
GET Get User	GET	http://127.0.0.1:3000/use...	200 OK	4098 ms	143 B
PASS Status code is 200	PASS				
PASS Response Body Has User Details	PASS				
PASS Response Body Has No Account Balance	PASS				
PUT Update User	PUT	http://127.0.0.1:3000/use...	200 OK	3661 ms	2 B
PASS Status code is 200	PASS				
GET Get User With Account Balance	GET	http://127.0.0.1:3000/use...	200 OK	3813 ms	164 B
PASS Status code is 200	PASS				
PASS Response Body accountBalance == 100	PASS				
DELETE Delete User	DELETE	http://127.0.0.1:3000/use...	200 OK	4501 ms	2 B

Figure 18: Screenshot running user API tests from Postman Collection Runner. All of these tests are available in our Git repo here:

https://github.com/jerclark/venkman-app/blob/master/test/Venkman.postman_collection.json

In Figure 18 above, the Postman Collection Runner was used to test the API. This is different from the tests using the Newman Utility that Postman offers. The Newman Utility (shown in Figure 19 below) allows us to run and test a Postman Collection directly from the command line. This is useful because we can easily integrate it with continuous integration servers and build Venkman’s systems.

Experiment Creation Testing

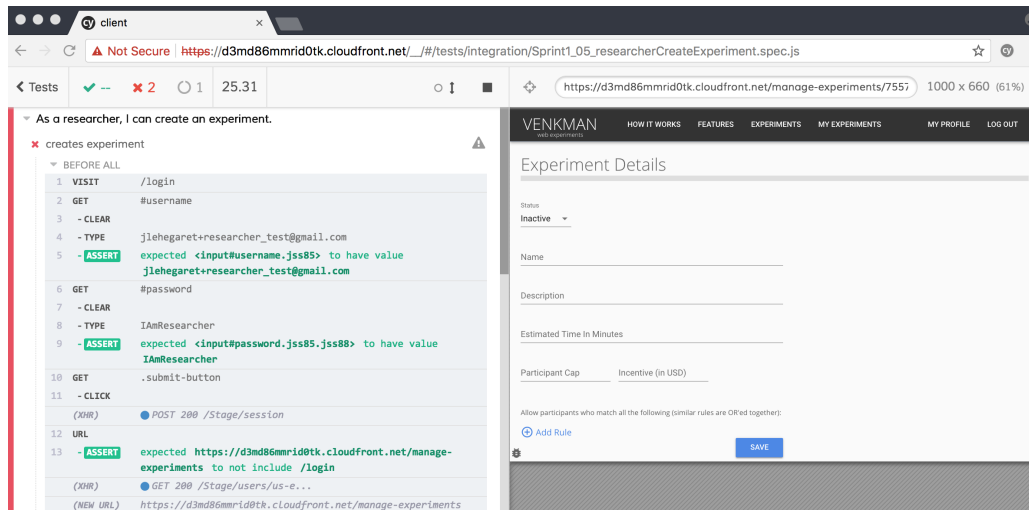


Figure 20: Screenshot running the requirement “As a researcher, I can create an experiment.” on Cypress.

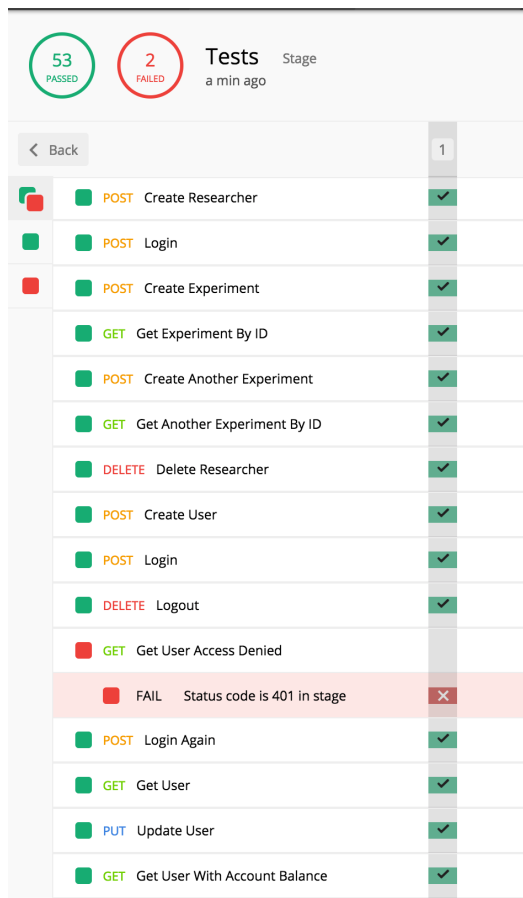


Figure 21: Screenshot of the results received from running Experiment API tests from Postman Collection Runner.

	executed	failed
iterations	1	0
requests	42	0
test-scripts	83	0
prerequisite-scripts	61	0
assertions	55	2
total run duration: 2m 49.6s		
total data received: 55.63KB (approx)		
average response time: 3.9s		

Figure 22: Screenshot of the result from running the same Experiment API tests as Figure 20 using the “Newman” utility.

Figures 20, 21 and 22 above shows our team using Cypress and Postman to test the steps associated with the creation of an experiment in Venkman.

UI Testing with Users

Team Venkman tested our software with at least a few users. Overall, we received comments that the “Preview” functionality worked well, the website is intuitive, and that individuals liked the “Publish Details” option.

Load Testing

The AWS platform is a well-tested environment with an earned reputation for stability, resilience, and performance under load. This reputation played no small role in the decision by the Venkman development team to adopt AWS as our hosting solution. Late in the development process, however, we learned that AWS does require thoughtful configuration to deliver on its promises of scalability and load tolerance. Put to the test by multiple simultaneous users, the application experienced failures that had never occurred previously. Following an intensive diagnostic exercise, we determined that our initial approach to the configuration of throughput values for the tables in our DynamoDB schema, while more than sufficient to support development and functional testing, was imposing a throttle under real usage scenarios. After performing a round of all-hands ad-hoc testing to verify this assessment, we defined a scripted solution exercising key, data-intensive behaviors of the application. The test, executed by scripting Postman, reproduces a participant experience:

- Create a new user account
- Login
- List eligible experiments
- Select an experiment
- Submit an experiment response
- Logout

Running this test repeatedly with thirty simultaneous processes has satisfied us that with appropriate configuration the application is performant under load, and can be configured to scale successfully to address real-world usage scenarios.

Development Process

Meeting the Requirements

Team Venkman meticulously recorded and followed all of the requirements

communicated to us from our client. We have completed all the requirements in the table below under Appendix D. We achieved this through meeting every Tuesday and Thursday for at least 2 hours, communicating almost everyday on Slack and organizing the requirements on Trello. One of the best decisions we made was to meet at least twice every week for the duration of the project. Our whole team is efficient in our communication through our Slack channel and each of us take less than 24-hours to answer questions posed by a team member. As seen in Figure 23 below, our team has sent more than 3,000 messages in the time-span of approximately 3 months.

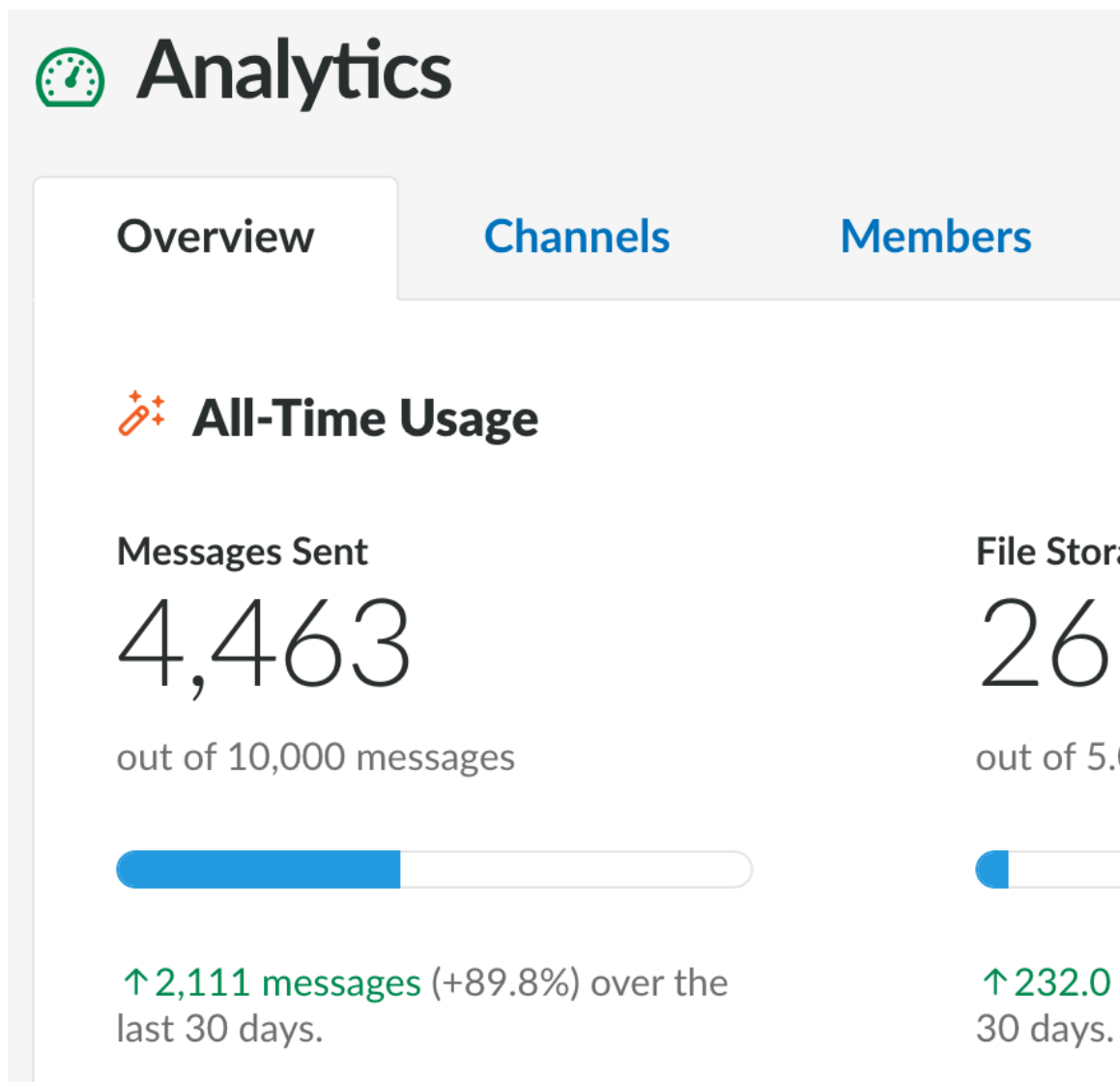


Figure 23: Analytics from Team Venkman's Slack channel.

One team decision that introduced some unanticipated risk was the commitment to build our software on a serverless model. The learning curve was unexpectedly steep at points but not, in the end, insurmountable. We also backed away

from our initial plan to test all requirements on Cypress/Postman rather than relying primarily on the Manual Tests documented in Appendix G. In practice, the pace of frequently dramatic changes in the evolution of the Venkman user interface turned out to be a rather poor fit with a programmatic testing model better suited to regression testing changes to a more mature application. Finally, we might, on reflection, reconsider or at least review our decision to rely on AWS roles for handling permissions within the application. We were ultimately able to achieve the authorization functionality we desired, but at the potential cost of some opacity in the behavior that a more bespoke approach might have avoided, as well as some churn in the early stages of application development as team members' understanding of the the AWS permissions model evolved.

For our software development process, Team Venkman did not use the agile 2-week iterations but rather a design-code-test paradigm that remained in operation throughout the semester. We quickly developed an appreciation for short iterations introducing small changes to the feature set. We would design a feature, code it, test it, get user feedback, and repeat that same process until we were satisfied with it. At any point in time there might be several feature loops occurring simultaneously, each at a different point in its cycle. This process allowed each of us to work continuously at full capacity, and thus as a team to complete all our project requirements. We therefore operated by the Fail-Fast system principle in system design. This principle is summarized in the Figure 24 below. We also separated our project into Sprints and have been able to complete every requirement on all Sprints detailed in Appendix D.



Fail Fast: A strategy of trying something, getting fast feedback, and then rapidly inspecting and adapting.



<http://www.innolution.com/resources/glossary/fail-fast>

Figure 24: Venkman team operated by a fail-fast system in system design, where one immediately reports at its interface any condition that is likely to indicate a failure.

Estimates

Timeline Estimates

Team Venkman's estimated timeline for our development schedule (detailed in Appendix D) was particularly accurate. Some requirements took longer than anticipated but the general total hours spent was very close to our estimates. We have met with our customer, shared our software, had our customer try using the software and proactively identified problems within the software ourselves in order to improve its processes. We did not change our hour-count timeline estimates to finish the project.

