Cognito Bot – Design Document

Keshav Pandey Monika Jengaria

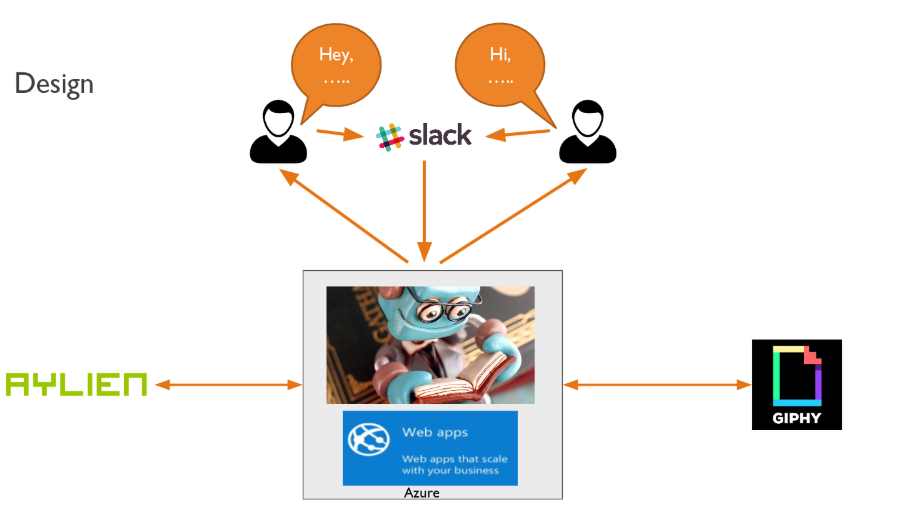
**Aim:**

Our project focuses on creating a chatbot which detects the mood of people interacting (in groups, channels and direct messages) and helps elevate the mood by sending out fun gifs based on the sentiment of the conversations taking place.

The project is to build a slack bot can integrate with a slack channel and detect the mood of the users on the channel as they exchange messages.

**Scope**:

* Build a slack bot that integrate with a slack channel, groups
* The bot should be able to read and respond to messages on slack
* The bot should integrate with an application that would do sentiment analysis on the inputs sent by users
* An application which can integrate with 3rd party service to fetch relevant media to generate responses (gifs)
* An application that receives input and uses 3rd party services to do sentiment analysis and generates a response (to be used by bot)



**APIS:**

**Aylien API (Sentiment Analysis):**

We explored APIs from AWS Rekognition and Azure’s cognitive services which were very promising. They provided detection of multiple sentiments, key word identifications etc. However, we couldn’t make use of it due to the pricing structure. Hence, on searching for free to use API’s we encountered Aylien which gives us a sentiment score between positive and negative with a score between 0-1.

**Giphy API:**

We are utilizing Giphy’s APIs to select a random GIF based on a search based from the response of the sentiment API to help elevate the user’s mood.

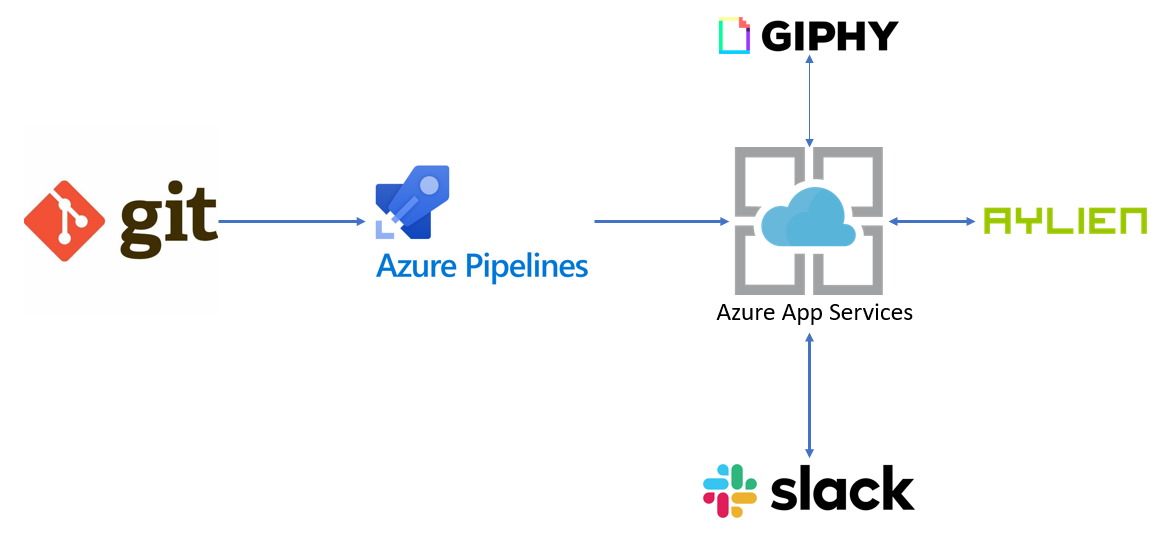
**Slack API:**

We used Slack’s APIs using events to listen to incoming messages from users and send back responses generated from our application.

**Infrastructure:**

We use Git for managing our code base. We have Azure pipeline which runs Builds and tests on every pull request to check for any breaking changes. We deploy our code base into an Azure app service. We have configured a Deployment pipeline which will push the latest code to our app service hosting on every commit pushed to the master branch.

Our application is deployed to Azure app service which is hit by slack’s events api. We analyse the messages using Aylien’s API and then use Giphy to fetch a good gif to send back to Slack as a response.



Infrastructure Diagram

**Testing**

We have unit tests for functionality and integration tests to check for the overall working of our application.

**Application Considerations & Future Work:**

Explore optimizations for scaling: We wanted to be prepared for scalability and large number of messages coming in and hence chose to go with a deployment on Azure. We should be able to configure load sharing on multiple VM’s which are spawned based on actual load.

How fast to respond: Often humans take some time to receive a message, understand and reply. In our case the bot might be able to do the entire process in a matter of 100s of milliseconds. To keep the dialog more natural we are delaying the responses we send in order to make it appear more humanlike.

Preferences and threshold: We have configured our bot to respond only based on negative or positive sentiments which are above a threshold. For now, we have chosen this to be at 70% polarity score. In the future we would like to allow users to configure this for their own use cases.

If we have an error, we default to positive responses in our application. We can improve this with context and situational awareness.