IBM Watson Assistant

An enterprise artificial intelligence (AI) assistant that helps businesses enhance brand loyalty and transform their customer experiences by delivering proactive and personalized services while ensuring data privacy.

An AI assistant for:

Your brand:
Increased engagement and loyalty

Your customers:
Guided learning for customized experiences

Your data:
Secure and personalized insights

Intent Password Reset

“ I forgot my password...”

“ How do I get a new password? ”

“ Can’t login into your site... ”

“ My login isn’t working, please help... ”

“ Can you reset my password? ”

“I’m frustrated, I haven’t been able to login into your online billing system”

Extract other key information from a question

Intent	Password Reset

"I'm frustrated, I haven't been able to login into your online billing system"

Extract other key information from a question

Intent	Password Reset
Entities	Online Billing System

“I’m frustrated, I haven’t been able to login into your online billing system”

Extract other key information from a question

Intent	Password Reset
Entities	Online Billing System
Emotional Tone	Anger → Leverages Watson Tone Analyzer

“I’m frustrated, I haven’t been able to login into your online billing system”

Extract other key information from a question

Intent	Password Reset
Entities	Online Billing System
Emotional Tone	Anger → Leverages Watson Tone Analyzer
Context	Bill Smith, 47 / Gold Member
Context	Mobile

