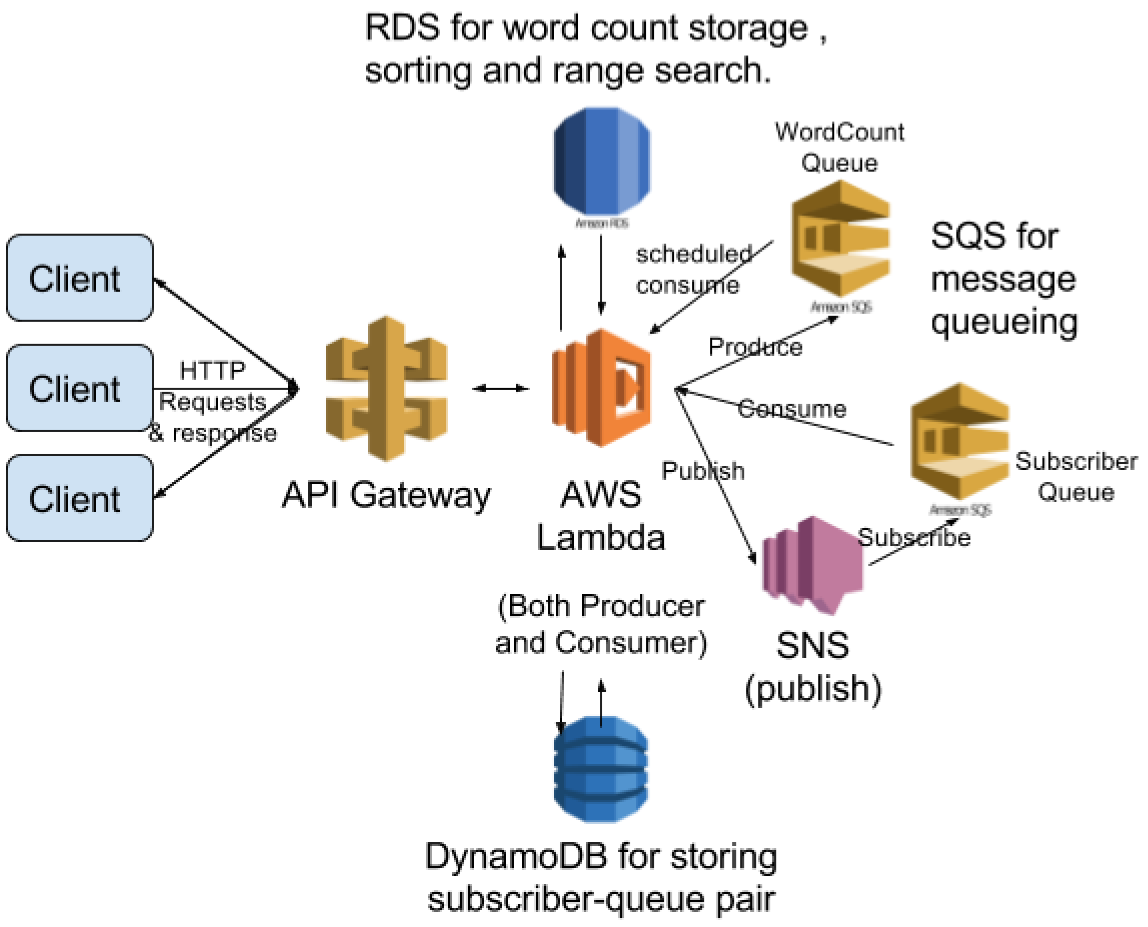
BSDS Assignment 3 Design Doc

Alec Lindberg, Yang Song

Design Overview

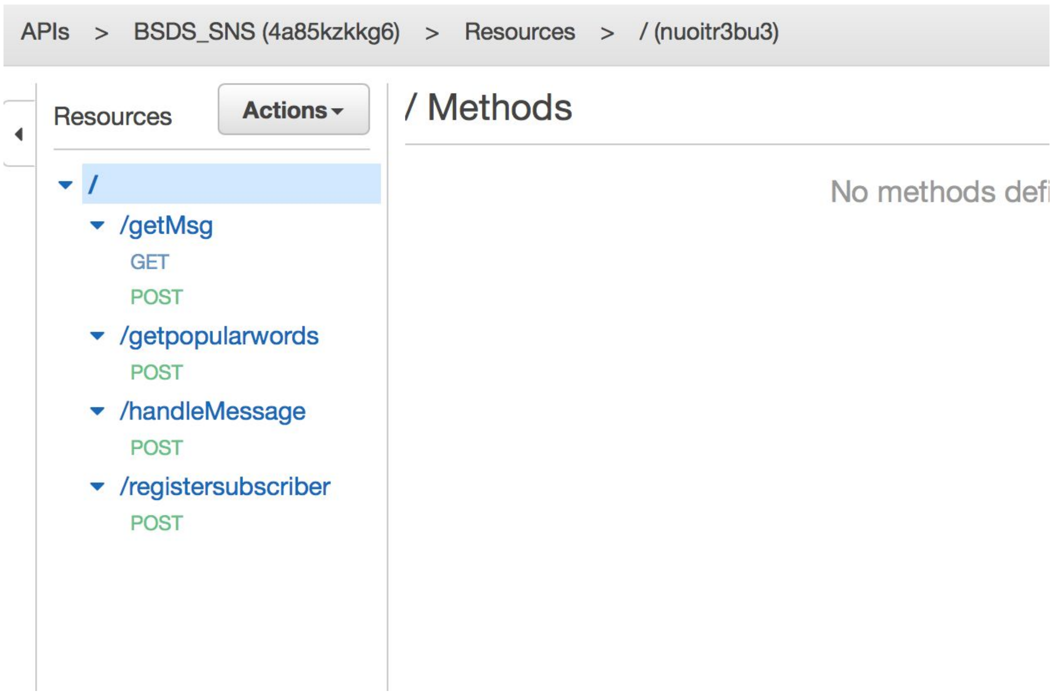
For assignment 3, after doing some investigation and research, we finally decided to go with a totally "Serverless" approach using AWS Lambda, API Gateway, SNS, SQS, DynamoDB and RDS. API Gateway will expose our APIs and handle incoming HTTP requests. The major business logic resides in Lambda, it will process requests, produce and consume messages from SQS and also talk to database. SNS (Simple Notification Service) is for message delivery, and SQS (Simple Queue Service) is for message queueing. DynamoDB stores the key-value pair for subscriber-topic pair, and RDS (Relational Database Service) will store word count. The graph below shows the overall architecture of our design:



Major Components

1.AWS API Gateway