Hours Estimates

Team Venkman's estimated hours to complete the requirements for the Sprints were slightly less than the actual hours. Although we have underestimated the time required to complete the requirements, we did add additional features to the software which extended the amount of time spent on fully completing the requirements. Details are in the table below:

	Estimated Hours	Actual Hours
Sprint 1	122	134
Sprint 2	130	136
Sprint 3	119	122
Sprint 4 & 5	$(130) + (104) = 234$	$(117) + (112) = 229$
Total Hours Spent on Original Requirements	595	621
Total Hours Spent on Additional Features	0	110
Total Hours:	595	731

Functional Point Accounting

Venkman Application Functional Point Count*

**Please refer to "Functional Point Accounting Estimates Details" in the Appendix F for more information.*

- **Purpose of FP Count:** This application functional point count assessed the current functionality provided by Venkman Version 1.0.
- **Scope of FP Count:** This count was defined by the data, experimental screens, and user information.
- **Boundary of FP Count:** The boundary of this FP count defined Venkman as a simple, standalone software application. FP counts involved the processes of creating experiment, taking experiments, and creating users.

Types of Functions–There are two types of functions

1. Data Function (ILFs and ELFs)

- a. **Internal Logical Files (ILFs)** = data that is stored and maintained within the boundary of Venkman
 - = 28 Unadjusted Functional Points
- b. **External Logical Files (ELFs)** = data that Venkman will use/reference, but is data that is not maintained by Venkman
 - = 0 Unadjusted Functional Points

**Currently in Venkman there are no ELFs, so ELFs contribute zero FPs to the overall FP count. However, going forward, the application may require the use of a file (e.g. “Researchers’ Data Incorporation”) and this file may need to be accessed by Venkman Version 2.0. In this case, this file will be counted as an ELF.*

2. Transactional Function (EIs, EOs, EQs)

- a. **External Inputs (EI)** = process that processes data or control information that comes from outside Venkman boundary
 - = 41 Unadjusted Functional Points
- b. **External Outputs (EO)** = process that sends data or control information outside Venkman boundary
 - = 0 Unadjusted Functional Points
- c. **External Inquiries (EQ)** = process that sends data or control information outside Venkman boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF or ELF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No

ILF is maintained during the processing, nor is the behavior of the system altered.

■ = 6 Unadjusted Functional Points

The total unadjusted function point count for Venkman is 75 function points.

Functional Point Analysis

The total unadjusted function point count for Venkman is 75 function points. For Milestone #3, our team has completed all of the 75 unadjusted function points (i.e. 100%) out of the 75 unadjusted function points for the software application shown in Figure 25 below.

Unadjusted Functional Point Analysis

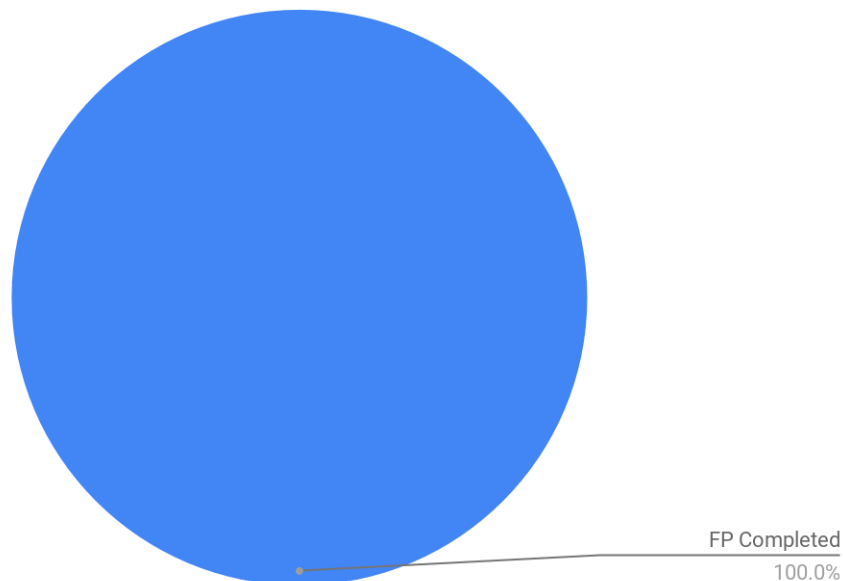


Figure 25: Pie chart of unadjusted functional points completed for Milestone #3

Lessons Learned

Amazon Web Services

The entire team took away a deeper appreciation of the range of capabilities on offer in AWS. We were particularly impressed with the power of the permissions model available through the use of AWS IAM in conjunction with Cognito, although we did find that the management of permissions through the AWS console was not especially intuitive, and some amount of trial-and-error time was required to achieve a level of

expertise. Similarly, the flexibility of the NoSQL DynamoDB database contributed significantly to the speed of our development cycles, but some of the risks associated with its deeper subtleties, such as finding the optimum per-table throughput configurations or coding correctly for retrieving information from large datasets, were hidden from view until rather late in the project.

Beware of New Technology

The “serverless” approach to web application architecture is still relatively young, and the tools available to support it are younger still. To remain faithful to our commitment to working with AWS-specific tools, we adopted Amazon’s Serverless Application Model (SAM) tool as the framework for our development process. We discovered that the SAM project is itself still evolving, and some features, such as localized versions of critical components like Cognito and IAM, are not available, requiring a distributed development process integrating locally-executed code with cloud-based system dependencies. Given a somewhat longer project lifecycle, it might have been useful to evaluate the capabilities of SAM against the more mature third-party product Serverless (<https://serverless.com/>), but we found that a full exploration of competing toolsets was a greater time-cost than we could afford.

Communication

The team remained in frequent communication throughout the life of the project over a variety of platforms, including Slack, Zoom, Google Hangouts and Github. In addition to twice-weekly scheduled team meetings, we often held smaller breakout sessions with two or three team members to address specific technical challenges and brought the results of those sessions back to the full team for review. We consider our communication model and overall team dynamic one of our chief success stories.

Roles

Early in the project we identified the strengths of each member and defined a process that allowed us to take the fullest advantage of our diversity of experience and expertise, while at the same time committing ourselves to a unified technology stack that enabled individual team members to move between components with relative ease.

Get a Functional Product Fast

Our iterative Fail Fast process which enabled us to define and satisfy minimally functional capabilities quickly also exposed issues early, giving us time to resolve them effectively with well-integrated code rather than with a panicky patchwork of unnecessary and potentially unmanageable complexity.

Continually Get User Feedback

Deploying minimally functional implementations early in the project also allowed us to consult with our customer and get valuable feedback early in the project lifecycle. Along with customer check-ins conducted as frequently as scheduling would allow, we sought evaluations from users having no previous experience with the product at various points during the development process, with an emphasis on identifying features that were confusing or unintuitive. This was critical in preventing our product from drifting from the customer's vision, allowing us to achieve a result that not only meets but, we believe, exceeds the customer's expectations.

Risks

Every software project faces risks. Here are some of the risks associated with our software and mitigation strategies for each:

Risk	Mitigation Strategy	Effectiveness
Communication and Coordination: a geographically dispersed team requires thoughtful coordination (<i>Week 1, Vasa</i>)	Early on, we established our communication and coordination tools. Slack's ability to provide asynchronous yet categorized communication works particularly well for a team dispersed across time-zones. The Activity Logs provided by Trello allow for clarity about who changes what as we communicate about the scope of the product. Google Docs allow for seamless collaboration on our mutual deliverables.	Yes, our usage of Slack and Trello was effective in coordinating our communication.
Division of Labor: as skill levels are not evenly distributed, experts may tend to work in silos. (<i>Week 1, Vasa</i>)	We have identified leads for certain tasks which play to each team member's strengths. However, we also plan to engage in collaborative design, identify dependencies early, and perform peer reviews across all work.	Yes, our labor was divided equally according to roles played in the team. We also collaborated in several tasks

		such as design and peer reviews.
Overcommitment: too many ideas and too little time.	We took full advantage of the time leading up to Milestone #1 to refine the requirements list. After the initial brainstorming sessions, we continued using peer review, combined with multiple customer check-ins, to move more and more of the non-core requirements to the “future” until we attained an average of 10 requirements per two-week Sprint.	Yes, using peer reviews were effective in setting a schedule for our Sprints. Peer reviews and collaborations are two of the top reasons our estimates were exceptionally accurate.
Overpromising: is millisecond accuracy of time capture even possible?	<p>The accuracy of response time capture is a fundamental requirement of the Venkman system from the customer’s point of view. Therefore, even if it is a high bar to attain, we must try. However, how can we evaluate whether or not we succeed? We will:</p> <ul style="list-style-type: none"> • perform timings with no network dependencies • create appropriate test cases to validate timing • model timings as a timed micro-event stream in which patterns can be defined as meaningful events 	Response time is accurately captured in Venkman.
Under-delivering: might the UI become little more than a “survey builder”?	While we do intend to provide as simple an experiment-building tool as possible for researchers, we will also focus on the critical value-added by Venkman: built-in	While subjective, Team Venkman believes that our

	<p>timing capture. In addition, we intend that Venkman will provide extensible features to many different types of media options which are generally not a part of surveys, such as the ability to add (and time) video clips, audio files, and multiple image files.</p>	<p>software is not under-delivering in its requirements for Version 1.0.</p>
<p>End User Failure: Can the product provide high flexibility and yet maintain sufficient ease-of-use?</p>	<p>We will strive to iterate over mockups and prototypes quickly via customer/user testing to create a UI that is sufficiently elegant, intuitive, and powerful enough to be used by researchers for real-world experiments.</p>	<p>From the comments gathered during our UI tests, Venkman was “easy to use” and “intuitive”.</p>
<p>Security and Overcommitment: the Payment system</p>	<p>Compensating participants for their time is a fundamental consideration for a real-world experiment. However, with the semester’s time, we would rather not take on the security requirements of dealing with real money, nor necessarily commit to learning an outside payment system’s API (nor necessarily commit Venkman to that particular payment system).</p> <p>Therefore, we plan to implement a minimum viable model for payments as a “stub” for a real-world system. Instead of currency, researchers may award fictitious “points” to participants for their effort.</p> <p>Given time, we may assess possible commercial integrations to see if any are easier to integrate than expected.</p>	<p>The compensation mechanisms in Version 1.0 shows that researchers and participants can access their balance from the dashboard.</p>

Team Dynamic

What had helped our team in terms of roles, dynamics, meetings and communication are the points below:

- Meeting twice per week
- Having a team with diverse skill sets
- Using Slack everyday which enabled quick responses
- Having additional pairing sessions separate from regular team meetings to discuss the project
- Using Git to manage changes to source code
- Using SAM to locally run Lambda and API Gateway through Docker containers.

What had hurt our team in terms of roles, dynamics, meetings and communication are the points below. We also detailed the steps we took to improve these points.

- Time Zone Differences: To improve upon this point we set two weekly meetup sessions through Google Hangout/Zoom to make sure we make time for the project.
- Work/Life Schedule Changes: To improve upon this point we communicated daily on Slack when there were any changes in schedule.
- Remote Google Hangout Instead of Actual Facetime: No initiatives to improve upon this point were taken because so far we have been able to work well from remote places.
- Technical Challenges with Screen Shares/Internet Issues: To improve upon this point we switch between Google Hangouts and Zoom when we encounter screen sharing issues. When we encounter Internet issues, we turn our videos off and put our microphones on mute.