Watson Assistant

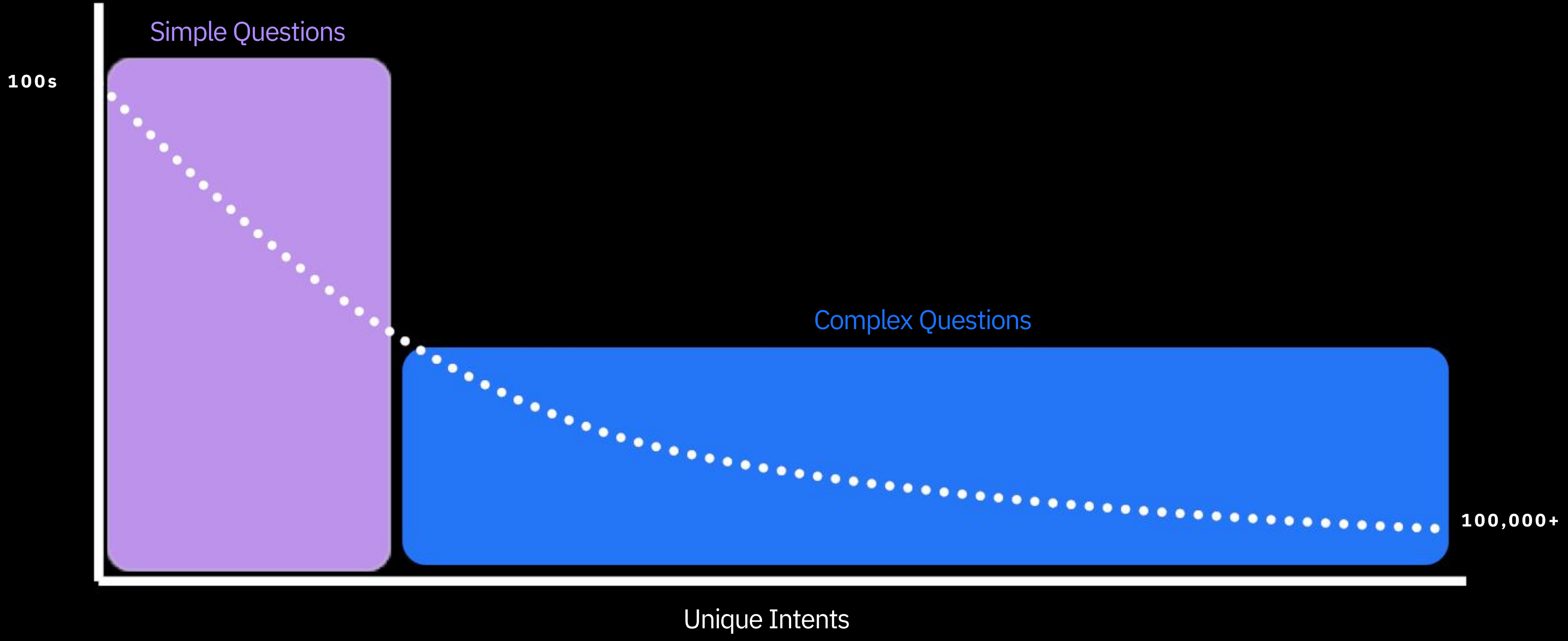
+

Watson Discovery

Watson uses reasoning strategies that focus on the language and context of the **question** to answer simple questions.

Watson uses reasoning strategies that focus on identifying the most appropriate **answer** for complex questions using the context of the entire corpus.

Watson Assistant + Discovery for all types of Qs

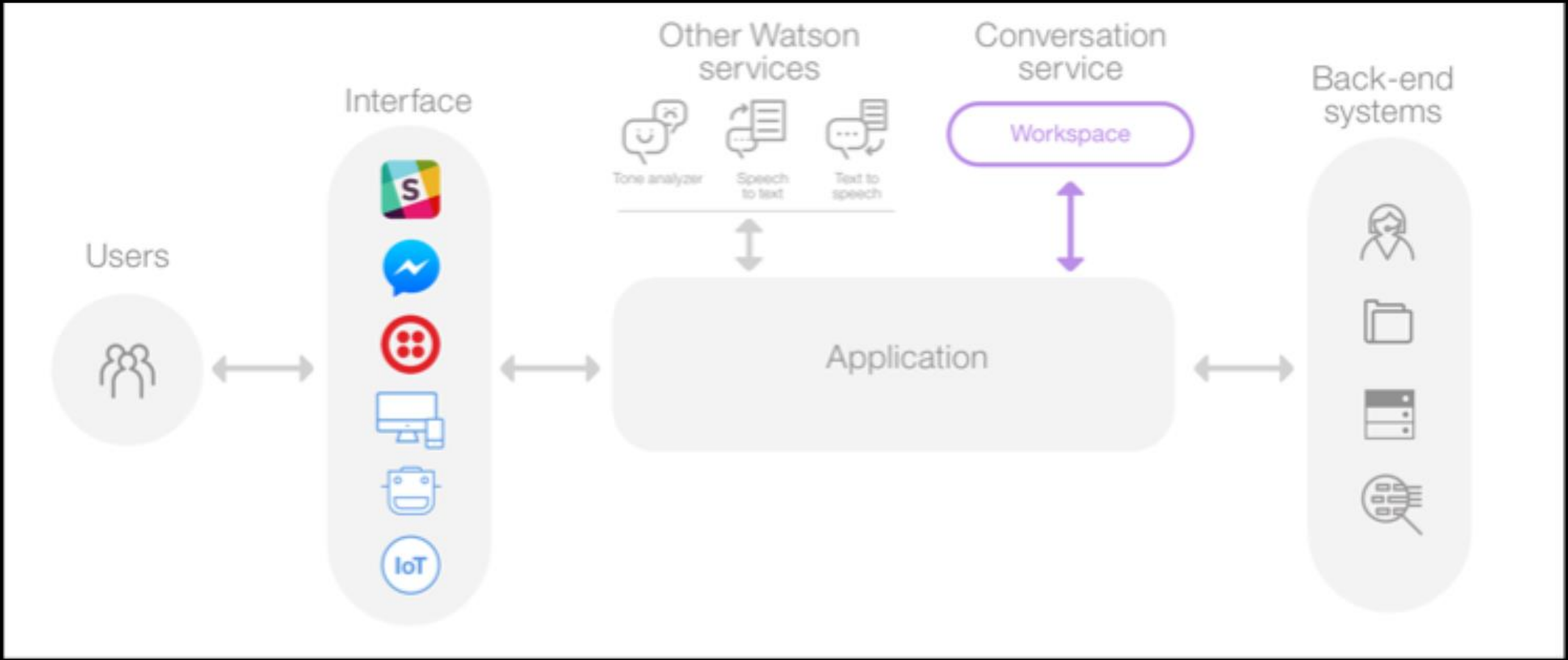


" My exhaust is making a rattling sound, how do I troubleshoot the problem? "

" How do I turn on the headlights?"