Amazon API Gateway is a fully managed service that makes it easy for developers to publish, maintain, monitor, and secure APIs at any scale. API Gateway provides a HTTPS API front end for clients, and clients can integrate with the APIs using standard HTTPS requests.


APIs > BSDS_SNS (4a85kzkkg6) > Resources > / (nuoitr3bu3)

Resources

/
/getMsg
GET
POST
/getpopularwords
POST
/handleMessage
POST
/registersubscriber
POST

/ Methods

No methods defi

2.AWS Lambda

AWS Lambda is a zero-administration compute platform for back-end web developers that runs your code for you in the AWS cloud and provides you with a fine-grained pricing structure.

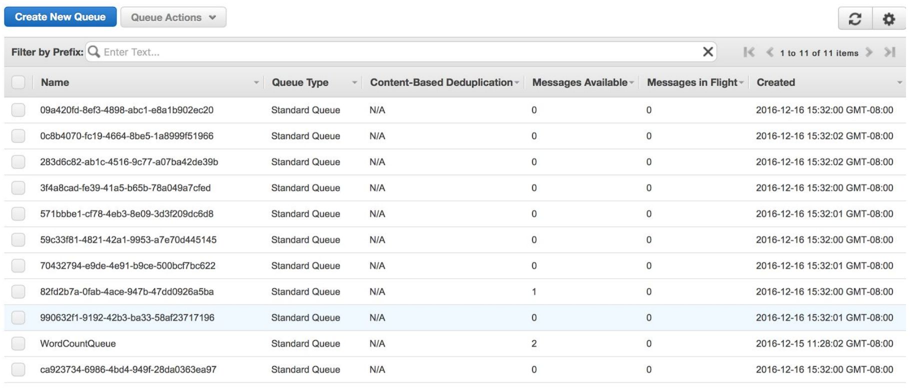
For our design, we have 5 main Lambda functions that handles the backend logic: addSubscriber, handleMessage, getMessage, schduledWordCountQueueConsumer and getPopularTerms. The source code on these functions are also inside the repo.

