On The Fly

Harsh Chawla, Hemendra Vyas, Namratha Upadhya, Remon Roshdy

New York University

Tandon School of Engineering

Brooklyn, NY 11201

hc3338@nyu.edu, hv2115@nyu.edu, nf2207@nyu.edu, rr3531@nyu.edu

Demo link: https://youtu.be/j8ei_crswu8

GitHub link: https://github.com/harshchawla-26/On_The_Fly

***Abstract* — Cloud computing is one of the greatest inventions. It is the need of today's generation. It freed mankind from storing the large quantity of data on physical hardware. It helps the users to access their data online from anywhere and at any time. Today, whether an individual or a group of organizations, everyone is bound to use this technology. Many organizations have shifted their whole data center on the cloud and are free from maintaining it physically. In this paper, we provide complete documentation on "On The Fly" - News Recommendation and Summarization System. On the Fly makes sense of the daily news by selecting the best of the U.S. and international media into a succinct, lively digest. It is created for busy people like everybody. This news app has been developed using Amazon Web Services cloud platform. In this paper, we'll provide details on how we provide intelligent recommendations to convenient summarization of news articles irrespective of their length.**

Keywords — Cloud Computing, Amazon Web Services, API Design, Architecture, HuggingFace Model

I. Introduction

Nowadays, news is one of the most important channels for acquiring high-quality information regarding our society. Today, readers can find various sources of news online, e.g., on the web presences of traditional newspaper companies, on digital only news sites, or on news aggregation platforms provided, for example, by Google or Yahoo!. Additionally, the digital form of information delivery allows publishers to distribute new or updated content in real-time, leading to an increased speed of publication. However, with the rapid growth of the Web, more and more news articles are available causing information overload. At the same time, however, the abundance of available information and the constant update cycle make it increasingly challenging for readers to keep track of news that are most relevant to them. This is where Recommendation Systems are a game changer. Recommendation Systems have shown to be a valuable tool to help users in such situations of information overload. The main tasks of such systems are typically to filter incoming streams of information according to the users' preferences or to point them to additional items of interest in the context of a given object. This has helped us with filtering but how to overcome the rapid pace of the human mind? We don't really have the patience to read all of them at length. But a basic description solves it. This is where summarization falls in. News summarization can help to combat this problem by distilling the most important information from a large amount of news articles for users. The paper is organized as follows. Next, in Section 2, we briefly summarize how we selected existing works for consideration in our project. Afterwards, in Section 3, we provide the architecture in more detail and specify about every component's functioning. Our paper ends with an outlook on possible directions for future research and improvements in Section 4.

II. Previous Work

We have extensively researched many news recommendation and summarization applications to develop one for ourselves. The most useful ones which we came across are :

- InShort [1]- A simple android application which provides news articles recommendations to the user.
- FliPboard [2]- A simple android application which provides news articles recommendations to the user.
- SmartNews [3]- A simple android application which provides news articles recommendations to the user.

