Lab 5

Scenario 1: Logging

In this scenario, you are tasked with creating a logging server for any number of other arbitrary pieces of technologies. Your logs should have some common fields, but support any number of customizable fields for an individual log entry. You should be able to effectively query them based on any of these fields.

How would you store your log entries?

I would store the logs in MongoDB as it is easy to fetch data and goes hand in hand with node.js.

How would you allow users to submit log entries?

I will create a form-based application using handlebars having fixed input fields and having an option to add customizable inputs.

How would you allow them to query log entries?

For querying log entries I would create a log list page with user authentication, having different permissions for different users. It will contain a list of log entries with drop down filter option. In the drop-down filter option there will be an option of “other” in which the user can filter with customizable field. It will also contain a search bar with filters to directly search a log.

What would be your web server?

I would use node.js as a web server as it is a technology I have experience in.

Scenario 2: Expense Reports

In this scenario, you are tasked with making an expense reporting web application.Users should be able to submit expenses, which are always of the same data structure: id, user, isReimbursed, reimbursedBy, submittedOn, paidOn, and amount.When an expense is reimbursed you will generate a PDF and email it to the user who submitted the expense.

How would you store your expenses?

I would store the expenses in mongoDb as it is a document based database and easy to access.

What web server would you choose, and why?

I would choose node.js as a web server as it is easy to fetch data through mongoDB using the same scripting language.

How would you handle the emails?

I would use NodeMailer as I am using node.js as server-side technology.

How would you handle the PDF generation?

I would use PDFKit as I am using node.js as server-side technology.

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

I would use handlebars for templating to stick to the same stack.

Scenario 3: A Twitter Streaming Safety Service

In this scenario, you are tasked with creating a service for your local Police Department that keeps track of Tweets within your area and scans for keywords to trigger an investigation.

This application comes with several parts:

An online website to CRUD combinations of keywords to add to your trigger. For example, it would alert when a tweet contains the words (fight or drugs) AND (SmallTown USA HS or SMUHS).

An email alerting system to alert different officers depending on the contents of the Tweet, who tweeted it, etc.

A text alert system to inform officers for critical triggers (triggers that meet a combination that is marked as extremely important to note).

A historical database to view possible incidents (tweets that triggered an alert) and to mark its investigation status.

A historical log of all tweets to retroactively search through.

A streaming, online incident report. This would allow you to see tweets as they are parsed and see their threat level. This updates in real time.

A long term storage of all the media used by any tweets in your area (pictures, snapshots of the URL, etc).

Which Twitter API do you use?

I would use twitter standard search api as its documentation is widely available.

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

I would use Amazon web services as they are scalable and cost effective for government agencies.

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

I would use Amazon web services as the system is stable and documentation and helpline is extensive.

What would be your web server technology?

I would use node.js as the web server technology as I have experience in it.

What databases would you use for triggers?

I would use AWS RDS as it goes hand in hand with the other technologies used.

For the historical log of tweets?

I would use amazon cloudwatch logs as it can be used easily with the other technologies used.

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

I would use amazon Kinesis hand in hand with the other technologies used

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

I would use amazon elemental mediaStore hand in hand with the other technologies used

What web server technology would you use?

I would use node.js web server technology

Scenario 4: A Mildly Interesting Mobile Application

In this scenario, you are tasked with creating the web server side for a mobile application where people take pictures of mildly interesting things and upload them. The mobile application allows users to see mildly interesting pictures in their geographical location.Users must have an account to use this service. Your backend will effectively amount to an API and a storage solution for CRUD users, CRUD 'interesting events', as well as an administrative dashboard for managing content.

How would you handle the geospatial nature of your data?

I would handle the geospatial nature of the data with geotagging and local seo technologies for location based search enhancement.

Lab 5

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

For short term usage I would use google drive or drop box due to quick set up and wide availability.

For long term usage I will use Aws elemental mediastore as it comes with variety of features and is good long term solution for this kind of application.

What would you write your API in?

I would write the API in node.js as it works seamlessly with mongoDB.

What would be your database?

I would use mongoDB as it is easy to fetch data from to store the link to the images.