The objective of the project was to build a end-to-end service that integrates a social media dashboard, which they wanted to build, with Facebook, Twitter, and Instagram feeds.

We should automatically fetch all the brand promotions and campaigns that we are trying to build, and analyse how people are responding to each of the brand promotions and campaigns.

We identify these brand promotions and campaigns based on hashtags. Each campaign will have a specific hashtag that we have taken from the client's promotions. On a daily basis, we go to Twitter and Facebook. Twitter has a package called Tweetpik, and Facebook has its own APIs. We have connected to the Twitter package as well as Facebook by using API keys provided by the companies. We started fetching the tweets and posts related to the promotions. This is scheduled to run every day.

I used AWS Cloud Services for automation, specifically Cloud Schedulers. I use the scheduler to automatically fetch this information and dump the data in an S3 bucket. The data they dump includes the user's name, the review they have written, how many people liked it, how many people shared it, and how many people commented on it. This kind of information is what I will get. The same process is followed for Facebook, including data related to posts, likes, comments, and other features.

Using inbuilt APIs from the companies, we started extracting the data. We built an automated pipeline using Python schedules, which runs every night to fetch the information. The data gets aggregated every day and is available for us to analyse in the morning.

After fetching the data from the S3 bucket, we built an entire preprocessing pipeline to clean the data. The steps include converting text to lowercase, removing stop words, lemmatization, parts of speech tagging, special character removal, URL removal, emoticon conversion, converting smileys to actual emojis, and emoji removal. All these steps are part of the pipeline.

Next, we use the TF-IDF vectorizer to convert the text to numerical data (embedding). After successfully converting the data to numerical form, we divide the data into training, testing, and validation sets. We then started building the model. Here, I prefer the Naive Bayes model for model building because it is easy to deploy and runs faster compared to other models. Just based on the speed of the algorithm, we opted for Naive Bayes.

After building the model, we went for model validation using a confusion matrix and K-fold validation. Once we completed this, we proceeded to deploy the model in an API form. This is my project.