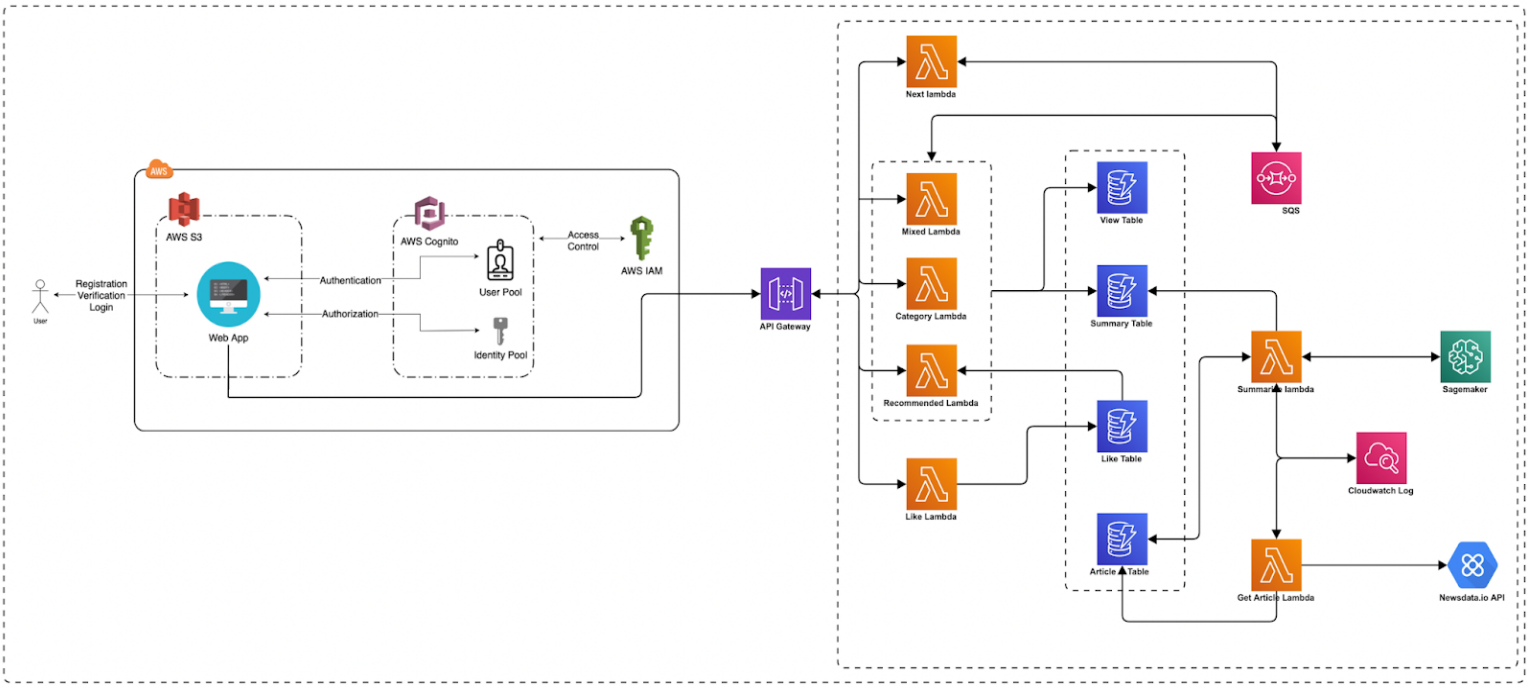

Figure 1: Architecture Diagram

III. Architecture

On the Fly application has different components which provide different functionality. Figure 1 is the fully fledged architecture of On the Fly application.

Features:

1. Initially, there is a login/registration page provided to the user. Once the users get validated the main page pops up.
2. Main page consists of different categories for the users to browse through. Users can now choose different news categories ranging from specific ones like sports and entertainment to recommendations and mixed/general.
3. Our application would now provide a main headline, summary of the article, URL to the actual news article for the chosen category.
4. Users can hit a 'like' button to any article of their choice. This would be taken into account while providing recommendations to the users.
5. Mixed/General category displays all news articles irrespective of the users.
6. Finally, users can log out at any time they prefer.
7. The architecture designed can help scale up to multiple users concurrently.
8. In the backend, news articles are being pulled from a 3rd party API and updated in the database everyday for users to gain access to.

AWS Components:

1. Lambda Functions
2. Simple Queue Services
3. CloudWatch trigger
4. Sagemaker
5. DynamoDB
6. API Gateway
7. AWS Cognito
8. S3 Bucket

IV. Implementation

Now that we have a basic understanding of the application let's dive right into the implementation. We have many modules implemented using Amazon Web Services like Lambda, Database, etc which have different functionalities. To understand it, we'll show and explain in detail all aspects of the architecture diagram.

IV.I. Backend

The backend aspect of "On the Fly" application needs to collect recent news articles, summarize it and store it in the database.

