Homework #3

Analytics System - Part 1: Sessionization and Ingestion

The first step of your analytics project now that we have awareness of the core technologies is to begin to build the analytics system specified in the shared diagram starting first with the distribution of the data collection script, appropriate sessionization, and data intake to the Google Cloud / Firebase environment.

In this step you must provide a way for your tracker.js script to collect information not to localStorage but to send that information to an endpoint in the Firebase/Google Cloud. Firebase Cloud Functions should provide the necessary platform to perform the assignment. You also should use one of the Firebase or Google cloud database services such as Cloud Firestore or Google Cloud SQL to store your data.

⚠️ Note: While you could also consider the Firebase Realtime Database you should understand that probably is not an appropriate idea given the sense of how we might deploy our analytic in a real application. TL;DR - don't use the old Realtime Database offering attempt to make lightweight calls to an endpoint!

Sessionization

For step one you need to sessionize your users, this can be done by handing out the script or it could be done using a call to a special sessionization end point.

Approach 1

`<script src="` `http://analytics.example.web.app/tracker` `"></script>` is included in each of your pages. This is an endpoint that does two things
i) Issues the appropriate user and session cookies
ii) Returns the text of the tracker.js file to run

Approach 2

`<script src="` `http://analytics.example.web.app/tracker.js` `"></script>` in this case the file is static, but upon load it immediately would call an endpoint (anlaytics.example.web.app/sessionize) to get the tracking cookies using fetch or XHR, the cookies might be set with a simple empty image response.

Script off Sessionization

Consider that approach #2 provides the advantage of allowing for something like

```
<noscript> <img src="http://analytics.example.web.app/sessionize?
script=off" height=1 width=1 /> </noscript>
```

HTML

Using the HTTP `referer` header and cookies you still may be able to rudimentary page and session tracking even without JS. Using approach one you may also be able to architect a similar solution with a bit a thought, but it will still require a second end point.

Data Collection

When JavaScript is enabled your tracker script should collect page load, dimensions, and event data as specified in your previous assignments. Your project team should minify the tracker script.

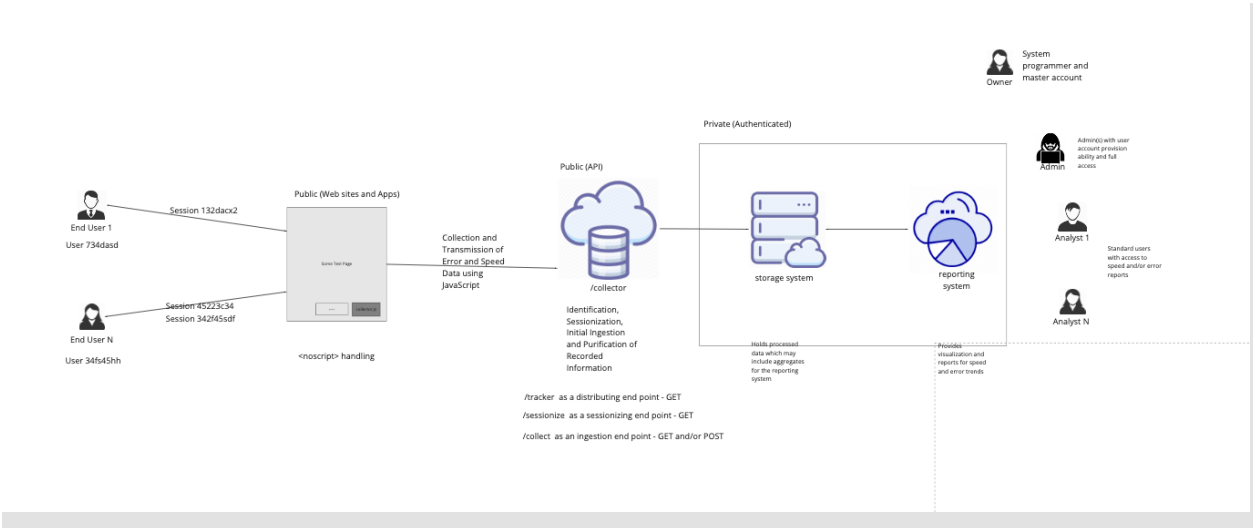
The tracker script should use any or all JS communication methods that are appropriate including: `fetch`, `XMLHTTPRequest`, `sendbeacon`, and `<img>` tag. The payload should be optimized for transmission. Do not send massive payloads, consider preprocessing the data before transmission.

The tracker should submit to a /collect endpoint which will take all properly formatted requests. You must be careful to sanitize inputs and drop bad data in case malicious requests are sent to your endpoints.