" How do I turn on my device "

" My device won't power on. I am concerned if I do a reset that I will lose my data. What should I do? "


Users
Interface
IoT
Other Watson services
Tone analyzer
Speech to text
Text to speech
Conversation service
Workspace
Application
Back-end systems

CREATE A SKILL

- ❑ First step in the Conversation tool is to create a skill.
- ❑ The natural-language processing for the Conversation service happens inside a skill, which is a container for all of the artifacts that define the conversation flow for an application.
- ❑ A single instance can contain multiple skills.
- ❑ A skill contains the following types of artifacts: Intents, Entities, Dialog
- ❑ As you add information, the skill trains itself, so you don't have to do anything to initiate the training.

INTENT

- An intent represents the purpose of a user's input.
- By recognizing the intent expressed in a customer's input, the Watson Assistant service can choose the correct dialog flow for responding to it.
- In the user interface, the name of an intent is always prefixed with the # character.
- To plan the intents for your application, you need to consider what your customers might want to do, and what you want your application to be able to handle.
- To train the skill to recognize your intents, you supply lots of examples of user input and indicate which intents they map to.

ENTITIES

An entity represents a class of object or a data type that is relevant to a user's purpose.

For example, an entity might represent a city where the user wants to find a business
location, or the amount of a bill payment.

By recognizing the entities that are mentioned in the user's input, the Watson Assistant
service can choose the specific actions to take to fulfill an intent.

If intents represent verbs (something a user wants to do), entities represent nouns (such as the object of, or the context for, an action).

In the user interface, the name of an entity is always prefixed with the @ character.

To train the skills to recognize your entities, you list the possible values for each entity and synonyms/patterns(regex) that users might enter.

DIALOG

A dialog is a branching conversation flow that defines how your application responds when it recognizes the defined intents and entities.