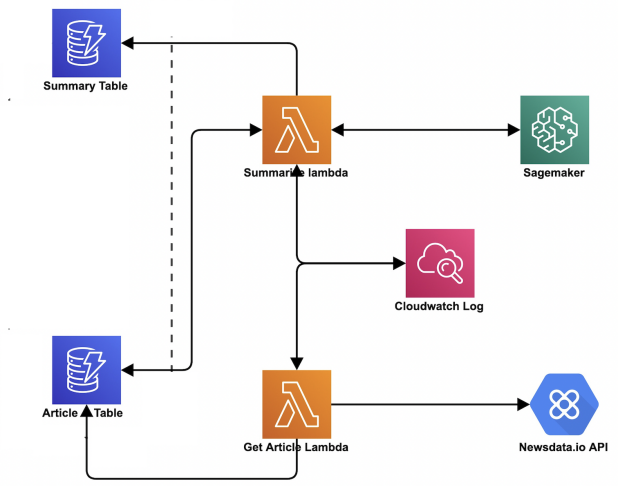

Figure 2: Backend part of architecture

So for this we are using:

1. AWS Lambda
2. DynamoDB
3. Newsdata.io API
4. AWS Sagemaker
5. AWS CloudWatch

To fetch news articles we use a third party newsdata.io API. The NewsData.io API uses API keys to authenticate requests. The latest news endpoint provides access to the latest and breaking news for a country, for a specific category in a country, or for a single or multiple domains.

The news articles are sorted by the publish date. Retrieving the latest news allows you to build experience such as showcasing the latest news, breaking news tickers and analyzing news to better understand their content.

IV.I.I Sagemaker

Amazon SageMaker[4] is a fully managed machine learning (ML) service that allows developers and data scientists to build, train, and deploy ML models quickly. It supports popular open-source libraries, provides pre-built ML models and algorithms, and offers tools and features for building, training, and deploying ML models. SageMaker integrates with other AWS services and is useful for building and deploying ML models in a production environment.

There are many trained models available for summarization purposes online. Hugging Face[5] is a community and data science platform that provides: Tools that enable users to build, train and deploy ML models based on open source (OS) code and technologies. There are many summarization models readily available in this repo. Since we are dealing with news articles which tend to vary in length, we wanted a text summarization model which is versatile. So, after regressive levels of testing we found the bart-large-cnn model as the most feasible one for our use case.

BART[6] is a transformer encoder-encoder (seq2seq) model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder. BART is pre-trained by (1) corrupting text with an arbitrary noising function, and (2) learning a model to reconstruct the original text.BART is particularly effective when fine-tuned for text generation (e.g. summarization, translation) but also works well for comprehension tasks (e.g. text classification, question answering). This particular checkpoint has been fine-tuned on CNN Daily Mail, a large collection of text-summary pairs.

IV.I.II CloudWatch

Amazon CloudWatch[7] is a monitoring service for AWS resources and the applications you run on the Amazon Web Services (AWS) cloud. CloudWatch provides a range of features and tools to help you monitor and manage your resources, including real-time data visualization, alarms, and automated actions. It is a useful tool for monitoring the health and performance of your AWS resources, and for detecting and reacting to issues in real-time.

There are two lambda functions which are triggered everyday by AWS CloudWatch:

1. Getarticles lambda function is used to fetch news articles from the third party newsdata.io API. These articles are then stored in the articles table of DynamoDB. Articles table consists of records where each row is {'url_link', 'category', 'content', 'title'}
2. Summarize Lambda function is used for fetching recent news articles from the articles table for summarizing purposes. Since a SageMaker endpoint is already deployed, we use it for text summarization.

3. Each news article would be summarized using this endpoint in lambda and pushed to another table called Summary table in DynamoDB. Each record in this table is of the form {‘url_link’, ‘category’, ‘summary’, ‘title’}.