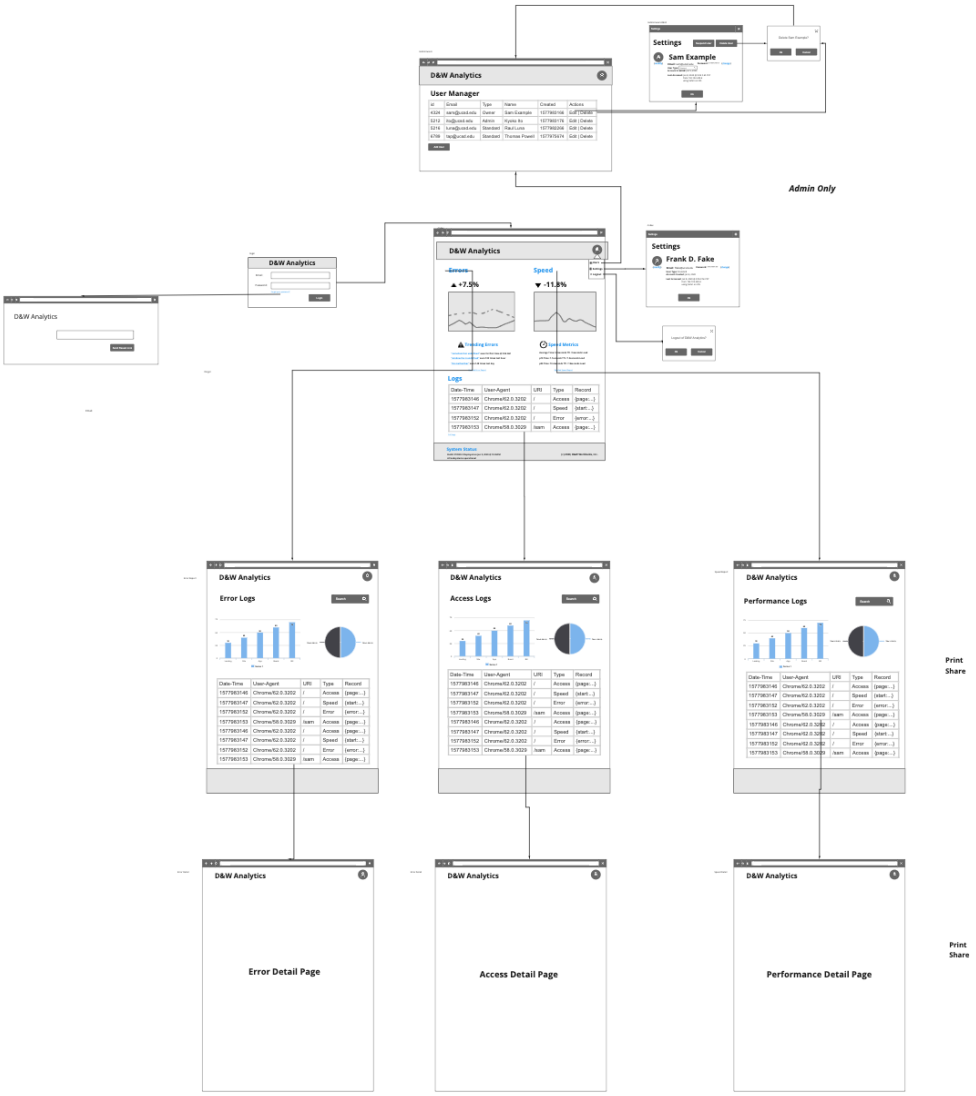
⚠️ Make sure you write a test script to send a larger volume of data (correct and incorrect) to your endpoint to understand if it is operating properly.

Finally, write a simple DB diagnostic page (/showdb) that shows you the contents of your database to see if data has been correctly sent. You may use any JavaScript library/widget necessary. This may give you a chance to explore various data grid libraries or begin to explore a framework you may be interested in trying.

Useful Resources



Rough Overview of System Architecture



Rough Wireframe of Eventual System

Sites for inspiration or Finding Useful Things

JavaScript Error Logging

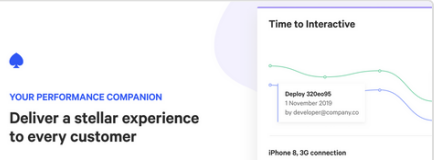
TrackJS catches errors from every user, every browser, and every framework. Errors are automatically tracked, enriched,

https://trackjs.com/

Calibre - Web Performance Monitoring

Addy Osmani, Engineering Manager Never question your performance data. Calibre runs an AI-powered validation

https://calibreapp.com/



https://calibreapp.com/

Rollbar - Error Tracking Software for JavaScript, PH...
Rollbar provides real-time error tracking & debugging tools for developers. JavaScript, PHP, Ruby, Python, Node.js,
https://rollbar.com/

Hotjar | Behavior Analytics Made Easy | Website He...
See how visitors are really using your website, collect user feedback and turn more visitors into customers.
https://www.hotjar.com/

Fun Stuff

😀 Oh and if you need a fun brand to work with for your system, otherwise knock yourself out!

dw-logo.png 107.7KB

Grading Details

Your README-step1.md must include the items below:

- A picture of your collection and sessionization architecture. (3pts)
- Provide a before and after minification/optimization size comparison of your tracker.js Make sure to also describe how you performed the optimization. (3pts)
- Provide a list of the end point(s) and the data that is expected at each. (1pts)
- Overview any data storage choices you made which may include tables (3pts)

Code grading considerations include:

- Working flowing of data with and without JavaScript (6pts - 3pts each)
- Malformed data handing (3pts)
- DB diagnostic page (3pts)
- Clean function, style, and comments. Since there is a small amount of code there is no reason not to make very clean code! (8pts)

Extra Credit

Advanced students may add a fingerprint mechanism to make the mechanism cookieless or have it address flushed cookies. You may modify existing JavaScript fingerprint code you can find in open source, but you must cite where you got it and make it your own by modifying in some meaningful way (variables, etc.) so that it would evade a blacklist it might be on. (up to 8pts extra)

Logistics

- You may work in teams of 3 - provide a list of team members and a short team name for clarity

- Turn in your README with appropriate credentials to the TAs via Gradescope by Friday 2/21 at 11:59PM

- Make sure to fork your project with a new Firebase project and not update it after the steps. Step 2 should have a different endpoint (just duplicate everything) so that we are able to grade your step in isolation.

- Step 2 will be provided sometime before the Step 1 due date. Roughly if you want to explore it early you will have to scaffold your back-end system with basic auth, build a routing system between pages (traditional or SPA style), try a data grid with pagination, searching, etc. and try at least two different visualizations. Step 2 will force you make progress before Step 3 which is the fully polished application