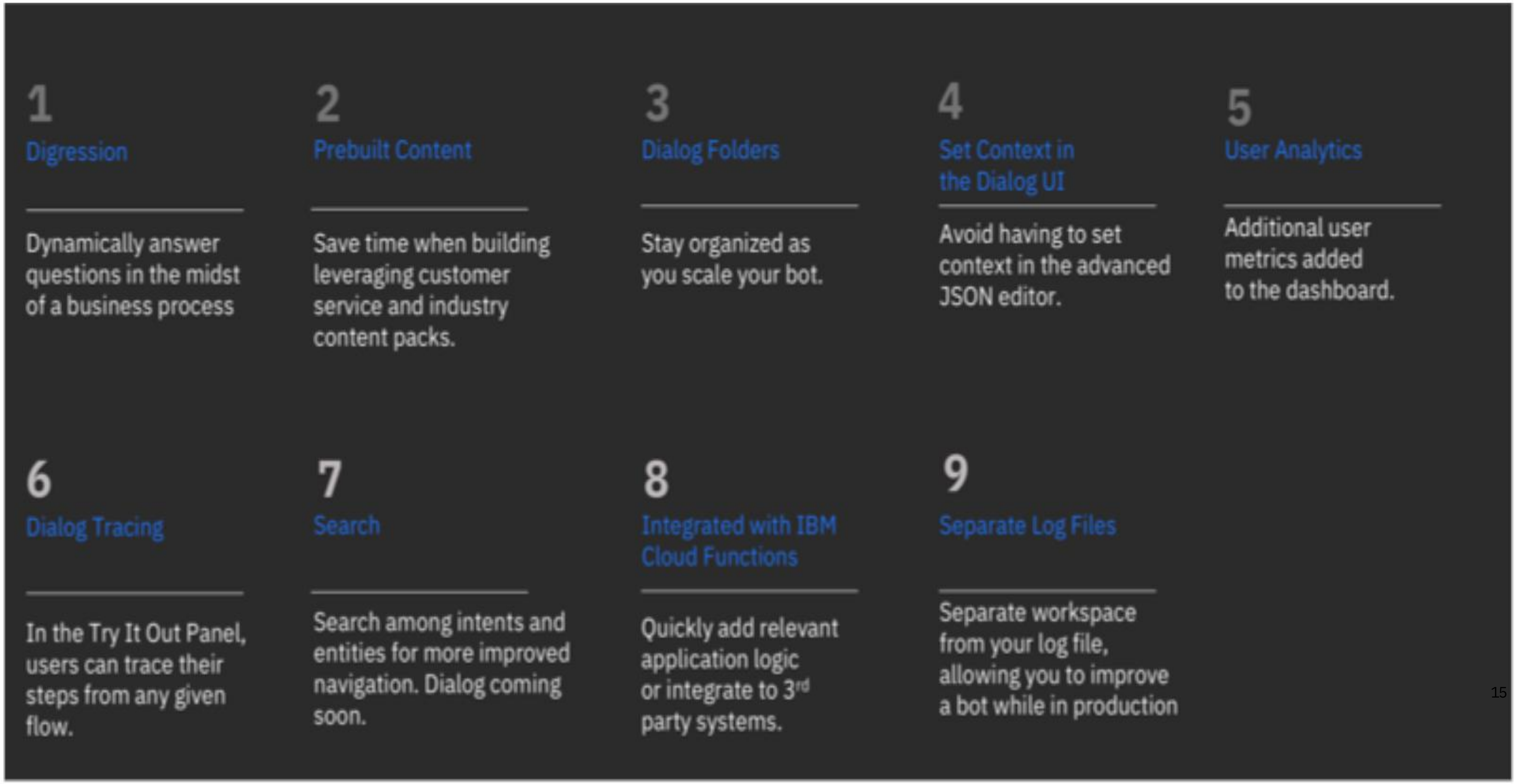
You use the dialog builder to create conversations with users, providing responses based on the intents and entities that you recognize in their input.

The application can pass information to the dialog, and the dialog can update the context information and pass it back to the application.

Valid expressions in conditions are written in the Spring Expression (SpEL) language.

Your dialog is represented graphically as a tree; create a branch to process each intent that you define.

Watson Assistant Features



1
Digression

Dynamically answer questions in the midst of a business process

2
Prebuilt Content

Save time when building leveraging customer service and industry content packs.

3
Dialog Folders

Stay organized as you scale your bot.

4
Set Context in the Dialog UI

Avoid having to set context in the advanced JSON editor.

5
User Analytics

Additional user metrics added to the dashboard.

6
Dialog Tracing

In the Try It Out Panel, users can trace their steps from any given flow.

7
Search

Search among intents and entities for more improved navigation. Dialog coming soon.

8
Integrated with IBM Cloud Functions

Quickly add relevant application logic or integrate to 3rd party systems.

9
Separate Log Files

Separate workspace from your log file, allowing you to improve a bot while in production

Watson Assistant Premium Features

Improve your workspace with Recommendations

Recommendations are presented to easily and efficiently improve your system.

Recommendations based on analysis of conversations

- How users interacted based on analysis on conversations
- The certainty that the system had about its response
- Training data currently used by your system

Improve Existing Intents

This recommendation involves taking individual phrases entered by users, that the system does not recognize, and then presenting them to you to select an intent for each phrase. This will help your workspace better understand what your users are saying.

One Premium deployment can have up to 30 instances, each with 20 workspaces.

The Premium Plan uses network containers and VMs (virtual machines) to provide isolation for cognitive service instances, and runtime storage isolation.