IV.II. User Interaction End

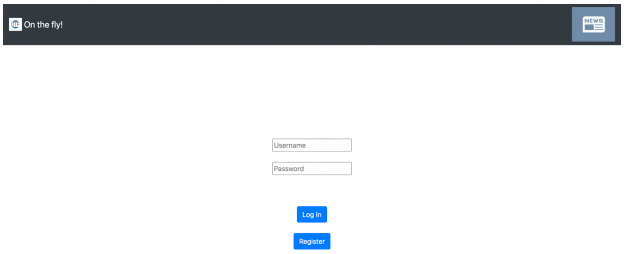

Figure 3: Login Page

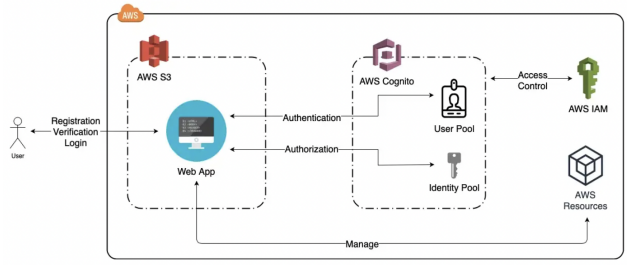

Figure 4: Architecture of Cognito

IV.II.I S3 and Cognito

Amazon Simple Storage Service (S3)[8] is a cloud storage service that can be used to store and retrieve any amount of data from anywhere on the web. It is a popular choice for storing and serving static assets, such as images, JavaScript and CSS files, for front-end applications. S3 offers high durability and availability, as well as tools for managing and securing data. Amazon Cognito[9] is a cloud service that provides user management and authentication for mobile and web applications. It offers features such as sign-up and sign-in, as well as social media and enterprise identity provider integration. Cognito also provides user data storage, which can be used to store user profile information and app preferences. It is a useful tool for building secure and scalable applications that need to authenticate and authorize users.

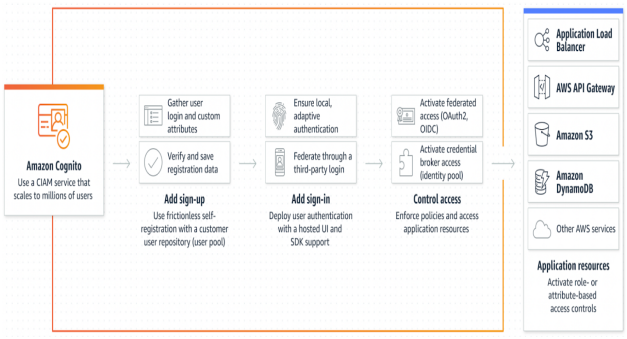

Figure 5: Working of AWS Cognito

Login and Sign Up pages are implemented using AWS Cognito, S3 Bucket. S3 bucket consists of all html files which would be used for frontend purposes. Making this bucket public would serve the beginning of the application. If it’s a new user, we could sign up to the application. With Amazon Cognito, you can add user sign-up and sign-in features and control access to your web and mobile applications. Amazon Cognito provides an identity store that scales to millions of users, supports social and enterprise identity federation, and offers advanced security features to protect your consumers and business. Built on open identity standards, Amazon Cognito supports various compliance regulations and integrates with frontend and backend development resources.

After the user has been successfully logged in, they get to see a main page like the one below. The user now has the provision of choosing a category.

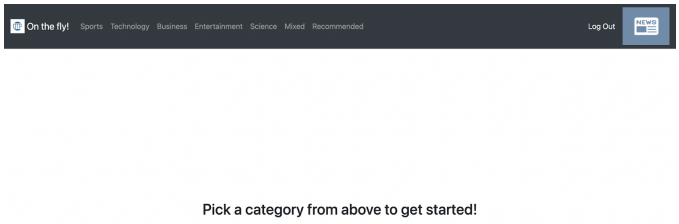

Figure 6 : Landing page

Every category is a request to the backend to fetch out all news articles for it. Users now would be able to browse through all articles of his/her choice. If the users love the

articles they can hit a like button. This would be taken into consideration during the Recommended category. Users would receive news articles from the liked categories.

After reading the articles, users can then log out for others to use the application.

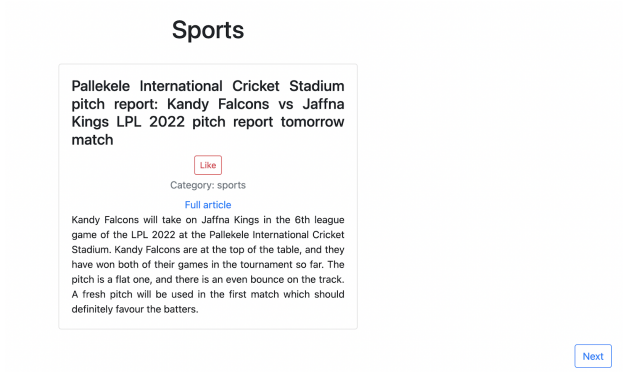

Figure 7: Summary card example

Say, we choose the Sports category, the above figure is the snapshot of how the application looks like.

