Community News Bulletin

Build your Platform using AWS Tech!

Uttara Shekar

Melonia Mendonca

Grace Hopper Conference, 2020

Contents

[Abstract 2](#_Toc51175254)

[What is a web service? 2](#_Toc51175255)

[What service are we building? 2](#_Toc51175256)

[Requirements 3](#_Toc51175257)

[Use cases 3](#_Toc51175258)

[Functional Requirements 3](#_Toc51175259)

[API requirements based on User stories and Functional Requirements 3](#_Toc51175260)

[Architecture 4](#_Toc51175261)

[Backend Technologies (AWS) 4](#_Toc51175262)

[Amazon S3 4](#_Toc51175263)

[Amazon DynamoDB 5](#_Toc51175264)

[AWS Lambda 6](#_Toc51175265)

[Amazon API Gateway 6](#_Toc51175266)

[AWS Amplify 7](#_Toc51175267)

[Front-end Technologies 7](#_Toc51175268)

[Babel 8](#_Toc51175269)

[Webpack 8](#_Toc51175270)

[ReactJS 8](#_Toc51175271)

[Instructions for setup 8](#_Toc51175272)

# **Abstract**

The software industry is increasingly using more and more cloud-based services to take advantage of benefits like price-per-usage, not having to manage their own servers, not having to handle different volumes of customer traffic, or ensure availability. AWS - the leading platform for cloud-based services offers a wide range of technologies that are used and recommended by leading software companies across the globe.   
  
This workshop, during Grace Hopper Conference, 2020, aims to offer an opportunity to work with some of the most popularly used AWS technologies, as well as get a high-level picture of good serverless architecture models. The objective of this workshop is to teach some basic AWS skills with proper guidance on how to get a deeper understanding of serverless architecture on AWS.

Since acquiring an in-depth knowledge of popular AWS technologies within the time allotted for this workshop is challenging, this handbook aims at providing a simple, easy-to-understand manual that will guide participants through the process of building a service. This handbook facilitates participants to take home information regarding exercises performed as part of the workshop, as well as has information for setup before diving into the building of services. The handbook also includes the location of downloadable code for participants to use during the workshop, and refer to later, after the workshop is completed.

# **What is a web service?**

A web service is a piece of software that enables machines to interact over a network. Cloud provided solutions enable users to quickly create web services of their own without worrying about managing costly infrastructure.

Amazon Web Services (AWS) provides on-demand cloud-based platforms and APIs for individuals as well as companies to use on a pay as you go basis. AWS offers over 175 highly available, scalable, reliable and flexible software solutions ranging from database storage and computing, IoT, machine learning and management tools for the benefit of their customers.

# **What service are we building?**

In light of the existing global pandemic due to the novel Coronavirus disease, COVID-19, being more in tune with the news within our communities has become increasingly vital. We are always striving to know more about the health of the people in our communities in order to better understand if there is a need to quarantine, or to help people in need. In addition to health, local businesses have been suffering in these difficult times. A lot of these struggling business might exist within your neighborhood with the need to advertise the work they are doing. People are often unaware of such businesses when they are right around the corner in many cases.

In this workshop, we will create a community blog that focuses on articles about local events, upcoming businesses, and helpful information relevant to the community. The blog will be backed by well-known AWS technologies like Amazon S3 - to store larger data like the body of the new articles, files etc., Amazon DynamoDB – to store and retrieve data quickly etc. and AWS Lambda - which provides the 'service' component in this architecture.

# **Requirements**

## Use cases

### U1: Write new blog articles

### As the editor of the Community News Bulletin, editors must be able to add articles to the news bulletin.

### U2: View articles listed as part of the community blog

As a user of Community News Bulletin, viewers must be able to open the website to view stories about the pandemic, other social issues and know more about local businesses.

### U3: Read specific articles within the list of articles

As a user of the Community News Bulletin, viewers must be able to click on an article that interests them and read that article.

U4: Add comments and like articles  
As a reader of the Community News Bulletin, readers should be able to like and comment on articles.

## Functional Requirements

|  |  |  |
| --- | --- | --- |
| Requirement | Use Case | Priority |
| Add stories or articles to the blog | U1 | Workshop! |
| View a list of articles or stories created | U2 | Workshop! |
| Click on a single article and read it | U3 | Take home exercise |
| Add comments and/or likes | U4 | Take home exercise |