Appendix

Appendix A: Final User Stories

While we gathered our requirements, we documented a few User Stories for our requirements. The ones that are from our Sprints 1 and 2 are partially listed in the table below:

Requirement	User Stories
As a user, I can reset my password.	GIVEN I have forgotten my password WHEN I click the "Forgot Password" link

	<p>THEN I am guided through a password reset flow which will leverage my registered email to validate my identity</p> <p>GIVEN I have reset my password WHEN I revisit the site THEN I will be prompted to login with my newly created password</p>
As a user, I can logout of the site to end my session.	<p>GIVEN the user has logged into the site THEN every page within the site will offer a "logout" option</p> <p>GIVEN the user chooses the "logout" option THEN the same conditions apply as if the user has never logged in (every page within the domain presents the welcome/login page)</p> <p>GIVEN a participant is actively engaged in an experiment WHEN she logs out THEN she will be notified that the experiment she will not be compensated for her participation until she completes the experiment</p>
As a user, I expect all transactions over public internet channels to be encrypted using widely accepted security mechanisms.	<p>GIVEN a user attempts to access the site with HTTP THEN the user will be redirected to a site which uses HTTPS</p>
As a user, I can create either a researcher account or a participant account.	<p>GIVEN a user has chosen to Create an Account THEN the user faces two choices: to create a Research account (to build tests) or a Participant account (to take tests).</p> <p>GIVEN a user chooses to become a researcher WHEN their UI loads</p>

	<p>THEN they see all UI options (researcher and participant)</p> <p>GIVEN a user chooses to become a participant WHEN their UI loads THEN they see all only participant options</p>
As a user, I must login before accessing the experiment site.	<p>GIVEN the user has not authenticated, WHEN the user tries to visit any page within the domain THEN the user sees a welcome/login page</p> <p>GIVEN the user has not provided the expected credentials, WHEN the user submits that login information THEN the user continues to see the welcome/login page</p> <p>GIVEN the user fails to provide correct credentials WHEN the user submits that login information THEN the user is presented with a 'forgot password?'</p>
As a participant I can create a profile with an email and password so that I quickly begin participating in experiments.	<p>GIVEN valid email address is provided WHEN the user proceeds with account setup THEN the user is invited to enter a password</p> <p>Given invalid email address is provided WHEN the user attempts to proceed with profile setup THEN the user is notified that address is already in use, please use different email</p> <p>GIVEN an invalid password is being entered (see constraints) WHEN the user attempts to proceed THEN they are notified why their</p>

	<p>password is invalid</p> <p>Given valid email and password are provided WHEN the user submits their profile THEN an account is persisted AND the user is sent an email notification AND the user is redirected to the "hub"?</p>
As a researcher I can create an experiment title and description which will be displayed on the public bulletin board to all prospective participants.	<p>GIVEN a researcher has chosen Create Experiment or Edit Experiment THEN the screen provided to the researcher has fields to collect/edit the following information: Title, Overview, Public Description.</p>
As a researcher I can add a new experiment in order to conduct new research.	<p>GIVEN a user logs into a researcher account THEN the user sees a page which lists the items listed in the title, as well as a "New Experiment" option and an "Edit Account" option.</p>
As a researcher, I can edit previously created experiments.	<p>GIVEN the researcher clicks on an experiment name listed as a draft on the dashboard THEN the researcher is taken to an "Edit Experiment" page for that experiment.</p> <p>PLUS:</p> <p>GIVEN the researcher clicks on an experiment name listed as published on the dashboard THEN the researcher sees a report page specifically for that experiment, probably containing time per question for that experiment.</p> <p>GIVEN editing an already published experiment THEN the researcher is warned that it could have negative effects</p>

As a researcher I can create instructions for an experiment that are available only to registered/active participants in the experiment	GIVEN a researcher has chosen Create Experiment or Edit Experiment THEN a screen presented to the researcher (sometime after the public description fields, but sometime before the real nitty-gritty of defining the experiment itself) will prompt the researcher to define a more thorough description / instruction set / disclosures about taking the test.
---	---

Appendix B: System Installation/Developer Installation Manual

Using AWS SAM

Our README file in the “README” section below (Appendix H) details how to install and use AWS SAM with Venkman.

Setting Up Accounts and Tools

To create or take experiments using Venkman web service, you must first sign up for an account. A Venkman account is an account that enables you to use services from Venkman.

To sign up for a Venkman account

1. Open <http://venkmanscience.com>, and then choose “Sign Up as a Participant” or “Sign up as a Researcher.”
NOTE: Only ‘Researcher’ can create an experiment.
2. Follow the online instructions.
Part of the sign-up procedure involves verifying your email address.

Appendix C: User Manual

Although our goal is for Venkman to be easy-to-use and self-documented, here are some instructions on using Venkman. **Animated clips on how to use Venkman can be found on the Venkman website at:** <https://d3md86mmrid0tk.cloudfront.net/tips>

Also, a video overview of Vekman is in the Venkman Homepage at:
<http://venkmanscience.com>

PLEASE TAKE A LOOK AT THE LINKS.

Intended Audience: These instructions are intended to be used by system administrators and others experienced in using websites. It will be used in concert with the Venkman software for researchers to create experiments and participants to delete their account. The software is available at: <http://venkmanscience.com>

Creating an Experiment

1. Click on the blue “Create Experiment” button to create a new experiment.
2. Use the Drag and Drop menu located on the top to build each screen of your experiment.
3. Click “Save” on the top menu to save your new experiment before exiting the builder or logging out.

Setting the Width and Height of an Experiment

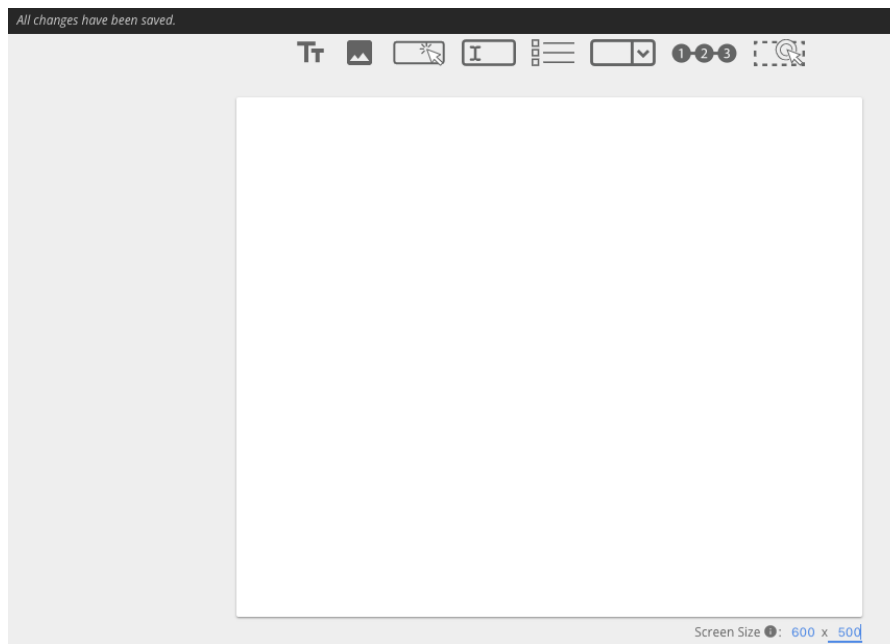


Figure 26: Changing the size of your screen.

1. Change the Screen Size option on the bottom right-hand corner of your experiment page by typing in your dimensions.
2. It is also suggested to set the screen size before building your experiment to avoid having to reposition elements.

Using the Left Rail to Organize your Experiment Screens

1. The left rail is used to organize, group, duplicate, and delete screens.
2. Click on the left rail and select the screen you want to change.
 - Multiple screens can be selected by using Cmd + Click

- Duplicate a screen by right clicking.
- Group screens together by right clicking.

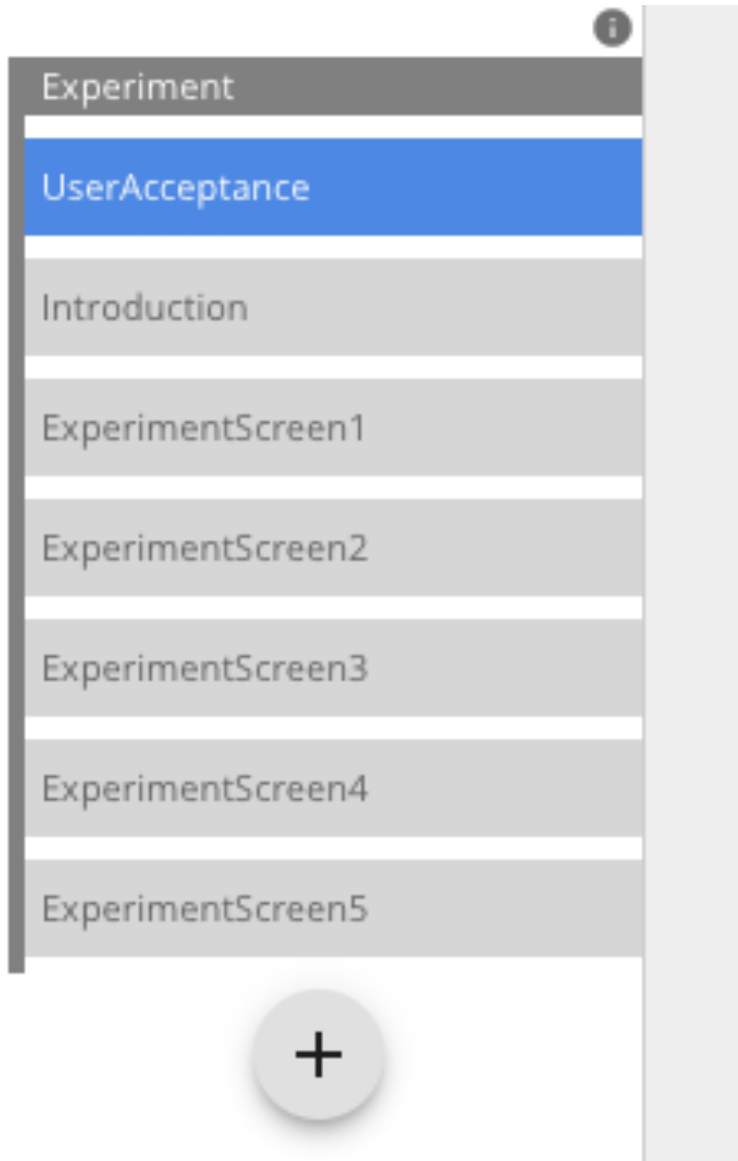


Figure 27: Using the left rail to organize your experiment on your experiment development page.

Using the Media Manager

1. The Media Manager is used to upload and identify media assets used in your experiment.
 - Preloading: All of the media assets uploaded to the Media Manager will be loaded into an experiment before an experiment starts. This ensures timings will be more accurate and not dependent on network speeds.
 - Unique Ids: Since assets are referenced in experiments by an id, the ids need to be unique.

- Supported formats: Images: jpg, png, gif. Audio: mp3. Videos: mp4, ogg, webm
- Uploading The Same File: In order to easily update images and re-upload them, if you re-use the same filename, it will replace the existing image.
- Using the Media Manager

Using Datasets (Group Properties)

1. Datasets allow you to dynamically load data into the elements of any child screen. For every row in a Dataset, the experiment will loop over the Group's children.
2. Datasets can be created/accessed via clicking on a "Group" in the left rail, and the selecting a data set in the right rail. (Learn how to create groups here.)

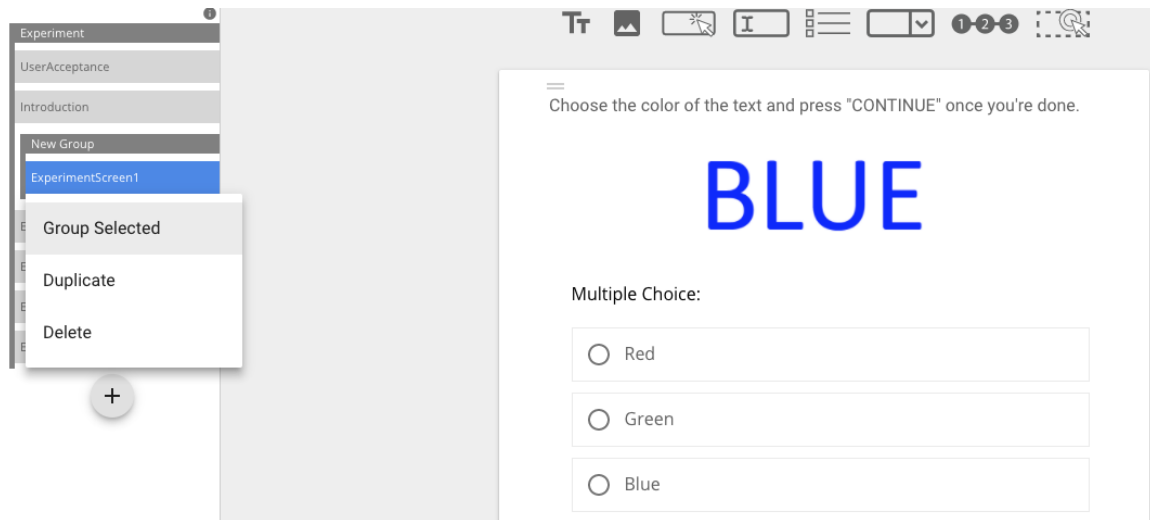


Figure 28: Using Datasets for your experiment.

3. The Dataset table supports adding and removing rows/columns by right clicking.
4. To use a cell value in an Element property, add the desired field in curly braces. For example, {row.option1} would be replaced by the cell value under the header "option1" in the Dataset table.