Being the crucial aspect of our application, below is the design we have used:

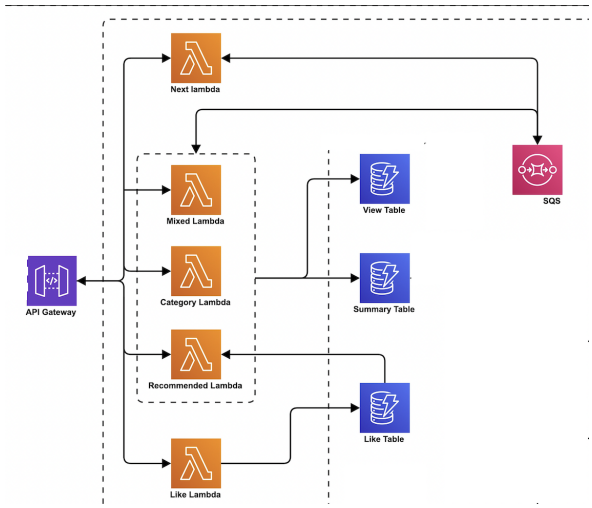

Figure 8: Architecture for User End

So, for this we are using:

1. AWS API Gateway
2. AWS Lambda Functions
3. AWS DynamoDB
4. AWS Simple Queue Service

IV.II.I API Gateway

AWS API Gateway[10] is a service that allows developers to create, publish, maintain, and secure APIs. It enables real-time communication through RESTful APIs and WebSocket APIs and can handle hundreds of thousands of concurrent API calls. API Gateway integrates with other AWS services and can also be connected to third-party services. It offers tools to build and manage APIs, including custom domains, stages, and authorizers. It is useful for building scalable APIs that can be accessed from web and mobile applications, as well as other APIs and backend systems.

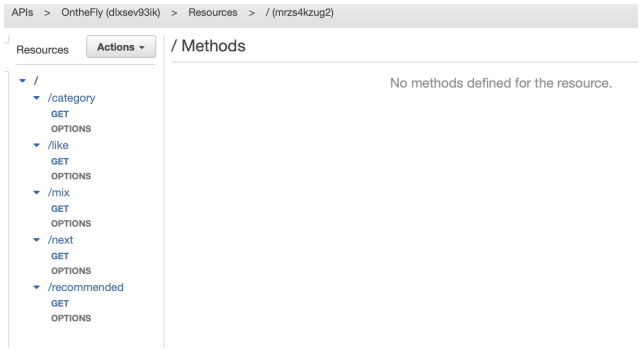

Figure 9: API setup

The AWS API Gateway is the connectivity being used to communicate from frontend to backend. API gateway sends GET requests per category to the respective lambda functions.

1. /category - This API would be used for retrieving news articles for specific categories for a specific username.
2. /like - When the users hit a like button to a news article, the API makes a request to the database.
3. /mix - This API would be used for retrieving news articles in general.
4. /next - This API would be used for retrieving the next article from the queue for the user.
5. /recommended - This API would be used for retrieving personalized news articles for the user.

IV.II.III Lambda Functions

AWS Lambda[11] is a cloud-based service that allows you to run code in response to specific events or triggers, such as changes to data in an Amazon S3 bucket or a request to an API. It executes your code in a fully managed environment, meaning you don't have to worry about the

infrastructure or scaling of your applications. Lambda functions can be written in multiple programming languages and are useful for building event-driven applications, microservices, and real-time data processing systems.

There are multiple lambda functions handling different functionalities. API Gateways are connected to the lambda function which performs backend operations and pushes to the queue.

1. Next Lambda : This function pushes the next news article details to the queue for the frontend to read.
2. Mixed Lambda : This function scans the summary dynamodb for all summaries and matches against the view dynamo db to push general news summaries that the user has not browsed before to the queue for the frontend to access.
3. Category Lambda : This function for a specific category scans the summary dynamodb and matches against the view dynamo db to push news summaries for that category to the queue for the frontend to access.
4. Recommended Lambda : This function computes the top 2 category that the user liked by scanning the like dynamodb against the view table to generate a list of 10 personalized news summaries are pushed to the queue for the frontend to access.
5. Like Lambda : This function pushes 'liked' news articles to the Like table in DynamoDB. In this table every record is of the form: {usernmae,link, category}.