In general, For each new subscriber client, we create a new SNS topic with the given topic and also a new SQS queue. The SQS queue will subscribe to the SNS topic. When a publisher client publishes a message to SNS (in handleMessage), SQS will enqueue that new message, and meanwhile the message will also be broken down into words, and each word will be enqueued into the WordCountQueue. Then when the subscriber client who subscribes to that topic, the getMessage function will poll the corresponding SQS queue and respond with the message. For the WordCountQueue, a schduledWordCountQueueConsumer will be invoked every 1 minute to read the messages and update the records in RDS accordingly. getPopularTerms is just a SQL query to the RDS instance to get the top n popular terms.

3.Amazon Simple Queue Service (SQS)

Amazon Simple Queue Service (Amazon SQS) is a messaging queue service that handles message or workflows between other components in a system.

Each subscriber will have its SQS queue with a specific topic. Also we maintain a WordCountQueue for updating the word count table in the relational database.



Name	Queue Type	Content-Based Deduplication	Messages Available	Messages in Flight	Created
09a420fd-8ef3-4898-abc1-e8a1b902ec20	Standard Queue	N/A	0	0	2016-12-16 15:32:00 GMT-08:00
0c8b4070-fc19-4664-8be5-1a8999f51966	Standard Queue	N/A	0	0	2016-12-16 15:32:02 GMT-08:00
283d6c82-ab1c-4516-9c77-a07ba42de39b	Standard Queue	N/A	0	0	2016-12-16 15:32:02 GMT-08:00
3f4a8cad-fe39-41a5-b65b-78a049a7cfed	Standard Queue	N/A	0	0	2016-12-16 15:32:00 GMT-08:00
571bbbe1-cf78-4eb3-8e09-3d3f209dc6d8	Standard Queue	N/A	0	0	2016-12-16 15:32:01 GMT-08:00
59c33f81-4821-42a1-9953-a7e70d445145	Standard Queue	N/A	0	0	2016-12-16 15:32:00 GMT-08:00
70432794-e9de-4e91-b9ce-500bcf7bc622	Standard Queue	N/A	0	0	2016-12-16 15:32:01 GMT-08:00
82fd2b7a-0fab-4ace-947b-47dd0926a5ba	Standard Queue	N/A	1	0	2016-12-16 15:32:00 GMT-08:00
990632f1-9192-42b3-ba33-58af23717196	Standard Queue	N/A	0	0	2016-12-16 15:32:01 GMT-08:00
WordCountQueue	Standard Queue	N/A	2	0	2016-12-15 11:28:02 GMT-08:00
ca923734-6986-4bd4-949f-28da0363ea97	Standard Queue	N/A	0	0	2016-12-16 15:32:00 GMT-08:00

4.Amazon SNS

Amazon Simple Notification Service (Amazon SNS) is a fast, flexible, fully managed push notification service that lets you send individual messages or to fan-out messages to large numbers of recipients.

We're using SNS for pushing messages to SQS queue, since SQS is a polling queue.

5.AWS Relational Database

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

Since the “Get Top N” operation is not naturally supported by a key-value Nosql database, we have to use a relational database for the word count functionality. We're using a MySQL instance with a large storage across multiple availability zones, so the throughput of database should not be a bottleneck for our service.

6. Amazon DynamoDB

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. Amazon DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity specified by the customer and the amount of data stored, while maintaining consistent and fast performance.

At first we decided to store all the publisher-topic and subscriber-topic pairs in Dynamo, but later we figured out for this assignment that might be an overkill. For simplicity, we could just return the SQS queue url as the subscriber ID, and whenever a subscriber client needs to retrieve a message, it needs to provide that queue url (i.e. its ID). For publisher clients, when they need to publish a new message, they need to include the topic in request body. In other words, we actually store those pairs at the client side. However, this approach may not be persistent as the system becomes more complicated, and it’s better to have Dynamo as a key-value storage engine for more complex systems.