Working with Randomize Children (Group Properties)

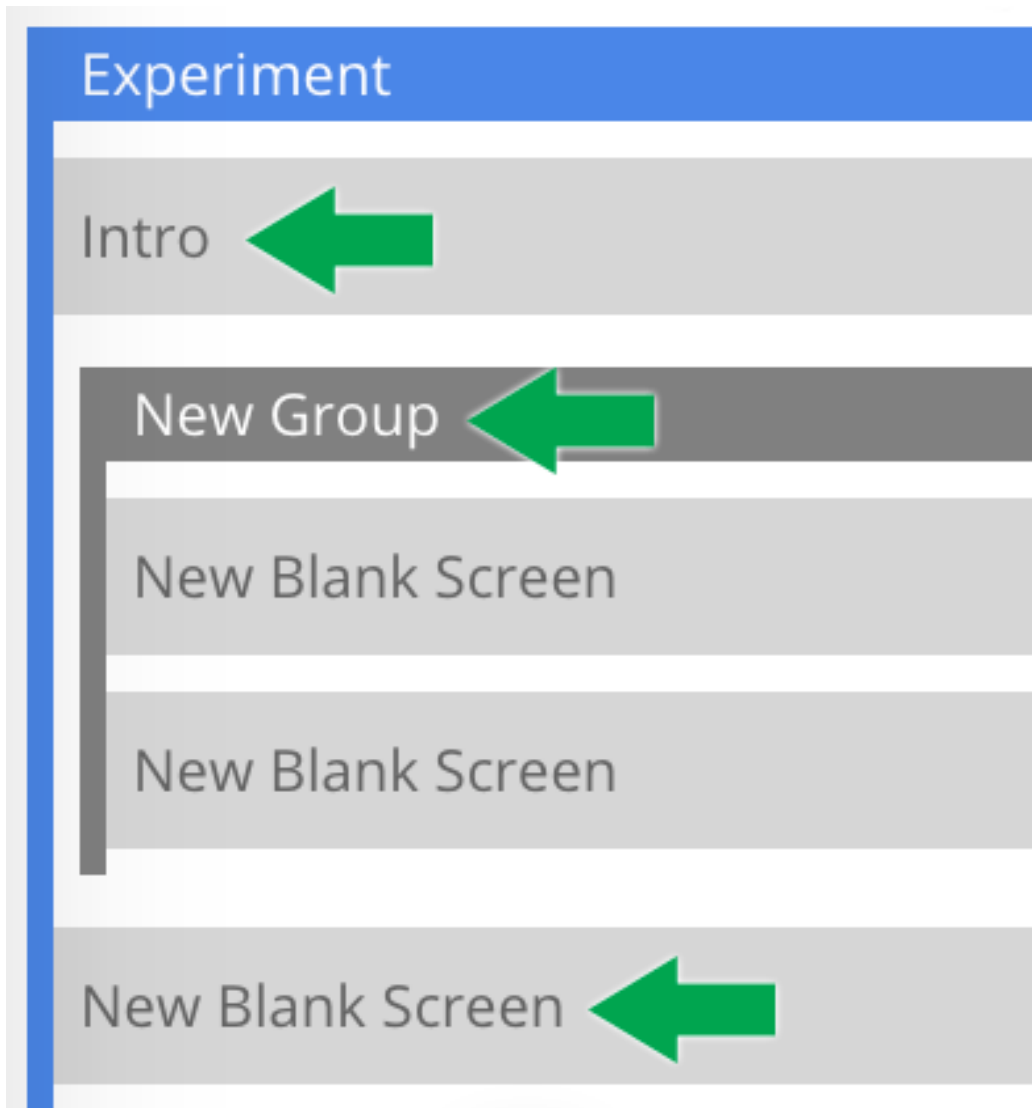


Figure 29: Randomizing Children in your experiment.

1. This option will randomize the direct children of the selected Group.
2. In this example, "Experiment" is selected, and the three green arrows indicate the screens that will be randomized. Note that the screens inside of "New Group" will not be randomized, because they are not direct children of "Experiment".
3. This option is available in the right rail by clicking on a "Group" in the left rail.

Contact A Participant

1. As a researcher, you may contact any participant who has participated in any of your experiments AND has opted in to be contacted by researchers on the "My Profile" page
2. To send a message, you will need the participant's message token. It can be

found in the CSV file downloaded on the Experiment Results page (as in Figure 31 below) or the downloaded JSON file.

experimentId	messageToken	deviceScreen
8c779e06-cb47f5270efa4d887a1c42a6ab3202baf3e5a836bd44a548c10d729db015f1ba19448e054c39ee5dc4ae00d63c9830b83em8dcksnOWYOJpdgsJ2ufR4Qb0cy9iPSRZHYDPM1Ms7knMZf5XZ+LZ3yl+J5Q51OG/6qyEU75o3AwwfEKxP48HYUFWpN5UKMpdUSueMhRrvQYUk		1920

Figure 30: A downloaded CSV file. The messageToken column in your experiment's CSV downloaded file has the particular participant's token.

3. The Contact Participants link is located at the top of the Experiment Results page.

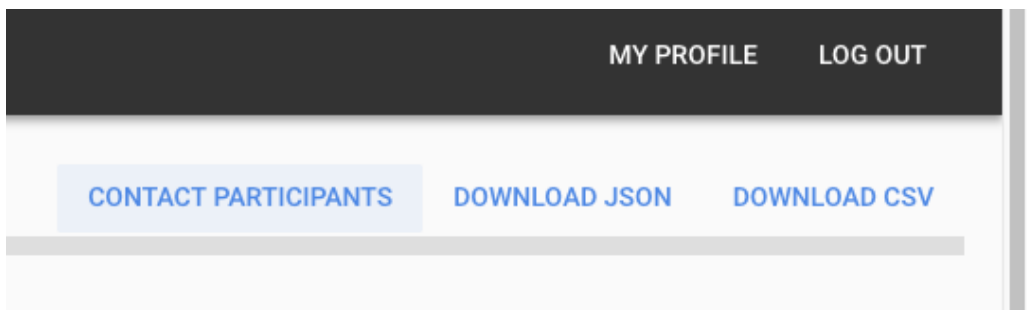


Figure 31: The “Contact Participant” option is located at the top right-hand corner of your experiment data page.

Contact A Participant

Participant Message Token

47f5270efa4d887a1c42a6ab3202baf3e5a836bd44a548c10d729db015f1ba19448e0

Subject

Your Participation in The Little Albert Experiment

Message

I, John B. Watson, would like to...

SEND

Figure 32: A Form that is used to contact a participant.

Deleting Your Account

1. Click on “My Profile” on the top right corner of the website.

2. Click on the blue “Delete Account” text located at the top right corner of your profile.

Appendix D: Development Schedule

The following table details the requirements we have completed. The estimated and actual hours below in brackets include time for research, development, testing, and documentation.

Sprint	Requirement
<p>1</p> <p>March 15</p> <p>(estimated hours: 122)</p> <p>(actual hours: 134)</p>	<ul style="list-style-type: none"> As a user, I can create either a researcher account or a participant account (estimate: 12h); (actual: 19h) As a user, I must login before accessing the experimental site (estimate: 13h); (actual: 15h) As a user, I can logout of the site to end my session (estimate: 3h); (actual: 3h) As a user, I can reset my password (estimate: 15h); (actual: 14h) As a user, I can cancel my account to remove my PII (Personally Identifiable Information) from the system. (estimate: 15h); (actual: 17h) As a researcher, I have a dashboard which lists my current balance(propose to do list on the profile page) as well as any experiments I have created. (estimate: 18h); (actual: 25h) As a researcher I can create an experiment (estimate: 10h); (actual: 12h) As a user, I expect the site to be available to me (estimate: 11h); (actual: 10h) As a user, I expect all transactions over public internet channels to be encrypted using widely accepted security mechanisms (estimate: 15h); (actual: 10h) As a user, I can expect that the data will be secure (estimate: 10h); (actual: 9h)
<p>2</p> <p>March 29</p> <p>(estimated hours: 130)</p>	<ul style="list-style-type: none"> As a researcher I can create an experiment title and description which will be displayed on the public bulletin board to all prospective participants (estimate: 5h); (actual: 4h) As a researcher I can create instructions for an experiment that are available only to registered/active participants in the

<p>(actual hours: 136)</p>	<p>experiment (estimate: 8h); (actual: 7h)</p> <ul style="list-style-type: none"> ● As a researcher, I can edit previously created experiments (estimate: 5h); (actual: 5h) ● As a participant, I can create a profile with an email and password so that I quickly begin participating in experiments (estimate: 9h); (actual: 7h) ● As a participant, I can maintain my anonymity (ie, my identity is protected from researchers) (estimate: 3h); (actual: 7h) ● As a participant, I may have to sign boilerplate agreement terms per test, but this will be handled in as efficient a manner as possible (estimate: 4h); (actual: 4h) ● As a researcher I can add a new experiment in order to conduct new research. (estimate: 5h); (actual: 5h) ● As a researcher I can create named groups of tests within each experiment (estimate: 8h); (actual: 7h) ● As a researcher I can organize my experiments at group / test / screen levels. (estimate: 6h); (actual: 5h) ● As a researcher, I can create an experiment using a visual UI (estimate: 9h); (actual: 10h) ● As a researcher, I can solicit a short answer response to participants to gather their response (estimate: 5h); (actual: 5h) ● As a researcher, I can create an experiment in which the primary stimulus is a digital image (estimate: 15h); (actual: 15h) ● As a researcher, I will have access to a participant's result data once the participant has completed all tests in the experiment (estimate: 9h); (actual: 9h) ● As a researcher I can review every event's start time (estimate: 12h); (actual: 15h) ● As a researcher, I expect elapsed timestamps to be accurate to the nearest ms in order to capture differences in short-duration time spans (estimate: 9h); (actual: 10h) ● As a researcher, I can download all test results into CSV (estimate: 12h); (actual: 15h) ● As a researcher, I can provide a video clip at the beginning of the experiment (estimate: 6h); (actual: 6h)
<p>3</p>	<ul style="list-style-type: none"> ● As a participant, if a screen times out, then I must 'ack' to proceed to the next screen (14h); (actual: 14h)

<p>April 12</p> <p>(estimated hours: 119)</p> <p>(actual hours: 122)</p>	<ul style="list-style-type: none"> ● As a researcher, I can duplicate any test within one of my experiments, and assign it to any group within the current experiment (8h); <i>{actual: 9h}</i> ● As a researcher, I can duplicate any screen within one of my experiments, and assign it to any test within any group within the current experiment (7h); <i>{actual: 6h}</i> ● As a researcher, I can preview my experiments (re: look and behavior) in an unpublished state (15h); <i>{actual: 16h}</i> ● As a researcher, I can specify that groups/tests be presented in a random order within an experiment (9h); <i>{actual: 10h}</i> ● As a researcher, I can save drafts of experiments, and then choose when to publish them (15h); <i>{actual: 16h}</i> ● As a researcher, I can specify that tests be presented in a random order within an experiment (9h); <i>{actual: 9h}</i> ● As a researcher, I can configure a single value to set the delay between all tests (and/or screens) in an experiment or group (9h); <i>{actual: 8h}</i> ● As a researcher, I can configure a single value to apply a time limit to all tests within an experiment or group (12h); <i>{actual: 10h}</i> ● As a researcher, I can duplicate an experiment—the copy preserves all timing and content assets from the original but none of the participant data (14h); <i>{actual: 16h}</i> ● As a researcher, I can duplicate an experiment. (7h); <i>{actual: 8h}</i>
<p>4</p> <p>April 26</p> <p>(estimated hours: 130)</p> <p>(actual hours: 117)</p>	<ul style="list-style-type: none"> ● As a researcher, I must set an approximate "time required" as well as point value to the experiment before it may be published (11h); <i>{actual: 12h}</i> ● As a researcher, I can view which tests are active, with participation rates (16h); <i>{actual: 18h}</i> ● As a researcher, my dashboard will present experiments categorized by status ("Active", "Inactive", and "Archive"). (9h); <i>{actual: 6h}</i> ● As a researcher I can specify the profile criteria of my target audience (17h); <i>{actual: 16h}</i> ● As a researcher I can review/define* technical requirements for an experiment to ensure commonality of platform and/or consistency of environment [*researchers review the participants' hardware but not filter participants according to

	<p>hardware-used] (16h); <i>(actual: 10h)</i></p> <ul style="list-style-type: none"> ● As a participant, I can update my profile with more detailed information about myself in order to fit specific target populations (13h); <i>(actual: 10h)</i> ● As a participant, while I browse the library of experiments for which I qualify, I can sort them by time required, and/or pay, and/or researcher* [*we took out sorting by researcher]. (16h); <i>(actual: 16h)</i> ● As a researcher, I can optionally set a participation limit or end date* that will close an experiment [*we chose participation limit to close an experiment, in addition to researcher just closing them manually]. (11h); <i>(actual: 11h)</i> ● As a researcher, I can count on an experiment disappearing from the board as soon as I have reached my participation goal.;(6h) <i>(actual: 6h)</i> ● As a researcher, I can reopen previously closed tests*. (10); <i>(actual: 6h) [*disallowing previous test takers is not implemented in Version 1.0]</i> ● As a researcher I can require each test be finished within a certain timeframe(5); <i>(actual: 6h)</i>
<p>5</p> <p>May 10</p> <p><i>(estimated hours: 104)</i></p> <p><i>(actual hours: 112)</i></p>	<ul style="list-style-type: none"> ● As a researcher, I can present multiple choice questions to participants (11h); <i>(actual: 12h)</i> ● As a researcher, I can close a published experiment before it has reached its maximum participation rate(7); <i>(actual: 6h)</i> ● As a researcher, I can present Likert Scale questions to participants (13h); <i>(actual: 16h)</i> ● As a researcher, I can play sound files within my tests (12h); <i>(actual: 16h)</i> ● As a researcher, I can play video clips within my tests (12h); <i>(actual: 11h)</i> ● As a researcher, I can ask participants to identify a region of the screen via touch/mouse click (12h); <i>(actual: 14h)</i> ● As a researcher, I have the ability to contact particular participants without violating their anonymity protection(10); <i>(actual: 10h) *We decided to do this requirement.</i> ● As a researcher, I can preview my experiments. (15); <i>(actual: 16h)</i>

