**CLOUD COMPUTING**

Cloud computing is the on-demand delivery of compute power, database storage, applications, and other IT resources through a cloud services platform via the internet with pay-as-you-go pricing.

Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet. A Cloud services platform such as Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application. A cloud services platform provides rapid access to flexible and low cost IT resources. With cloud computing, you don’t need to make large upfront investments in hardware and spend a lot of time on the heavy lifting of managing that hardware. Instead, you can provision exactly the right type and size of computing resources you need to power your newest bright idea or operate your IT department. You can access as many resources as you need, almost instantly, and only pay for what you use.

**AMAZON WEB SERVICES (AWS)**

Amazon Web Services provides a highly reliable, scalable, low-cost infrastructure platform in the cloud.

AWS offers low, pay-as-you-go pricing with no up-front expenses or long-term commitments. Amazon Web Services (AWS) provides a broad set of products and services you can use as building blocks to run sophisticated and scalable applications. Running your applications in the AWS Cloud can help you move faster, operate more securely, and save substantial costs, all while benefitting from the scale and performance of the cloud. The AWS Cloud provides a broad set of infrastructure services, such as computing power, storage options, networking and databases, delivered as a utility.

**AWS MOBILE HUB and PINPOINT**

The AWS platform handles demanding requirements for virtually any application, and for our project we used variety of services provided by the AWS such as the mobile services of AWS known as AWS Mobile Hub for the deployment of our Android application on the AWS cloud. AWS Mobile Hub gives you a single place to easily configure AWS services. It generates a cloud configuration file, which stores information about configured services. The 'awsconfiguration.json' file you download contains the configuration of backend resources that Mobile Hub enabled in our project. We add the backend service configuration file to our app which enables us to connect to AWS services and use the AWS SDKs. The AWS Mobile Android SDK helps to build high quality mobile apps quickly and easily through the integration of various AWS services for better functionality and wide variety of usages. They provide easy access to a range of AWS services, including Amazon Cognito, AWS Lambda, Amazon S3, Amazon Kinesis, Amazon DynamoDB, Amazon Pinpoint and many more.

Amazon Pinpoint helps to engaging by sending email, SMS, and mobile push messages. Whether you're a developer, marketer, or business user, you can use Amazon Pinpoint for target messages (such as promotional alerts), as well as direct messages (such as order confirmations and password reset messages) to your customers.

You can integrate Amazon Pinpoint into your apps to check usage, which helps you better understand how customers interact with your apps. Amazon Pinpoint also tracks the ways that your customers respond to the messages you send.

You can send direct messages—such as order confirmations, welcome messages, and one-time passwords—using the console or the Pinpoint REST API. You can also use the API to build custom applications that deliver campaign and transactional messages across multiple channels.

**CONNECTING TO AWS BACKEND**

After downloading and adding the cloud configuration file in the 'raw' directory of 'res' of our android project for backend, we connect to the backend services to utilize the benefits of the cloud. The AWS SDK provided code has to be added to the application source code to connect.

Uses permission had to be added to the AndroidManifest.xml file to allow Internet and Network State Access.

Dependencies were added to our app/build.gradle and synched by choosing Sync Now in Android Studio. These dependency libraries enable basic AWS functions, like credentials, and analytics.

AWS provided code had to be added to the onCreate function of our MainActivity and AWSMobileClient class had to be imported. AWSMobileClient is a singleton that establishes your connection to and acts as an interface for our services.

And the app is set up to interact with the AWS services configured in our Mobile Hub project. Various features can be easily added through the source code and by using the AWS Mobile Hub Management Console.

To check the connectivity to the AWS cloud, we run the application and in our logcat, we search for 'Welcome to AWS!' which ensures our app is successfully able to connect to cloud and can utilize its benefits.

**CAPTURING ANALYTICS AND MONITORING THROUGH PINPOINT ANALYTICS**

To gather the data that helps improve our app's usability, sessions, and engagement with our users. Mobile Hub deploys our analytics backend when enabling the Messaging and Analytics feature, which uses the Amazon Pinpoint service. Analytics were by default enabled when adding and connecting to the backend. To ensure that the analytics are enabled or not we check out Mobile Hub console of our application, and a green tick present there ensures it.

Calls to start and stop session for analytics were added using AWS SDK code. Analytics session is started at the beginning of the onCreate() method of our MainActivity. To stop the session, we use stopSession() and submitEvents() at the last point in the session we wanted to capture. PinpointMangager and PinpointConfiguration classes were imported for the analytics session capture and sending reports to the console.

Metrics for the application usage can be checked on the console by going to the 'Analytics' tab in Mobile Hub window of our project which opens up the Pinpoint services where our metrics are sent. The metrics that were monitored were 'Daily Active endpoints', 'New Endpoints', 'Monthly Active endpoints', 'Sessions', '7-day retention period' and 'Revenue'.

**ADDING PUSH NOTIFICATIONS THROUGH PINPOINT CAMPAIGN AND DIRECT MESSAGING**

To add push notification to our application we use the 'Messaging and Analytics' feature of Amazon Pinpoint Service. Amazon Pinpoint enables apps to receive mobile push messages sent from the Apple (APNs) and Google (FCM/GCM) platforms. For android we have the option of using either the Google Cloud Messaging(GCM) or the Firebase Cloud Messaging(FCM). We require that in order to obtain the application API key and Sender ID as it is required to send notification to the devices where our application is installed.

We create a new Firebase project for our application that provide us with required application key and Sender ID. After obtaining the required credentials from the Firebase console, we add the application API key and Sender ID under the 'Android' section which is in 'Push notification' option in Messaging. When the operation is complete, an alert will pop up saying "Your Backend has been updated", prompting to download the latest copy of the cloud configuration file. After completion of configuring the feature, we choose the pop-up banner to return to the project details page and click on the 'Integrate' button.

We have to update our application with the latest configuration file and then we need to connect to the backend service using the AWS Mobile SDKs. AndroidManifest.xml file needs to be edited and uses-permission have to added to be able to receive notification. Dependencies have to be added in app/build.gradle and also project level build.gradle. We have to create an Amazon Pinpoint client in the location of your push notification code and it is done so by adding the SDK code into our onCreate() function and importing the GoogleCloudMessaging class and the InstanceID class. A new PushListener class is created and making required updates on AndroidManifest file.

Amazon Pinpoint console enables you to target your app users with push messaging. We can send individual messages or configure campaigns that target a group of users that match a profile that you define. For instance, we could email users that have not used the app in 30 days, or send an SMS to those that frequently use a given feature of our app and these updates of features can be managed easily through the Pinpoint Console. Through the console we can add title and body of our notification, add endpoints i.e., the users of our application to whom we send notification to, and the function that it notification will perform when user interacts with it. The 'Direct Message' can be used to send a direct message. A direct message is a one-time message that can be send to a limited audience without creating a campaign.

We configure a campaign based on certain criteria based on which our users will receive notifications and interact with the app. These campaign can be edited any time easily through the console. Some of the basic settings include how frequently user will receive notification, the criteria on which user will receive the notification, quiet time when no notification will be sent, and obviously the title and the body of the notification and what action that takes place when uses clicks on it.