IV.II.III DynamoDB

Amazon DynamoDB[12] is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. It is a key-value store that can be used to store and retrieve data using primary keys, allowing for fast and efficient querying and updating of data. DynamoDB is designed for high availability and durability, with automatic data replication across multiple availability zones. It is a popular choice for building serverless applications, as it can automatically scale with the demands of the application and does not require any infrastructure maintenance.

We have implemented Dynamodb to store various types of records to keep track of the user activity. This helped us build a more peronalizable application

1. Summary DynamoDB: This stores the summaries of the articles pulled on a daily basis. The schema is {article_link, category, summary,title} with the link as the partition key.
2. View DynamoDB: This table stores the record of all the summaries the user has already browsed through to make sure a user is never provided with the same summer twice. The schema for this table is {username, link, category} with username as the primary key and link as the sorting key.
3. Like DynamoDB: This table stores records of all the summaries every user has every like. This helps us build a recommendation list of summaries for the user. The schema for the table is {username, link, category}. With the username as partition key and the link as the sorting key.

IV.II.IV SimpleQueueService

Amazon Simple Queue Service (SQS)[13] is a fully managed message queue service that enables you to send, store, and receive messages between software systems in a reliable and scalable manner. It helps you to decouple and scale microservices, distributed systems, and serverless applications, and supports both standard and FIFO (First-In-First-Out) queues. SQS provides features such as message scheduling, message deduplication, and message retention, and is useful for building distributed applications, integrating with cloud-based applications, and implementing workflows.

SQS enabled us to generate quicks with the required list of summaries to present to the user with a quick turnaround.

1. Category Queue: This is generated for a specific category summaries list.
2. Mix Queue: This is generated for all summaries so that the user can browse through the entire collection in one go.
3. Recommended Queue: This is a queue generated for a customised list of 10 summaries based on the users liking.

V. Future Scope

There are several potential areas for future expansion and improvement for the news summary web application hosted on AWS. One potential avenue for growth is the integration of a sharing feature, which would allow users to send summaries they find particularly interesting or valuable to their friends and colleagues. This feature could be implemented using a combination of AWS services,

such as Amazon SNS[14] for messaging and Amazon SES for email delivery.

Another interesting direction for the web application could be the integration of Amazon codestar[15] to enable Alexa compatibility. This would allow users to access and interact with the web application using voice commands through their Alexa-enabled devices. This feature could be implemented using the AWS Lambda service, which allows developers to run code in response to specific triggers or events.

Overall, these enhancements would not only provide value for our users, but also help to differentiate the web application from competitors and drive additional traffic and engagement.

VI. CONCLUSION

In conclusion, the implementation of a news summary web application on AWS has proven to be a successful and scalable solution. By utilizing a variety of AWS services, such as Amazon S3 for static asset hosting, Amazon Cognito for user management and authentication, and Amazon Lambda for serverless back-end logic, we were able to build and deploy a performant and reliable application. The use of AWS allowed us to focus on the development of our application, rather than on the infrastructure and maintenance of the hosting environment. Overall, the decision to host our news summary web application on AWS was a wise choice that has paid off in terms of performance, security, and cost-effectiveness.

VII. REFERENCES

[1] https://www.inshorts.com/
[2] https://about.flipboard.com/
[3] https://www.smartnews.com/en/
[4] https://docs.aws.amazon.com/sagemaker/index.html
[5] https://huggingface.co/docs
[6] https://huggingface.co/facebook/bart-large-cnn
[7] https://docs.aws.amazon.com/cloudwatch/index.html
[8] https://docs.aws.amazon.com/s3/index.html
[9] https://docs.aws.amazon.com/cognito/index.html
[10] https://docs.aws.amazon.com/apigateway/index.html
[11] https://docs.aws.amazon.com/lambda/index.html
[12] https://docs.aws.amazon.com/dynamodb/index.html
[13] https://docs.aws.amazon.com/dynamodb/index.html
[14] https://docs.aws.amazon.com/sns/index.html
[15] https://aws.amazon.com/codestar/