	<ul style="list-style-type: none"> ● As a researcher I can randomize or order the presentation of the tests within an experiment at group / test / screen levels. (12h); <i>(actual: 11h)</i>
<h2 style="text-align: center;"><u>Additional Features Added</u></h2>	
<p><i>(actual hours: 110)</i></p>	<ul style="list-style-type: none"> ● Researchers can use built-in Screen Templates to facilitate creating an experiment. (10h) ● Data sets for researchers to incorporate into their experiment can be added. (9h) ● Researchers can easily organize their media through the Media Manager that can be access by clicking on the media link. (12h) ● Correct vs. Incorrect Answers: Researcher can set an expected value of a participant's response. (10h) ● Acknowledgement of Participant Pay can be seen in the participant's My Profile page. (9h) ● For quick tips in creating their experiments, researcher can look at the Within-Site Documentation that can be access by clicking on the "i" icon.s (10h) ● Researchers can get a quickly Within-Site Results Visualization(s) by clicking on the char icon next to the title of an experiment in their My Experiment page. (16h) ● Researcher can contact participant without knowing the participant's PII. (6h) ● Additional Sort Features for Participants (2h) ● Additional Sort Features for Researchers (2h) ● Search Feature for both Experiments and My Experiments pages ● Researcher's experiments are Auto-Save as they are created. ● In addition to CSV, researchers can download a JSON format of their Results Data. (10h) ● Quick Animated Documentation On How To Use the Website Venkman can be found here: https://d3md86mmrid0tk.cloudfront.net/tips (10h) ● Overview Video of Venkman can be found here: http://venkmanscience.com (4h)

Appendix E: Journal Submission

We will submit our paper to “Current Directions in Psychological Science” pictured in Figure 33 below. In 2016 its impact factor was ranked 9 of 128 journals in Psychology, Multidisciplinary².

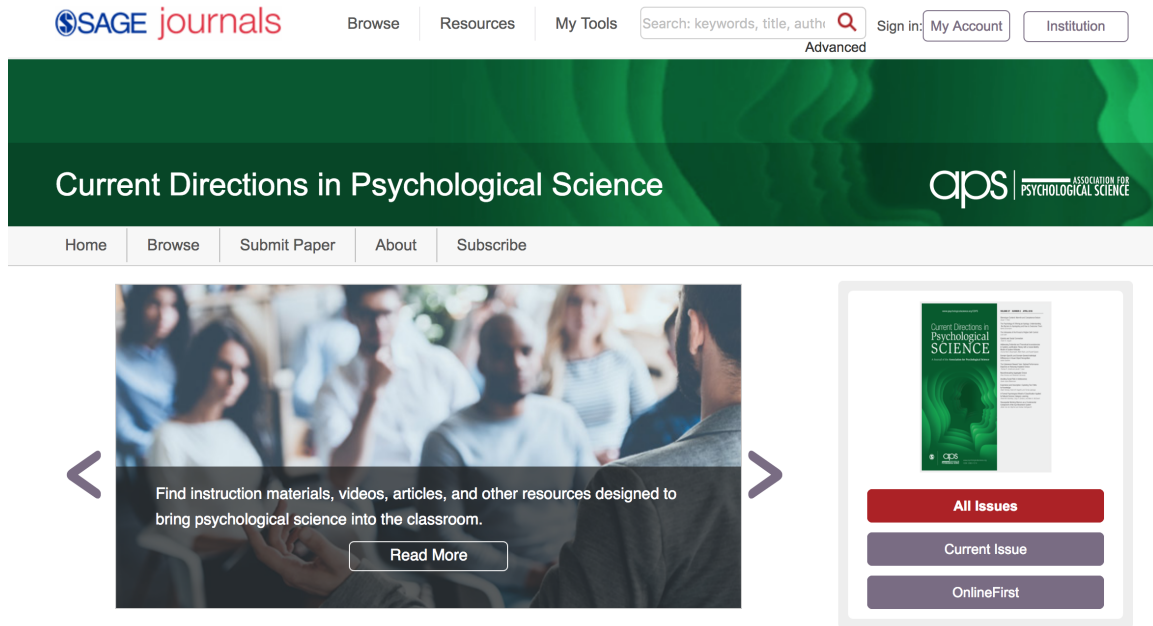


Figure 33: Homepage of “Current Directions in Psychological Science.”

²Engle, R. W. (2018, May 02). Current Directions in Psychological Science. Retrieved April 30, 2018, from <https://us.sagepub.com/en-us/nam/journal/current-directions-psychological-science#description>

Appendix F: Functional Point Accounting Estimates Details

Data Function (ILFs and ELFs)

- **Internal Logical Files (ILFs)** = data that is stored and maintained within the boundary of Venkman

List of Database Tables

The following is a list of the database table names used and their logical groupings in the Venkman application:

- 1) Experiment
 - TestType
- 2) Experiment Response
- 3) Message
- 4) User

- Profile Attribute

Details of the "Experiment" database table

Field	Description	Count as a DET?	Notes
experimentId	Sequential id, system-assigned	No	This is a technical artifact. It is not user-recognizable, and therefore not counted.
researcherId	Sequential id, system-assigned	No	This is a technical artifact. It is not user-recognizable, and therefore not counted.
name	The name a researcher assigns to a given project.	Yes	
status	The status (e.g. "Running", "Not Running", and "Archived") of the experiment	Yes	Note: currently the database can have 2 statuses: 'active' and 'inactive'
definition	Components of database (e.g. dataset, number of screen, maps, etc)	Yes	
description	Table description	Yes	
incentive	Integer representing compensation	Yes	
participationCap	Integer representing number of participations cap	Yes	
participants	Integer representing number of participants	Yes	

rules	IAM rules	Yes	
start	Date the experiment was created	Yes	
time	Record of experiment timing	Yes	
TOTAL DETs:	10		

Details of the "Experiment Result" database table*

Field	Description	Count as a DET?	Notes
experimentId	Sequential id, system-assigned	No	This is a technical artifact. It is not user-recognizable, and therefore not counted.

Details of the "User" database table*

Field	Description	Count as a DET?	Notes
id	Sequential id, system-assigned	No	This is a technical artifact. It is not user-recognizable, and therefore not counted.
email	User supply information	Yes	
profile	User supply information	Yes	
sub	Foreign key	Yes	A foreign key that is required by the user to establish a relationship with another ILF or EIF.
accountBalance	Integer representing compensation (i.e. incentive) balance	Yes	

TOTAL DETs:	5		
-------------	---	--	--

Weights for ILF Complexity Level [Low (L), Average (A), or High (H)]

Complexity	Point
Low	7
Average	10
High	15

To tally up the FP count due to ILFs, we identified four ILFs, with the number of RETs and number of DETs as shown in the table below.

ILF	#RETs	#DETs	Complexity	Function Points
<i>Experiment</i>	2	10	Low	7
<i>Experiment Result</i>	1	0	Low	7
<i>Message</i>	1	0	Low	7
User	2	6	Low	7
			Total Number of FP:	28

External Logical Files (ELFs) = data that Venkman will use/reference, but is data that is not maintained by Venkman

- **Currently: NONE**

Weights for EIF Value Level [Low (L), Average (A), or High (H)]

Value	Point
Low	5
Average	7
High	10

Transactional Function (EIs, EOs, EQs)

1. External Inputs (EI) = process that processes data or control information that comes from outside Venkman boundary

The table below lists the External Inputs (EI) in Venkman. It also lists the number of DETs (Data Element Type) and FTRs (File Type Reference) for each process, and the complexity that results from the number of DETs and FTRs.

Process	#DETs	FTR Names	#FTRs	Resulting Complexity	#Functional Point Weights
Create (i.e. Publish) Experiment <ul style="list-style-type: none"> Status, Name, Description 	5	Experiment	1	Low	3
Edit Experiment <ul style="list-style-type: none"> Button, text, multiple choice, add screens, etc. 	~15	Experiment	1	Average	4
Delete Experiment	1	Experiment	1	Low	3
Clone Experiment	1	Experiment	1	Low	3
Add Entity (e.g. experimental screens, buttons, surveys, etc)	~15	Experiment, Experiment Response	2	High	6
Edit Entity	~15	Experiment, Experiment Response	2	High	6
Delete Entity	~15	Experiment, Experiment Response	2	High	6
Create User <ul style="list-style-type: none"> email, password 	2	User, Profile Attribute	2	Low	3

Delete User	1	User, Profile Attribute	2	Low	3
Edit User Information <ul style="list-style-type: none"> date of birth, gender, marital status, etc 	11	User, Profile Attribute	2	Average	4
				Total Number FP:	<u>41</u>

Ratings for EI Complexity Level [Low (L), Average (A), or High (H)]

File Type Reference (FTR)	Data Element Type (DET)		
	1-4	5-15	Greater than 15
Less than 2	Low	Low	Average
2	Low	Average	High
Greater than 2	Average	High	High

Weights for EI Complexity Level [Low (L), Average (A), or High (H)]

Complexity	Point
Low	3
Average	4
High	6

2. External Outputs (EO) = process that sends data or control information outside Venkman boundary. The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information . The processing logic must contain at least one mathematical formula or calculation, create derived data maintain one or more ILFs or alter the behavior of the system.

3. External Inquiries (EQ) = process that sends data or control information outside Venkman boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF or ELF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered.

The table below lists the External Outputs (EQ) in Venkman.

Process	#DETs	FTR Names	#FTRs	Resulting Complexity	#Functional Points
<i>Export Experiment CSV & JSON Report</i>	<i>At least 5</i>	<i>Experiment, Experiment Response, User</i>	<i>3</i>	<i>High</i>	<i>6</i>
				Total Number FP:	6

Ratings for EQ Complexity Level [Low (L), Average (A), or High (H)]

File Type Reference (FTR)	Data Element Type (DET)		
	1-5	6-19	Greater than 19
Less than 2	Low	Low	Average
2-3	Low	Average	High
Greater than 3	Average	High	High

Weights for EQ Complexity Level [Low (L), Average (A), or High (H)]

Complexity	Point
Low	3
Average	4
High	6

Computing Unadjusted Functional Points:

Function Type	Complexity	Multiplier	Line Item Total	Section Total
ILF	4 Low (i.e. 4 Tables)	x 7 =	28	
	0 Average	x 10 =	0	
	0 High	x 15 =	0	28
ELF	0 Low	x 5 =	0	
	0 Average	x 7 =	0	
	0 High	x 10 =	0	0
EI	5 Low	x 3 =	15	
	2 Average	x 4 =	8	
	3 High	x 6 =	18	41
EO	0 Low	x 4 =	0	
	0 Average	x 5 =	0	
	0 High	x 7 =	0	0
EQ	0 Low	x 3 =	0	
	0 Average	x 4 =	0	
	1 High	x 6 =	6	6
		Unadjusted Function Point Count =		75

The total unadjusted function point count for Venkman (on April 5, 2018) is 75 function points.

Important terms and definitions used in describing the five functions:

- **Data Element Type, or DET:** A data element type is a unique, user recognizable, non-repeated field. This definition applies to both analyses of data functions and transactional functions.
- **Record Element Type, or RET:** A record element type is a user recognizable subgroup of data elements within an Internal Logical File or External Interface File.

Appendix G: Manual Tests Documentation

Here is a list of our requirements, with directions on how to verify the fulfillment of each one.

All Users

As a user, I can create either a researcher account or a participant account.

Cypress option: run Sprint1_01_createAccounts.spec.js

Manually:

1. Visit <http://venkmanscience.com>
2. Follow the steps to create a participant account
3. Login and see a list of experiments to take
4. Visit <http://venkmanscience.com>
5. Follow the steps to create a researcher account
6. Login and see a blank “My Experiments” page

Status as of May 3, 2018: PASS

As a user, I must login before accessing the experimental site.

Cypress option: run Sprint1_02_mustLogin.spec.js

Manually:

1. Make sure you’re not logged in. Then, try visiting various pages that should only appear behind a successful login:
2. <https://d3md86mmrid0tk.cloudfront.net/experiments>
3. <https://d3md86mmrid0tk.cloudfront.net/manage-experiments>
4. <https://d3md86mmrid0tk.cloudfront.net/my-profile>

5. Login to access a researcher's account with an existing, active experiment, get the experiment ID.
6. Logout, and try to access:
7. <https://d3md86mmrid0tk.cloudfront.net/edit-experiment/:experimentID>
8. <https://d3md86mmrid0tk.cloudfront.net/experiments/:experimentID>

Status as of May 3, 2018: PASS

As a user, I can logout of the site to end my session.

Cypress option: run Sprint1_03_canLogout.spec.js

Manually:

1. Login to access a researcher's account with an existing, active experiment, get the experiment ID.
2. Logout
3. Verify that you can no longer access
4. <https://d3md86mmrid0tk.cloudfront.net/experiments>
5. <https://d3md86mmrid0tk.cloudfront.net/manage-experiments>
6. <https://d3md86mmrid0tk.cloudfront.net/my-profile>
7. <https://d3md86mmrid0tk.cloudfront.net/edit-experiment/:experimentID>

Status as of May 3, 2018: PASS

As a user, I can reset my password.