## API requirements based on User stories and Functional Requirements

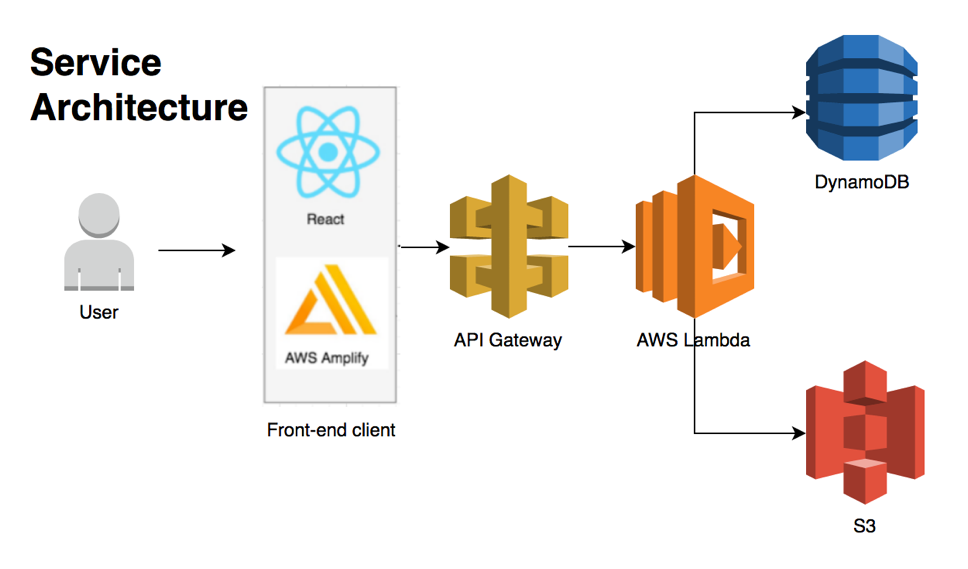
|  |  |  |  |
| --- | --- | --- | --- |
| API | Request | Response | Comment |
| createStory | title, content, image | none | API that allows you to add news articles to news bulletin |
| readStories | none | List of stories | API that lists all the available stories along with their titles and identifiers |
| readStory | id | content, image | API that reads an existing story given its identifier |
| addLikeOrComment | id, like, comment | none | API that allows you to like or comment on a given story |

# Architecture

In order to build a community bulletin that allows you to create, read and list existing stories, we have the following need, that is satisfied by the respective AWS services:

* Storage for the article content – [Amazon S3](https://aws.amazon.com/s3/)
* Storage for the article metadata/information – [Amazon DynamoDB](https://aws.amazon.com/dynamodb/)
* Computing platform that runs code in response to events – [AWS Lambda](https://aws.amazon.com/lambda/)
* An API management tool that “converts” client-side requests to code understandable by the back-end services – [Amazon API Gateway](https://aws.amazon.com/api-gateway/)

These services will communicate with each other in the following manner:



# Backend Technologies (AWS)

The following sections will go over the services mentioned before and their design to be used in the Community blog.

## Amazon S3

Amazon S3 is a simple web service that allows you to store and retrieve any amount of data, from anywhere in the web. It is an object storage service that offers industry leading scalability, data availability, security and performance. It allows users to store any amount of data securely, for a range of use cases not limited to archiving data, back-up and restoring data, mobile applications, big data analytics and, in our case, websites.

**When would I use it?**

Amazon S3 allows you to store infinite amount of data in a “bucket” which is mainly the fundamental container for your objects. Amazon S3 can be used in the following situations:

* when you need to store large data like videos, large file/libraries, images etc.
* when you want to access data in the cloud frequently.
* when you need encrypted, or secure file uploads, and downloads

**How are we using it in this workshop?**

Because of its ease of use and manageability features, we will be using Amazon S3 to store the content and images of our articles.

**Bucket design**

* S3 Bucket names are unique, so we will be using the prefix “news-bulletin” followed by your first name and birthdate - MMDDYYYY - Example: news-bulletin-usa-07041776
* For every new story, create a Story ID = GUID. This will be treated as the name of the file.

For example, https://s3.amazonaws.com/news-bulletin-usa-07041776/1234-5678-1234-5678.txt will store the following blog content

Good News!

No new cases in the community for the past 4 weeks!

|  |  |
| --- | --- |
| S3 bucket name | Prefix |
| news-bulletin-usa-07041776 | 1234-5678-1234-5678 |

## Amazon DynamoDB

NoSQL databases use a variety of data models for accessing and managing data. These types of databases are optimized specifically for applications that require large data volume, low latency, and flexible data models, which are achieved by relaxing some of the data consistency restrictions of other database types.

Amazon DynamoDB is a fully managed NoSQL database that allows you to provide complex query models to access data.

**When would I use it?**

Amazon DynamoDB can be used as an alternative to traditional SQL databases

* For storage of unstructured data that can be accessed using specific keys
* When development of the service is fast paced such that the structure to the data can be added eventually
* When there is a requirement for high-performance reads and writes, with low latency and high durability

**How are we using it in this workshop?**

We will be using this NoSQL service to store information regarding our blog articles. A variety of information can be stored as metadata within the DynamoDB table. Information regarding an article can range from who created the article, when it was created, what is the title, to information like how many likes the article has, how many comments, etc.

