Scenario 1: Logging

- I would like to store my all logs inside mongo DB by using `Winston. Transports. mongo DB`. I would use Winston for Logging. In Winston, I can create log along with the level, format, and storage. From that I can easily access each api independent. There is total 6 levels such as Fatal, Error, Warn, Info, Debug, Trace, and I can get that each log into mongo DB. On the other hand, we can create one file for all the logs. But it is going to be very hard for file to handle many logs. For that we have to create new files constantly. That's why I would like to use Database for storing all the logs. I would allow users to enter their level for logs to get it to that log page. Just like that I would create page where users can see their log entry as well as query through that. For query through that level must be there. I would build a web app with NodeJS using express and host it to AWS.

Scenario 2: Expense Reports

- I would like to store my all-expense data such as id, user, is Reimbursed, reimbursed By, submitted On, paid On, and amount into mongo DB. I would use Node.js rather than Apache. As I love to use JavaScript than php. Node.js is also offering easy scalability as well as Node.js is way faster than Apache that's why I would use Node.js. For generate a pdf I will use pdf - creator and phantom.js. which are inbuilt modules for npm. We just need to install that. The modules which we cannot get in Apache. I would email the pdf using one of the modules namely node mailer. It's very easy to generate the pdf and send it to email in node.js. I would use handlebars for templating. It will allow me to add data into row data and render it to the any web browser. All the properties of expense I will mention in one table and that table I will send to user who submitted the expense by email.

Scenario 3: A Twitter Streaming Safety Service

- I would like to use api called ‘Account Activity API’. From that we can access tweets from nearby. This is a method that takes an array as coordinates and a radius as input and gives tweets from that region. To make sure that the system is working or not, I would use logging system as I discussed in my first question, and I would keep track of every event happening on the website. If something unusual happens, I will act on it. The script will continuously check the log file and if something is going wrong, it will send me an email with the event log that was gone wrong. So, I will know immediately when it is not stable. As always, I would choose NodeJS for web server. I would use MongoDB to store logs of tweets. Account Activity API provides a way to subscribe for streaming tweets. That will give tweets in real time. When someone posts a new tweet, the API will trigger an event and send the new tweet data as JSON format. I will check then if the tweet contains keywords such as fight, drugs, SMUHS, etc. If it’s having such words, I will log that tweet into DB and store the image in a folder on the server.

Scenario 4: Mildly Interesting Mobile Application

- I would like to work with Node.js. By using Node.js I am going to generate URL and I will store all the events into mongo DB. Since it's a mobile app, I am going to take a picture with geo coordinates as a body parameter for the upload path. Once I get the picture I will upload the image on the server, get the URL of the image and add it to the mongo DB containing the location. For retrieval We will also create an API endpoint to easily retrieve interesting images of the user's geolocation. Use the user's location coordinates as the route's body parameter. Then add a radius, create an array of coordinates to form an imaginary shape, and get all images captured in that area. Another way to work with spatial data is with node modules. There is one module available it takes the radius and converts and returns it to a circle with geographic coordinates. With this circle you can get all images from this area. My database going to be mongo DB.