Cypress option: run Sprint1_08_resetPassword.spec.js

Manually:

1. Visit <https://d3md86mmrid0tk.cloudfront.net/login>.
2. Choose "Forgot Your Password?" and follow its steps.
3. Visit <https://d3md86mmrid0tk.cloudfront.net/login>.
4. Login with a current account and password.
5. Choose My Profile.
6. Choose Edit (immediately after the Password label).
7. Again, follow the steps provided via the Change Password screen.

Status as of May 3, 2018: PASS

As a user, I can cancel my account to remove my PII from the system

Cypress option: run Sprint1_09_cancelAccount.spec.js

Manually:

1. Login with a current account and password.
2. Choose My Profile.
3. Choose Delete Account.
4. Choose Permanently Delete My Account.
5. Go to the AWS User Pool service.
6. Verify that that account is no longer listed.

Status as of May 3, 2018: PASS

Researcher Manages Experiments

As a researcher, I have a dashboard which lists my current balance as well as any experiments I have created.

Cypress option: run Sprint1_04_researcherDashboard.spec.js

Manually:

1. Login with a current researcher's account and password.
2. Notice that the first screen you see is the "My Experiments" page, which lists any experiments this researcher has created.
3. Note that the current balance owed to Venkman by the researcher, based on participant response rates, is listed in the upper right-hand corner.

Status as of May 3, 2018: PASS

As a researcher I can create an experiment.

Cypress option: run Sprint1_05_researcherCreateExperiment.spec.js

Manually:

1. Login with a current researcher's account and password.
2. On the "My Experiments" page, note the current list of Experiments.
3. Choose Create Experiment.
4. From the resulting "Edit Experiment" page, choose to go back to "My Experiments".
5. Note the new experiment listed on the "My Experiments" page - it will be named "New Experiment - " followed by today's date and a timestamp.

Status as of May 3, 2018: PASS

As a researcher I can create an experiment title and description which will be displayed on a public bulletin board to all prospective participants.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Choose Publish Details.
4. Change the Name to the desired title.
5. Add the Description you would want to be displayed.
6. Fill out all of the other required metadata fields in order to set the status to Active.
7. Set the status to Active.
8. Log out.
9. Login as any participant.
10. Find that title and description listed on the Experiments page.

Status as of May 3, 2018: PASS

As a researcher I can create instructions for an experiment that are available only to active participants in the experiment.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Use the UI to add text boxes to the first page -- as many as are needed for your desired Instructions. Add the Instruction text by typing into the Element Properties / Text field.
4. Choose Publish Details.

5. Change the Name to a desired title.
6. Add the Description you would want to be displayed.
7. Fill out all of the other required metadata fields in order to set the status to Active.
8. Choose a rule to limit this Experiment's availability only to people who completed their Education level.
9. Set the status to Active.
10. Make note of the Experiment's ID.
11. Log out.
12. Login as any participant who did NOT fill out his/her Education Level.
13. Verify that the Experiment is NOT listed on your Experiments page.
14. Attempt to access the following. You should fail:
15. <https://d3md86mmrid0tk.cloudfront.net/edit-experiment/:experimentID>
16. <https://d3md86mmrid0tk.cloudfront.net/experiments/:experimentID>
17. Login as any participant who DID fill out his/her Education Level.
18. Verify that the Experiment is listed on your Experiments page.
19. Attempt to access the following. You should fail:
20. <https://d3md86mmrid0tk.cloudfront.net/edit-experiment/:experimentID>
21. However, visiting this next link is the same as clicking on the title in your Experiments list in order to take the Experiment:
22. <https://d3md86mmrid0tk.cloudfront.net/experiments/:experimentID>
23. Verify that you are able to see the instructions provided within the Experiment.

Status as of May 3, 2018: PASS

As a researcher, I can edit previously created experiments.

Manually:

1. Login with a current researcher's account and password, choosing some researcher who has already created an experiment, or go ahead and create one per directions above if not.
2. Click on the "pencil" to edit some experiment on the My Experiments page.
3. Change the experiment in some way - it could be the UI, it could be via Publish Details, whatever you want to do.
4. Click on My Experiments.
5. Choose the same experiment.
6. Verify that your changes persisted.

7. If the experiment is active, log in as a participant who should be able to see that experiment on his/her Experiments list.
8. Take the experiment and verify that you receive the updated version.

Status as of May 3, 2018: PASS

As a researcher, I must set an approximate "time required" as well as incentive value to the experiment before it may be published.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Choose Publish Details.
4. Change Status to Active.
5. Note that you have to fill out both "Estimated Time in Minutes" and "Participant Cap" fields before you are able to change the Status to Active.

Status as of May 3, 2018: PASS

As a researcher, I can specify the experiment's language.

Per discussion with client, this requirement was dropped for V1.

As a researcher, I can save drafts of experiments, and then choose when to publish them.

Manually:

1. Login with a current researcher's account and password.
2. Follow the steps to Create and Edit an experiment, without changing its status to Active.
3. Switch to some participant account that should be able to see this experiment once it is published.
4. Verify that the participant does not see this experiment on the Experiments list.
5. Log back in as the researcher.
6. Edit the experiment via the Publish Details page so that its status is Active.
7. Log back in as the participant.
8. Verify that the experiment now shows on the Experiments page.

Status as of May 3, 2018: PASS

As a researcher, I can preview my experiments.

Manually:

1. Login with a current researcher's account and password.
2. If necessary, create an experiment, otherwise click to Edit an existing one.
3. On the Edit Experiment page, choose Preview.
4. Verify that the preview does indeed simulate the participant's experience of the experiment as you have defined it, along all dimensions.

Status as of May 3, 2018: PASS

As a researcher, my dashboard will present experiments categorized by status (active, inactive, or archive).

Manually:

1. Login with a current researcher's account and password.
2. If necessary, create experiments and edit their statuses via Publish Details so that this account contains experiments of every status.
3. Choose your My Experiments page.
4. Click each tab: "All", "Active", "Inactive", and "Archive".
5. As each tab is highlighted, verify that only the expected experiments show on each tab.

Status as of May 3, 2018: PASS

As a researcher I can specify the profile criteria of my target audience.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Choose Publish Details.
4. Click on Add Rule.
5. Choose all of the applicable rules that you would like to apply in order to filter out certain members of the participant population.

6. Publish the experiment.
7. Logout and login as various participants who do or do not fall into the desired population.
8. Verify that those participants the researcher did want to target do see the experiment as an option on their Experiments page, while those participants the researcher wanted to exclude do not.

Status as of May 3, 2018: PASS

As a researcher I can review/define technical requirements for an experiment to ensure commonality of platform and/or consistency of environment.

Per discussion with customer, we deliver the “review” possibility in V1.

Manually:

1. Login with a current researcher’s account and password.
2. Choose an experiment which has some number of participant responses (if none, logout, login as a participant, and take some experiments).
3. Click on the chart icon to review results.
4. Download the CSV.
5. Scroll across the columns - you will see information about each participant’s hardware and software.

Status as of May 3, 2018: PASS

As a researcher, I can duplicate an experiment.

Manually:

1. Login with a current researcher’s account and password.
2. Choose any experiment on your My Experiments page.
3. Click Duplicate, located at the bottom of that experiment’s listing.
4. You are taken to the new copy of the original experiment.
5. Verify that the UI is as you’d defined the original experiment.
6. Verify that the Publish Details are as you’d defined the original experiment - except that Active experiments are copied into an Inactive state.
7. Go to My Experiments.
8. Verify that both the original and the copied experiment are now listed.

Status as of May 3, 2018: PASS

Researcher Defines Experiments

As a researcher, I can create an experiment using a visual UI.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Have fun! You are now in the visual UI to create the experiment.

Status as of May 3, 2018: PASS

As a researcher I can randomize or order the presentation of the tests within an experiment at group/test/screen levels.

Note: our understanding of the vocabulary has since changed, and "test" is redundant.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Add a number of screens (using the provided templates as you see fit).
4. Create groups by clicking on several screens, right-clicking, and choosing Group Selected.
5. Move screens via drag and drop.
6. Move groups via drag and drop.
7. Create nested groups.
8. To randomize the presentation of screens within a group: click on that group's element in the left-hand panel, then click "Randomize children".
9. To randomize the presentation of groups within a parent group: click on the parent group's element in the left-hand panel, then click "Randomize children".
10. Choose Preview to verify that the experiment performs as expected.

Status as of May 3, 2018: PASS

As a researcher, I can specify that groups be presented in a random order.

Duplicate of “As a researcher I can randomize or order the presentation of the tests within an experiment at group / test / screen levels.”

As a researcher I can create named groups of tests within each experiment.

Manually:

1. Login with a current researcher’s account and password.
2. Choose Create Experiment.
3. Add a number of screens (using the provided templates as you see fit).
4. Create various groups by clicking on several screens, right-clicking, and choosing Group Selected.
5. For each group created, click on its element in the left-hand panel.
6. Change its name by typing into the ID field in the right-hand panel.
7. Verify that the groups are now named as desired in the left-hand panel.
8. Make sure that the experiment is in a condition to be published, and publish this experiment.
9. Logout, login as at least one participant, and take the experiment.
10. Logout, login as the researcher who authored this experiment, and choose to download its CSV.
11. Verify that the named groups you created are clearly presented in the CSV report.

Status as of May 3, 2018: PASS

As a researcher, I can assign some incentive level to an introductory set of questions and another to the experiment itself, so that I can "weed out" people who do not meet my requirements.

Per discussion with customer, this requirement was dropped for V1.

As a researcher, I can provide a video clip at the beginning of the experiment.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. While on the "Intro" screen, click on the video icon to add a video.
4. Upload some .mp4 video.
5. Add it to the screen.
6. Create whatever else you want to follow the Intro page in this experiment.
7. Preview the experiment - you should see the video behave as you specified.
8. Publish the experiment.
9. Logout and login as a participant and take the experiment.
10. Verify that the video plays properly for the participant as well.

Status as of May 3, 2018: PASS

As a researcher, I can duplicate any screen within one of my experiments, and assign it to any test within any group within the current experiment.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Within this experiment, create some series of screens and groups in order to create some number of tests.
4. Right-click on any screen and choose "Duplicate".
5. Drag the new screen (marked by the same name as the original, plus "(Copy)", to any desired location within the various groups in the experiment.
6. Go to My Experiments, then return to editing this experiment.
7. Verify that the copied screen is in its last known location.

Status as of May 3, 2018: PASS

As a researcher, I can configure a single value to set the delay between all tests (and/or screens) in an experiment or group.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Within this experiment, create some series of screens and groups in order

- to create some number of tests.
4. Click on the Experiment root group element.
 5. Assign a value to the “Auto-Advance Children (seconds)” field.
 6. Verify that screens within subgroups do not have an auto-advance value applied.
 7. Preview the experiment and verify that all direct child screens of the root group element advance at the specified rate, but no other
 8. Assign a value to some subgroup which has direct child screens.
 9. Verify that each screen within that subgroup now has an “Auto-Advance Children (seconds)” field with that value, plus a “(from parent)” notation.
 10. Change the value for one of those screens to something quite different.
 11. Preview the experiment again, and verify that each screen advances at the rate which is applied to that particular screen.

Status as of May 3, 2018: PASS

As a researcher, I can configure a single value to apply a time limit to all tests within an experiment or group.

For V1: delivered a time limit per experiment

Manually:

1. Login with a current researcher’s account and password.
2. Choose Create Experiment.
3. Click on the Experiment root group element.
4. Input a value into the “Overall Time Limit (minutes)” field.
5. Publish the experiment.
6. Logout, and login as a participant.
7. Take that experiment - but do not interact with it.
8. Verify that the experiment exits out after the specified time.
9. Logout and log back in as the author researcher.
10. Verify that the response was recorded for the experiment.
11. Verify that the results data contains the end event “Overall Timeout”

Status as of May 3, 2018: PASS

Researcher Defines Tests

As a researcher, I can present a stimulus which can be a combination of text, image, audio, and video, with varied possibilities to collect participant response (short answer, multiple choice selection, slider, mouse click, Likert scale, clickable regions).