**Table Design**

We will be designing a table with simple metadata as follows:

* Use the Story ID created in = GUID;
* Update the DDB with
  + Primary Key = 1234-5678-1234-5678,
  + Title - GOOD NEWS!
  + Content\_url - <https://s3.amazonaws.com/news-bulletin-usa-07041776/1234-5678-1234-5678.txt>
  + Num\_likes - 0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table name | Partition Key (PK) | Attributes | |  |
| Stories | **ID** | **Title** | **Content\_url** | **Num\_likes** |
| 1234-5678-1234-5678 | GOOD NEWS! | https://s3.amazonaws.com/news-bulletin-usa-07041776/1234-5678-1234-5678.txt | 0 |

## AWS Lambda

AWS Lambda is an event-driven, serverless computing platform provided by Amazon as a part of Amazon Web Services. It is a computing service that runs code in response to events and automatically manages the computing resources required by that code.

**When would I use it?**

AWS Lambdas:

* are light-weight, serverless applications providing as a perfect compute solution without having to worry about managing the servers or containers
* are event driven, meaning they are triggered based on events
* auto-scale depending on the usage

Other compute solutions offered by AWS include, but are not restricted to [Amazon EC2](https://aws.amazon.com/ec2/), [Amazon ECS](https://aws.amazon.com/ecs/), [Amazon EKS](https://aws.amazon.com/eks/), [AWS Outposts](https://aws.amazon.com/outposts/) etc.

**How are we using it in this workshop?**

In our case, the serverless compute mechanism will interact with the other services – Amazon S3 and Amazon DynamoDB, to either add information or to get the required information.

**Function design**

Each API created for this service will be backed by an AWS Lambda Function. We will call these *APIname*-dev.

## Amazon API Gateway

Amazon API Gateway is an API management tool that sits between the client and backend services. It accepts all the API requests made to the service, translates them and appropriately transfers them to the service that will be able to react to the requests made. It is also the service that is responsible to return the response sent by the back-end service using a contract that is understandable by the user. API developers can create APIs that access AWS or other web services as well as data stored in the AWS Cloud using Amazon API Gateway.

**When would I use it?**

If you are creating a service, you are going to use APIs, and you will need a solution to manage the reliable processing of those APIs. Amazon API Gateway provides:

* Authorization and access control
* Throttling
* Monitoring
* Traffic management
* API versioning and its management

**How are we using it in this workshop?**

For the purpose of this workshop, we will be using RESTful API. API Gateway creates RESTful APIs that:

* Are HTTP-based.
* Enable stateless client-server communication.
* Implement standard HTTP methods such as GET, POST, PUT, PATCH, and DELETE.

## AWS Amplify

AWS Amplify is a set of powerful tools and services that enable developers to build secure, scalable full stack applications, powered by AWS. With the help of [AWS CloudFormation](https://aws.amazon.com/cloudformation/), this tool helps build back-end layers like Amazon DynamoDB tables, Amazon S3 buckets, AWS Lambda functions and layers, etc. It also helps setup framework for front-end code. With AWS Amplify, it’s easy to create custom onboarding flows, develop voice-enabled experiences, build AI-powered real-time feeds, launch targeted campaigns, and more.

**When would I use it?**

With its user-friendly and simple usage, use AWS Amplify to build the bare-bones of your service architecture.

**How are we using it in this workshop?**

We will be using Amplify in this workshop to build the bare bones of our service dependencies including S3 buckets, DynamoDB tables and Lambda functions. We also use this tool to setup the React JS framework for the front-end application.

# Front-end Technologies

Along with the AWS Services mentioned in the Service design, we have used a few other technologies in this workshop. This section goes over those.

## Babel

<https://babeljs.io/>

Babel is a JavaScript compiler that can translate programming languages into JavaScript. It allows us to use the newest features of JavaScript version ES6. We use Babel in this project as a mechanism for React to translate JSX files into JavaScript.

## Webpack

<https://webpack.js.org/>

Webpack is a bundler for JavaScript files. When webpack processes our React front-end application, it internally builds a dependency graph which maps every module the project needs and generates one single bundle.

## ReactJS

<https://reactjs.org/>

React JS is a javaScript library for building user interfaces that was created by Facebook. It works as a neat tool for building reusable UI Components. We use React JS in our project as a tool to set up our News Bulletin front-end.

# Instructions for setup

In order to provide faster access to scripts and CLI, the working instructions for setup are added to <https://github.com/uttarashekar/ghc_2020_aws_workshop.git>. Please navigate to this link for further instructions.