Logging

How would you store your log entries?

To store log entries SQL or NoSQL databases can be used. MongoDB is the NoSQL database I will prefer. MongoDB is a cross platform document-oriented database program. MongoDB stores data in the form of JSON objects. MongoDB is well regarded for high scalability, availability, and performance.

How would you allow users to submit log entries?

Users can submit logs by forms, text documents.

How would you allow them to query log entries?

MongoDB find queries can be used to query log entries. Also, a node.js backend application can be used to query a Mongodb database.

How would you allow them to see their log entries?

Log entries can be queried and be displayed on a web page using front end frameworks like React or can be simply rendered using express handlebars.

What would be your web server?

Web server will be Express which uses Node.js as backend.

Expense Reports

How would you store your expenses?

MongoDb will be used to store expenses. It is a document store database where data can be stored in key value format.

What web server would you choose, and why?

Express will be used as the web server with Node.js as the backend. It is easy to implement REST methods like GET,POST,PUT, DELETE.

How would you handle the emails?

Nodemailer can be installed as a dependency using npm. Nodemailer can be used to send emails.

How would you handle the PDF generation?

We can use PDFKit to handle PDF generation. It can be installed using Node Package Manager(npm).

How are you going to handle all the templating for the web application?

Express Handlebars is a lightweight templating system for Node.js. From the backend we can send data to the frontend in the form of a JSON Object. Values of this object can be displayed on the front-end using Express Handlebars.

A Twitter Streaming Safety Service

Which Twitter API do you use?

There PowerTrack API from twitter is an enterprise level solution which filters real time tweets.

https://developer.twitter.com/en/docs/twitter-api/v1/tweets/filter-realtime/overview/powertrack-api

How would you build this so its expandable to beyond your local precinct?

PowerTrack API has attributes like geolocation which makes it easy to expand beyond local precincts.

What would you do to make sure that this system is constantly stable?

I would incorporate error logging using the middleware to detect errors early. I would write clean and efficient code which reduces loading time.

What would be your web server technology?

I would use Node.js and express to implement REST API based methods. It is easy to code and maintain code using Node.js and Express . It is easy to define and implement code for each URL route.

What databases would you use for triggers?

I would use MongoDB as the database. It is an object-oriented database which stores data in JSON like format. It is a dynamic and scalable database.

For the historical log of tweets?

To avoid repeated calls of API we can store the historical log of tweets in a MongoDB database. We can also cache the tweets using Redis. This can be done to avoid repeated calls to the database.

How would you handle the real time, streaming incident report?

Twitter has a stream API which can handle tweets in real time. Web sockets are used for two-way full duplex communication.

How would you handle storing all the media that you have to store as well?

The tweets along with the media can be stored in a MongoDB database. File Systems can be used to store media, but it is not feasible to store objects in file System.

What web server technology would you use?

I would use Node.js and express to implement REST API based methods. It is easy to code and maintain code using Node.js and Express . It is easy to define and implement code for each URL route.

A Mildly Interesting Mobile Application

How would you handle the geospatial nature of your data?

Geospatial data can be handled by google maps API, here maps API, or open street API. Google Maps is the most preferred API.

How would you store images, both for long term, cheap storage and for short term, fast retrieval?

Amazon AWS and Microsoft Azure are the two most popular public cloud which can be used to store images in the long term. For short term solution Images can be stored in a local database or local storage. Images should be minified and cached to reduce retrieval time and cost. Redis can be used to cache the database.

What would you write your API in?

API will be written on JavaScript.

What would be your database?

I would use MongoDB as the database. MongoDB is a NoSQL database which stores data in JSON like format. Advantages of MongoDB are high scalability, availability, and performance.