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. *Tips:*
 - a. While doing the following, use as many screens as you like so that each screen makes sense and will collect one input from the participant.
 - b. Make sure that the participant always has a way to move to the next screen (via the {next} option), and that there is a {next} option on the very last screen, so that the results will be able to be submitted.
 - c. Rename the IDs of the various screens and elements, so that you will be able to trace through the results CSV more easily.
4. Use the text icon to place text on a screen.
5. Use the image icon to place an image on a screen (via the Media Manager, which will pop up).
6. Use the audio icon to embed an audio file in a screen (via the Media Manager, which will pop up).
7. Use the video icon to embed a .mp4 video file in a screen (via the Media Manager, which will pop up).
8. Choose Button to place a button on the screen.
9. Choose Textfield to place a short-answer text field on the screen.
10. Choose Text Area to place a longer-answer text field on a screen.
11. Choose Multiple Choice to place a multiple choice field on a screen.
12. Choose Dropdown to place a dropdown field on a screen.
13. Choose Scale to place a Likert scale on a screen.
14. Choose an image, and then overlay it with at least one Click Area, so that you will be able to capture where a user clicks on a screen.
15. Preview the experiment to be sure that it behaves as you expect.
16. Publish the experiment.
17. Logout, and login as a participant. Take the experiment.
18. Logout, and login as the researcher who authored this experiment.
19. Download the CSV file.
20. Review the results - verify that each participant responses is present.

Status as of May 3, 2018: PASS

Researcher Reviews Results

As a researcher, I will have access to a participant's result data once the participant has completed all tests in the experiment.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Create an experiment, define some filter rules, and publish it.
4. Logout, login as a participant who meets that filter criteria, and take the experiment.
5. Logout, login as the researcher who authored this experiment, and choose to download its CSV.
6. Verify that the participant's responses are clearly presented in the CSV report, as well as the participant's profile data for the attributes used in the filter.

Status as of May 3, 2018: PASS

As a researcher I can review every event's start time.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Create an experiment and publish it.
4. Logout, login as a participant, and take the experiment.
5. Logout, login as the researcher who authored this experiment, and choose to download its CSV.
6. Verify that every participant widget click or text input registered a timestamp.

Status as of May 3, 2018: IN PROGRESS

As a researcher, I expect elapsed timestamps to be accurate to the nearest ms in order to capture differences in short-duration timespans.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment.
3. Create an experiment which employs auto advance as well as response prompts and publish it.
4. Logout, login as a participant, and take the experiment.
5. Logout, login as the researcher who authored this experiment, and choose to download its CSV.
6. Verify that the auto-advance screens loaded at the expected time (in milliseconds) after the prior screen, and that the timestamps and elapsed-time calculations for participant responses were also captured at the millisecond level.

Status as of May 3, 2018: PASS

As a researcher, I can optionally set a participation limit or end date that will close an experiment.

Note: per discussion with customer, a researcher may end an experiment via participation limit or by manually closing it on whatever date the researcher desires, but not via a previously-scheduled end date.

Manually:

1. Login with a current researcher's account and password.
2. Choose Create Experiment. Set its participation cap to 1.
3. Create an experiment and publish it.
4. Logout, login as a participant, and take the experiment.
5. Logout, login as another participant, and note that this experiment is not available via the Experiments list.
6. Logout, login as the researcher who authored the experiment, and note that the experiment is no longer listed as "Active" on the "My Experiments" page, and that the number of responses equals the participant cap.
7. Edit the experiment, setting its participant cap to 50 and changing its status to Active.
8. Logout, login as that second participant, and note that the experiment is

- listed on the Experiments page.
9. Logout, login as the author researcher, and change the experiment's status to Inactive or to Archived.
 10. Logout, login as the second participant again, and see that the experiment is no longer listed on the Experiments page.

Status as of May 3, 2018: PASS

As a researcher, I can download all test results into a CSV file.

Manually:

1. Login with a current researcher's account and password, choosing an account for which hopefully many experiments have been published and participants have taken them. Otherwise, create, publish, and take some experiments.
2. For every experiment in the My Experiments list for which there are 1+ responses, click the chart icon, then choose Download CSV.
3. Verify that each CSV presents that experiment's data-so-far.

Status as of May 3, 2018: PASS

As a researcher, I can view which tests are active, with participation rates.

Manually:

1. Login with a current researcher's account and password.
2. If necessary, create experiments and edit their statuses via Publish Details so that this account contains experiments of every status.
3. Choose your My Experiments page.
4. Click the "Active" tab.
5. Verify that only Active experiments are now listed.
6. Verify that the response rates cited per experiment match your expectations. Choose one to remember.
7. Logout, then login as a participant who hasn't taken that particular experiment yet.
8. Take that experiment.
9. Logout, then login as the author researcher of that experiment.
10. Verify that both the number of participants and the number of completions have increased by 1.

11. Logout, then login as some other participant who hasn't taken the experiment.
12. Click to take the experiment, but then cancel out.
13. Logout, then login as the author researcher again.
14. Verify that the number of participants has increased by 1, but not the number of completions.

Status as of May 3, 2018: PASS

Participant Creates/Accesses Account

As a participant, I can create a profile with an email and password so that I quickly begin participating in experiments.

Manually:

1. Visit <http://venkmanscience.com>
2. Follow the steps to create a participant account
3. Login and see a list of experiments to take on the Experiment screens
4. Click on an experiment and take it.

Status as of May 3, 2018: PASS

As a participant, I can maintain my anonymity (ie, my identity is protected from researchers).

Manually:

1. Login with a current researcher's account and password, choosing an account for which hopefully many experiments have been published and participants have taken them. Otherwise, create, publish, and take some experiments.
2. For every experiment in the My Experiments list for which there are 1+ responses, click the chart icon, then choose Download CSV.
3. Verify that there is not enough PII in the CSV to identify any particular respondent.
4. Browse around the website - verify that there is no researcher access to any list of participants.

Status as of May 3, 2018: PASS

As a participant, I can update my profile with more detailed information about myself in order to fit specific target populations.

Manually:

1. Login with a researcher's account and password.
2. Create an experiment with a filter on it, and publish it.
3. Logout, and then sign up as a new participant.
4. Browse the Experiments lists and note that the newly created experiment is not listed.
5. Click on My Profile and add profile information - being sure to match the filter criteria applied by the researcher in step 2.
6. Return to the Experiments page, and verify that the new experiment is now listed.

Status as of May 3, 2018: PASS

Participant Manages Experiments

As a participant, while I browse the library of experiments for which I qualify, I can sort them by time required, and/or pay.

Manually:

1. Login with a participant's account and password.
2. Interact with the "Sort by" field to choose "Estimated Time".
3. Verify that the experiments are sorted by time.
4. Click the direction arrow, and verify that the experiments are sorted by time, but in the opposite order.
5. Interact with the "Sort by" field to choose "Reward".
6. Verify that the experiments are sorted by incentive amount.
7. Click the direction arrow, and verify that the experiments are sorted by incentive amount, but in the opposite order.

Status as of May 3, 2018: PASS

Participant Takes Experiment

As a participant, I may have to sign boilerplate agreement terms per test, but this will be handled in as efficient a manner as possible.

Manually:

1. Login with a researcher's account and password.
2. Create an experiment.
3. Click the plus sign to add a new screen, and choose the UserAcceptance template. If there are two buttons, proceed, otherwise:
 - a. delete the second screen, and return to the first screen.
 - b. Add the terms via a text box, plus two buttons - one for "I agree" and "I don't agree".
 - c. Assign "{next}" to the "I agree" button and "{finish}" to the "I don't agree" button.
4. Either move the Intro screen to the end, or, if the template had not been sufficient, add another screen to conclude the experiment.
5. Publish the experiment with a cap greater than 1.
6. Logout, and login as one participant.
7. Take the experiment, choosing "I don't agree".
8. Verify that the experiment ends immediately.
9. Logout, and login as another participant.
10. Take the experiment, choosing "I agree".
11. Verify that you access the next screen and can complete the experiment.

Status as of May 3, 2018: PASS

As a participant, if a screen times out, then I must 'ack' to proceed to the next screen.

Manually:

1. Login with a researcher's account and password.
2. Create an experiment with a set of screens in which any screen that has an "auto-advance" set is followed by a screen that requires a manual interaction.

Status as of May 3, 2018: PASS

Non Functional

As a user, I expect all transactions over public internet channels to be encrypted using widely accepted security mechanisms.

Cypress option: run Sprint1_10_secureChannels.spec.js

Manually:

1. Use the website in a variety of ways.
2. Note that the URL always begins with the “https://” protocol.

Status as of May 3, 2018: PASS

As a user, I expect the site to be available to me

Cypress option: run Sprint1_06_sitesAvailable.spec.js

Manually:

1. Use the website in a variety of ways.
2. Note that you were able to do so!

Status as of May 3, 2018: PASS

As a user, I can expect that the data will be secure.

Manually: AWS S3 buckets and DynamoDB databases are private and guaranteed security by Amazon Web Services.

Status as of May 3, 2018: PASS

Additional Features

Screen Templates

1. Login with a researcher's account and password.
2. Create an experiment.
3. Click the plus sign to add a new screen.
4. Choose the UserAcceptance template.
5. Verify that it inserted boilerplate terms and a simple response gathering mechanism.
6. Click the plus sign to add a new screen.
7. Choose the Image loop template.
8. Upload at least 3 images via the Media link.
9. Change the IDs to image1, image2, image3, etc.
10. Preview the experiment.
11. Verify that you saw all images, followed by a multiple choice question.

Status as of May 3, 2018: PASS

Datasets

1. Login with a researcher's account and password.
2. Create an experiment.
3. Create a new screen, and right-click in order to "group selected".
4. Click on the group element in the left panel.
5. Click Use a dataset in the right panel.
6. Choose Create New Data Set.
7. Add some words into the template provided.
8. On the screen which forms this group, choose two widgets in which you can use the variables "{row.column_1}" and "{row.column_2}" to display the text from the dataset. Also make sure there is a way to progress to the next screen.
9. Preview the experiment.
10. Verify that you see the screen presented once for each row in your dataset, and, each time, the two words from one row of the dataset are presented.

Status as of May 3, 2018: PASS

Media Manager

11. Login with a researcher's account and password.
12. Create an experiment.
13. Click the media link.

14. Upload various files.
15. Verify that unsupported formats result in error messages.
16. Verify that supported formats are listed.
17. Verify that each category, Images, Videos, and Audio, list only the relevant files.
18. Verify that no files are not listed within one of those three categories.
19. Verify that every file is able to be added to an experiment from this page.

Status as of May 3, 2018: PASS

Correct vs. Incorrect Answers

1. Login with a researcher's account and password.
2. Create a new experiment.
3. Use the User Acceptance template.
4. Apply a multiple choice widget on a new screen, and note one of the provided options as the "Expected" value.
5. Apply a text field widget on a screen, and provide an "Expected" value.
6. Apply a dropdown widget on a screen, and provide an "Expected" value.
7. Apply a scale widget on a screen, and provide an "Expected" value.
8. Publish the experiment.
9. Logout, and then perform the following steps some number of times:
 - a. Login as a participant.
 - b. Take the experiment.
10. Logout, and login as the author researcher.
11. Visit the results page for that experiment.
12. Review the "Correctness" column on the far right.
13. Verify that the percentage correct for the user acceptance matches the percentage of participants who accepted the user agreement.
14. Verify that the percentages correct for each answer match the expected percentages, based upon the participants' responses.

Status as of May 3, 2018: PASS

Acknowledge Participant Pay

1. Login with a participant's account and password.
2. Visit My Profile.
3. Note the balance listed under the email address and password, as well as the existence (or not) of a list of any experiments already taken, and its

- contents.
4. Visit Experiments.
 5. Take an experiment, noting its title and its incentive amount.
 6. After submitting the experiment, return to My Profile.
 7. Verify that the current balance has been incremented by the amount of that experiment's incentive, and that the list of experiments taken exists and includes the experiment just taken.

Status as of May 3, 2018: PASS

Within-Site Documentation

1. Login with a researcher's account and password.
2. Click the Docs link in the upper left hand corner.
3. Verify that much "how-to" information is provided there.
4. Choose to Create an Experiment.
5. As you add various elements to experiment pages, verify that there are many, many little "i" icons scattered across the left and right panels which launch context-sensitive tooltips.

Status as of May 3, 2018: PASS

Within-Site Results Visualization(s)

1. Login with a researcher's account and password, preferably one who has published experiments that participants have taken.
2. If none of the experiments listed on My Experiments have participant data, ensure that an experiment is available and take it as a participant. Then, return to the My Experiments page.
3. On the My Experiments page, click the chart icon next to the title of an experiment which has received participant data.
4. Verify that the Results screen presents some statistics about the contents of the experiment and the responses chosen by the participants in an accurate and interesting manner.

Status as of May 3, 2018: PASS

Researcher Contacts Participant

1. Login with a researcher's account and password.

2. Create a simple experiment and publish it.
3. Logout, then login with a participant's account and password.
4. Go to My Profile.
5. Choose to Opt-In to Researcher Contact.
6. Verify the email address via the email you receive from AWS.
7. Take an experiment.
8. Logout, and login as the author researcher.
9. Click the chart icon to view the results for that experiment.
10. Download the CSV file.
11. Copy a Message Token.
12. Click Contact Participants.
13. Paste the token into the Participant Message Token field.
14. Add a subject and message, then click Send.
15. Verify that that participant receives the email.

Status as of May 3, 2018: PASS

Additional Sort Features for Participants

1. Login with a participant's account and password.
2. Choose to sort the experiments list by name, and verify that the sort works.
3. Choose to sort the experiments list by start date, and verify that the sort works.

