Name: Abhay Sharma
CWID: 10459806

Scenario 1: Logging

How would you store your log entries?

I would use Pino library which is available in npm ecosystem to log the information with timestamp, hostname , etc and console it so that we can see it in server console logs and store it in mongoDB with some mandatory field for further reference.

How would you allow users to submit log entries?

I would create a POST endpoint localhost/logs specifically to get the data and store it in DB with some mandatory field such as timestamp, hostname, pid, and etc
We can execute it using POSTMAN or a simple form with UI where user can submit the log entries.

How would you allow them to query log entries?

Using GET Api request a user would be able to pull information with specified key, value pair. User can add timestamp/hostname to get logs depending on those fields.

How would you allow them to see their log entries?

User will be able to see the logs in server's console, also we can make a UI wherein we can pull up the information from the db and provide filter functionality so that user can see logs depending on their requirements.

What would be your web server?

I would use NodeJs as my web server because of npm ecosystem and make it as module so we can use it in multiple projects.

Scenario 2: Expense Reports

How would you store your expenses?

I would use mongoDB as database and store my expenses as documents. Because MongoDb is flexible document schemas we can add more fields in the future with out any problems and run the migration scripts for consistency.

What web server would you choose, and why?

I would choose Node Js because there are bunch of libraries available for generating PDF and e-mailing which are two of the main features that is required in building this application. We can focus more on creating a schema and architecting our application than writing code from scrape to generate PDF and mailing services.

How would you handle the emails?

I would use node mailer package which is available on npm ecosystem. It is open source and easy to integrate module. We can get the email address of user and pass it in node mailer function with pdf to send an email
once the user submitted the expense details.

How would you handle the PDF generation?

For this we can use **pdf-creator-node** package and integrate it in out application. We need to create the template for pdf with all the fields which we can separate it out in a common file and pass in all the fields and get the result, also we need to pass in all the information such as format(A3, A4), orientation, etc for generating PDF. We can create a helper function which when invoke will do all the things and get us the pdf file which then we can send it through email using node mailer.

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

We can use handlebars for templating our web application. It renders on the server side which is good for indexing, also we can separate out the templates and use it at multiple places there by resulting in code reusability.

Scenario 3: A Twitter Streaming Safety Service

Which Twitter API do you use?

I would use **Enterprise** Twitter API as this api provides us with full archive and real time data which we need to build these applications.

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

I would choose Multitier scalable web application architecture which will result in each layer performing only the most essentials functionality thereby improving application performance and scalability. As this application can grow from a local area dataset e.g Hoboken to supporting entire world dataset we would need robust architecture that distributes the processing modules more intelligently, which in this case are performed on one or several separate servers.

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

I would make sure that we implement load balancer so that it can distribute application traffic across number of servers and can increase server capacity depending on the concurrent users so that our application does not get crash because of resource unavailability.

What would be your web server technology?

We can choose between NodeJs/Django, and Ruby on Rails however NodeJS is the best for scalable and asynchronous programming we will go with it. It also has a bunch of packages available so the user can focus more on building the application functionality.

What databases would you use for triggers?

I would use Firebase real time database which is fast and reliable. We can create events with functions and specify what function to trigger depending on the event handlers.

For the historical log of tweets?

The enterprise api of twitter gives us endpoints(Historical PowerTrack API/Search API) where in we can search the full archive of public tweets using advanced filtering tools and get all the data we want removing the need to store any tweets in our own DB.

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

We are using real time firebase DB which stores all the data in JSON object that lets us store and sync all the data between users in Realtime. Once the incident is parsed and updated all the user will get information on real time.

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

I would prefer using Amazon S3 bucket to store all the media with unique ID and get the link from the amazon S3 bucket and then store it in our database. This way the read write operation on our DB will be efficient.

What web server technology would you use?

I would use Node Js because it is scalable and support millions of libraries on the go. We can use multi tier architecture with load balancer to support n numbers of data and concurrent users.

Scenario 4: A Mildly Interesting Mobile Application

How would you handle the geospatial nature of your data?

I would use MongoDB as my database which supports GeoJSON objects to store data based on the location. It is well suited to store geolocation data because it provides points, polygons, and other geo location objects.

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

I would use Amazon S3 buckets to store all the images and get the links from it and then store it in mongoDB for fast retrieval. It is cheap compare to other available technologies.

What would you write your API in?

I would create a web server using express js, node js and integrate it with mongo db using mongoose library and expose all the API using Swagger for reference. On the Mobile application developer can use Flutter/React Native to build front end and then make API call to pull up information and also perform CRUD operations.

What would be your database?

I would choose mongo db as my database because it supports geo json objects to store and retrieve data based on specified location, in addition to that it is scalable and schema flexible.