Status as of May 3, 2018: PASS

Additional Sort Features for Researchers

1. Login with a researcher's account and password.
2. Choose to sort the My Experiments experiments list by name, and verify that the sort works.
3. Choose to sort the My Experiments experiments list by start date, and verify that the sort works.
4. Choose to sort the My Experiments experiments list by status, and verify that the sort works.
5. Choose to sort the My Experiments experiments list by number of participants, and verify that the sort works.
6. Choose to sort the My Experiments experiments list by estimated time, and verify that the sort works.

7. Choose to sort the My Experiments experiments list by reward, and verify that the sort works.

Status as of May 3, 2018: PASS

Search Feature for both Experiments and My Experiments pages

1. Login with a researcher's account and password.
2. On the My Experiments page, type into the search box, and verify that the resulting list of experiments contains the search term within each title.
3. On the Experiments page, type into the search box, and verify that the resulting list of experiments contains the search term within each title.

Status as of May 3, 2018: PASS

Auto-Save

1. Login with a researcher's account and password.
2. Choose Create Experiment
3. Add various elements and screens to the experiment.
4. Verify that, upon each change, a message in the upper left appears which says "There are unsaved changes."
5. Verify that, after a small amount of time, a "Saving" progress icon appears.
6. Verify that, until the next change, the message in the upper left appears as "All changes have been saved."

Status as of May 3, 2018: PASS

JSON Download of Results Data

1. Login with a researcher's account and password, preferably one who has published experiments that participants have taken.
2. If none of the experiments listed on My Experiments have participant data, ensure that an experiment is available and take it as a participant. Then, return to the My Experiments page.
3. On the My Experiments page, click the chart icon next to the title of an experiment which has received participant data.
4. On the Experiment Results screen, click Download JSON.
5. Verify that the JSON data matches the experiment and expected participant results.

Status as of May 3, 2018: PASS

Appendix H: Developer README from GitHub

Dev Setup

AWS Access

To be fully engaged, you will need an AWS account on the project tenant. Please contact jeremyclark@g.harvard.edu to request access. Once credentials have been provided to you:

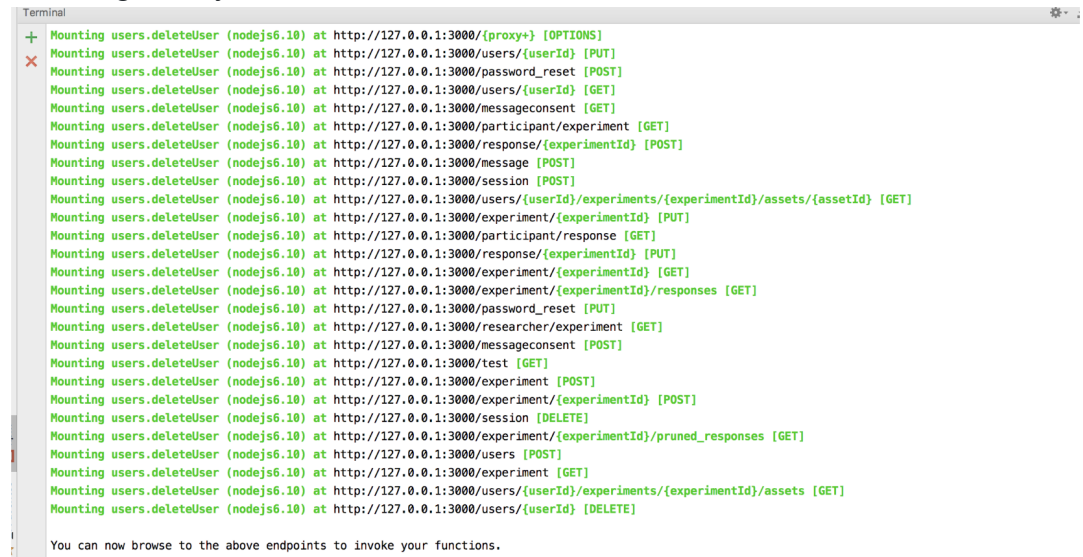
1. Open the URL for the "Console login link" provided in the response.
2. Enter your username and password as noted in the response file. You will be forced to reset your password upon first login.

To configure your AWS CLI toolchain, you can use your access key and secret key as detailed here: <https://docs.aws.amazon.com/cli/latest/userguide/cli-config-files.html>

Once you've cloned this repo, to get things cookin':

1. Install AWS CLI Toolkit
 - <https://docs.aws.amazon.com/cli/latest/userguide/installing.html>
2. Install SAM Local (This requires installing Docker)
 - <https://docs.aws.amazon.com/lambda/latest/dg/sam-cli-requirements.html>
3. To test that everything is running, run:
 - `sam local start-api`
 - This should create a docker image of the project and start it up, making the experiment API endpoints available locally.

- If you see something like this after startup in the console - the API is running locally:



```

Terminal
+ Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/{proxy+} [OPTIONS]
x Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/users/{userId} [PUT]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/password_reset [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/users/{userId} [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/messageconsent [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/participant/experiment [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/response/{experimentId} [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/message [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/session [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/users/{userId}/experiments/{experimentId}/assets/{assetId} [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment/{experimentId} [PUT]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/participant/response [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/response/{experimentId} [PUT]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment/{experimentId} [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment/{experimentId}/responses [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/password_reset [PUT]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/researcher/experiment [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/messageconsent [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/test [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment/{experimentId} [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/session [DELETE]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment/{experimentId}/pruned_responses [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/users [POST]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/experiment [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/users/{userId}/experiments/{experimentId}/assets [GET]
Mounting users.deleteUser (nodejs6.10) at http://127.0.0.1:3000/users/{userId} [DELETE]

You can now browse to the above endpoints to invoke your functions.

```

4. To start the client, do the following:

- `cd client`
- `npm install` (You only need to do this one time ever, or if a new package is added)
- `npm run start` (This will start the server)
- `npm run cypress` (This will start Cypress, You need the previous command as well for it to work)
- `npm run build` (This is just for deploying. It will bundle the files into the /build directory)

Running tests against the locally running API:

Once you have the API running locally, you should be able to run `npm run -s test-backend-dev` to validate that its running and everything is setup properly.

REMEMBER: You'll need to have your AWS credentials set using the developer access_key_id and secret_access_key granted to you as part of your developer setup (see first section of the readme)

Packaging and Deployment

1. To package the 'application' for deployment, run:
 - `sam package --template-file template.yaml --s3-bucket venkman-app --output-template-file packaged.yaml`
2. To deploy the 'application', run:

- `sam deploy --template-file ./packaged.yaml --stack-name Venkman -
-capabilities CAPABILITY_IAM`

A NOTE ABOUT DYNAMO SCHEMA UPDATES

If you will be updating dynamo schemas as part of your updates, you will need to deploy the changes to the AWS tenant using `npm run -s deploy-backend` to use them from the locally running API.

(As of the writing of this document, `sam local` won't dynamically create the dynamoDB tables locally, and we haven't devised a working solution to dynamically inject them into the docker instance that hosts the local API)

The latest client side application is already deployed to AWS at:

<https://d3md86mmrid0tk.cloudfront.net>

The latest version of the API is already deployed to AWS at:

<https://220mtkd7ng.execute-api.us-east-1.amazonaws.com/Stage> If you want to run tests against the stage API, you can execute: `npm run -s test-backend-stage`

Testing

Cypress Tests

Setup

These directions assume that you've already

1. Pulled the latest from GitHub
2. Installed Cypress on your machine
3. Opened Cypress so that you see all of the "Sprint#..." tests listed in the Cypress client.
4. Opened a text editor so that you can tweak all of the files in the client folder.

Configuration

1. Open the `/client/cypress.json` file.
2. Determine which email address you have that will accept "+extra" notation after your usual name and yet still email you (ie, I am usually

- "jlehegaret@gmail.com", but stuff sent to "jlehegaret+extra@gmail.com" still gets to me.)
3. Change the "tester" variable to your email address.
 4. Change the "tester_domain" variable if you need to.
 5. Change the "verify_email" variable to mimic what our app should report as the destination email address at times.

Running the tests

These tests are not as pure as they should be - they do depend on each other more than they should. Here's what you need to know:

Sprint1_01_createAccounts: assumes that you have not already created accounts in the Venkman User Pool with these names. If you have, use AWS to delete the accounts, or label the two "it"s which create the accounts with ".skip".

Sprint1_03, 04, 05, and 09 tests all rely on the user accounts created in Sprint1_01.

Sprint1_08 uses two new user accounts (that append one or two random letters to your usual email name) to test resetting passwords, as you can claim to forget a password only so many times before Cognito locks you out. Please delete these yourself each time you run this test.

Cypress moves fast. When you want to see what's going on, it is very useful to label some contexts as ".skip" or as ".only".

Sometimes, you will need to do some manual steps, such as verifying the email, or entering a verification code. I've put in alert messages so that you'll know what to do when Cypress pauses, and then, after clicking the OK button, you'll also need to "resume" Cypress so that the tests start again.

Reporting Bugs

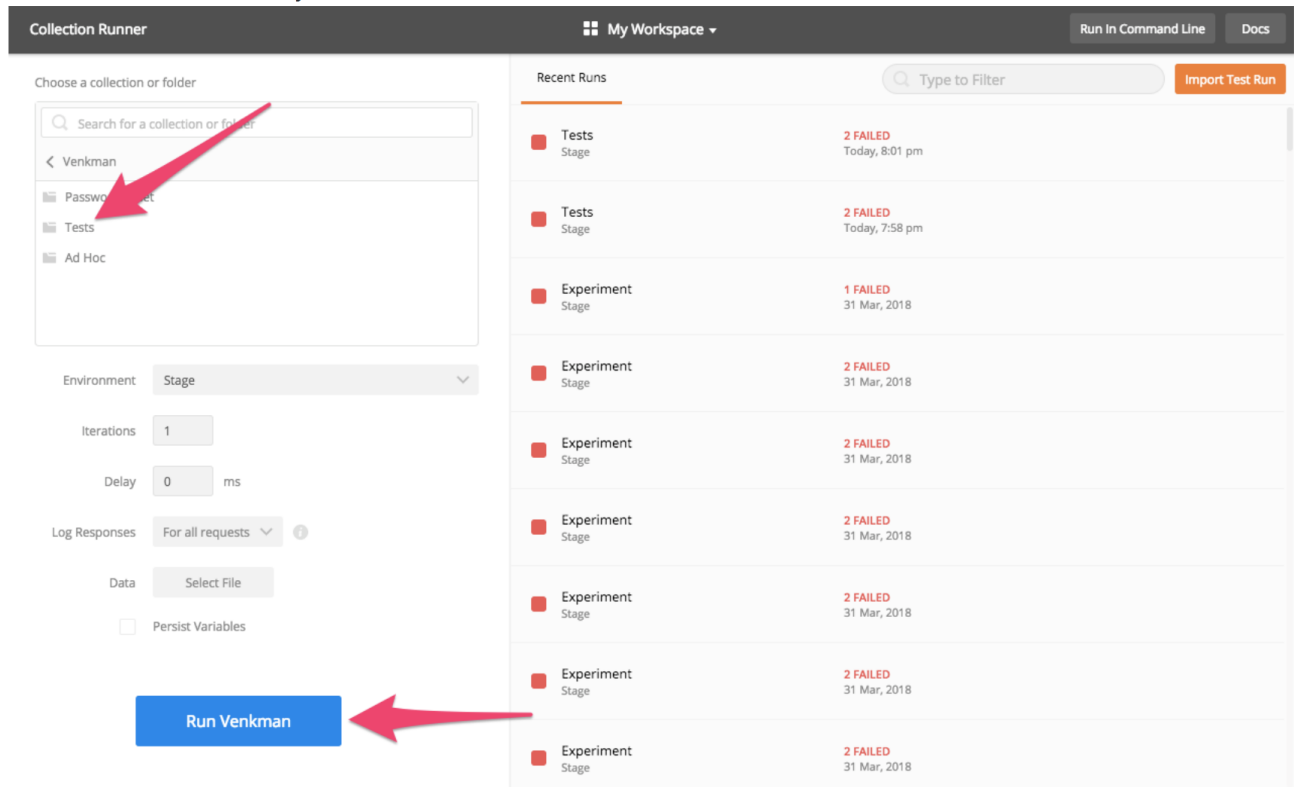
For our class deliverable, we'll first focus on verifying that the "golden path" works. If one of our core requirements is not currently possible, create a GitHub Issue.

Postman Tests for testing the API

To run the postman tests:

1. Install Postman (tested with v6.0.8)

2. Import the Venkman.postman_collection.json, Dev.postman_environment.json and Stage.postman_environment.json files in the test folder in the project.
3. Open the postman runner, and run the tests in the "tests" folder in the Venkman collection:
4. Select the environment of your choice:



If you want to run the postman tests from the CLI:

1. Run `npm install` from the project root. This installs the 'newman' app which can be used to run postman tests from the CLI.
2. Run `npm run -s test-backend-dev` OR `npm run -s test-backend-stage` depending on your target environment for tests.