Make programmatic calls from Dialog Node to third party API’s

https://github.com/watson-developer-cloud/assistant-with-discovery-openwhisk

Workshop:

https://developer.ibm.com/patterns/create-cognitive-banking-chatbot/

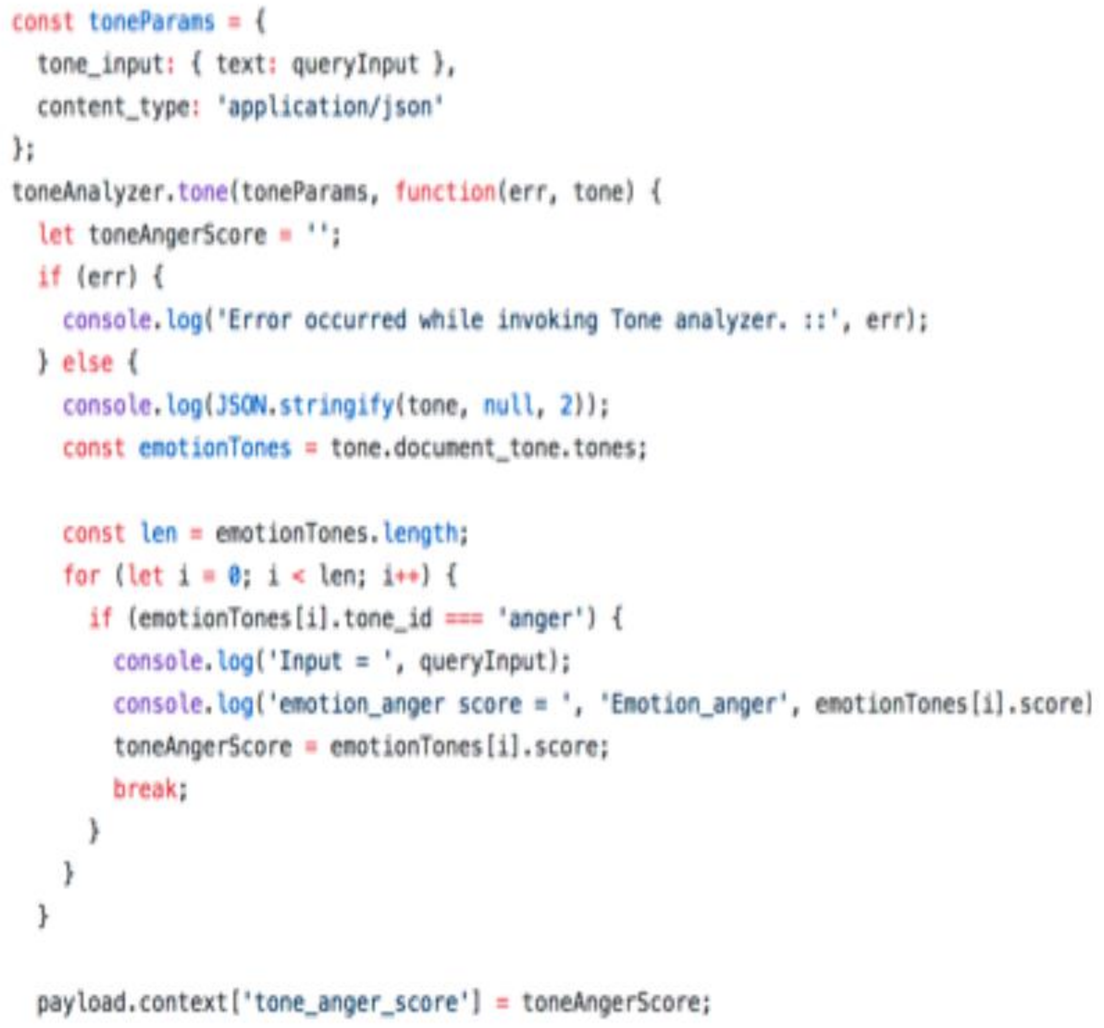
https://ibm.co/2NRqDoK

Adding a Promocode:

https://cognitiveclass.ai/applying-ibm-cloud-promo-code/

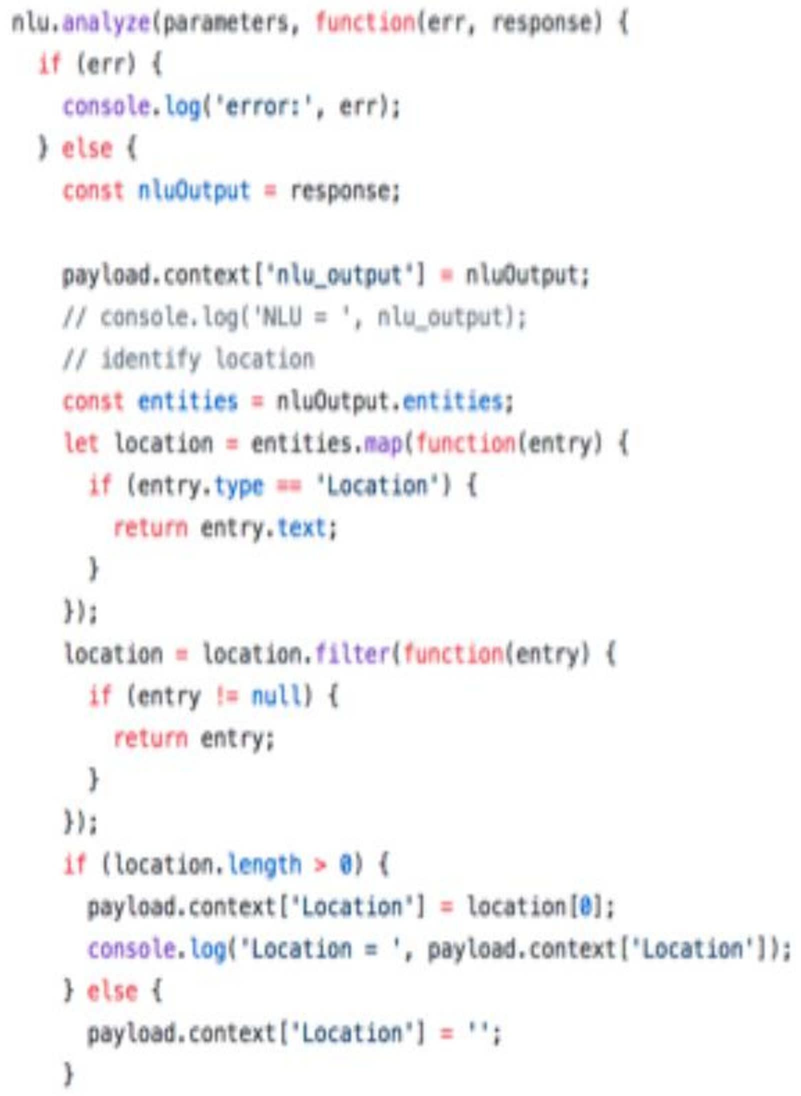
https://bit.ly/2TOeEv6

Tone Analyzer implementation



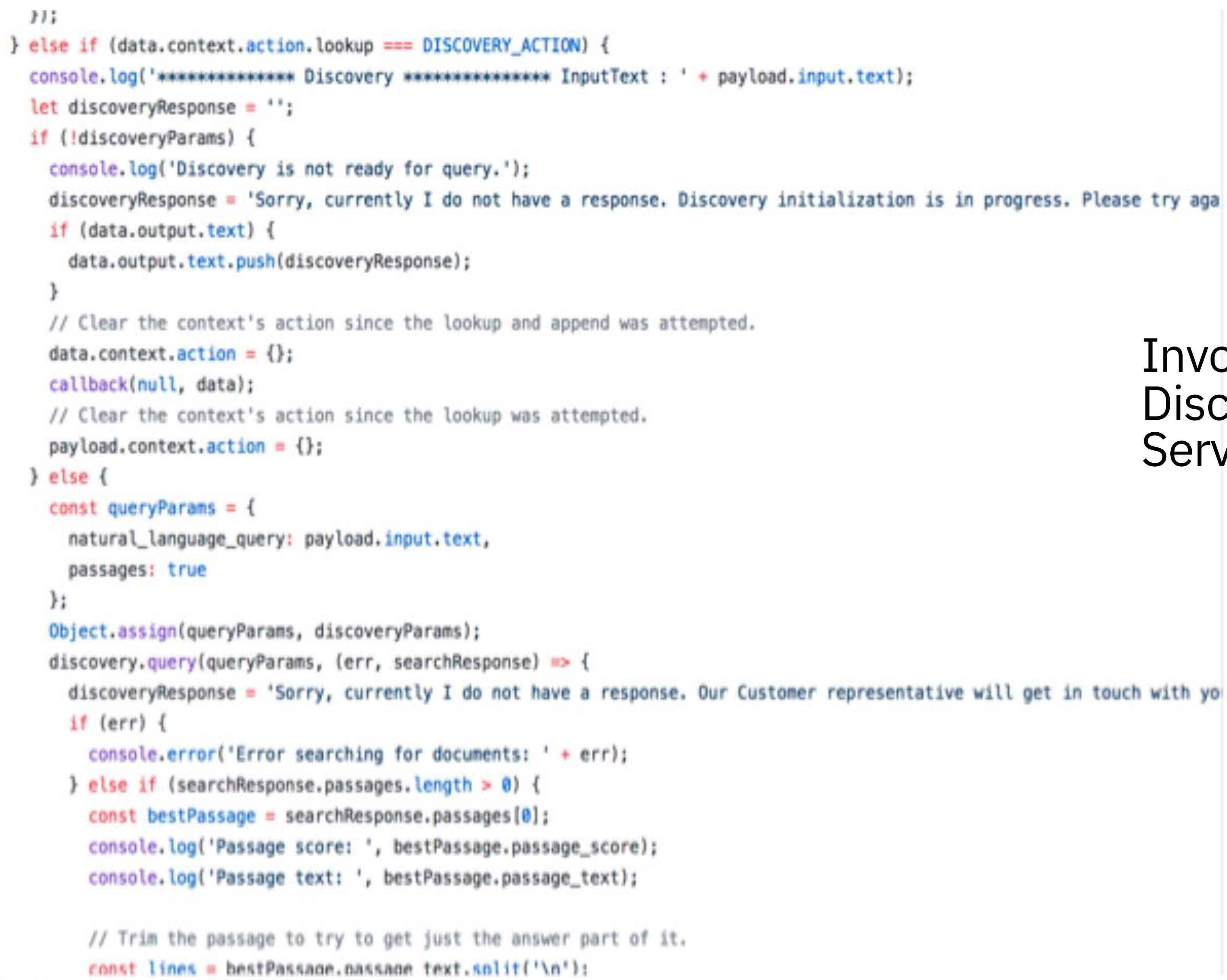
```
const toneParams = {
  tone_input: { text: queryInput },
  content_type: 'application/json'
};
toneAnalyzer.tone(toneParams, function(err, tone) {
  let toneAngerScore = '';
  if (err) {
    console.log('Error occurred while invoking Tone analyzer. ::', err);
  } else {
    console.log(JSON.stringify(tone, null, 2));
    const emotionTones = tone.document_tone.tones;

    const len = emotionTones.length;
    for (let i = 0; i < len; i++) {
      if (emotionTones[i].tone_id === 'anger') {
        console.log('Input = ', queryInput);
        console.log('emotion_anger score = ', 'Emotion_anger', emotionTones[i].score)
        toneAngerScore = emotionTones[i].score;
        break;
      }
    }
  }

  payload.context['tone_anger_score'] = toneAngerScore;
```

Invoking NLU service to extract entities from utterance



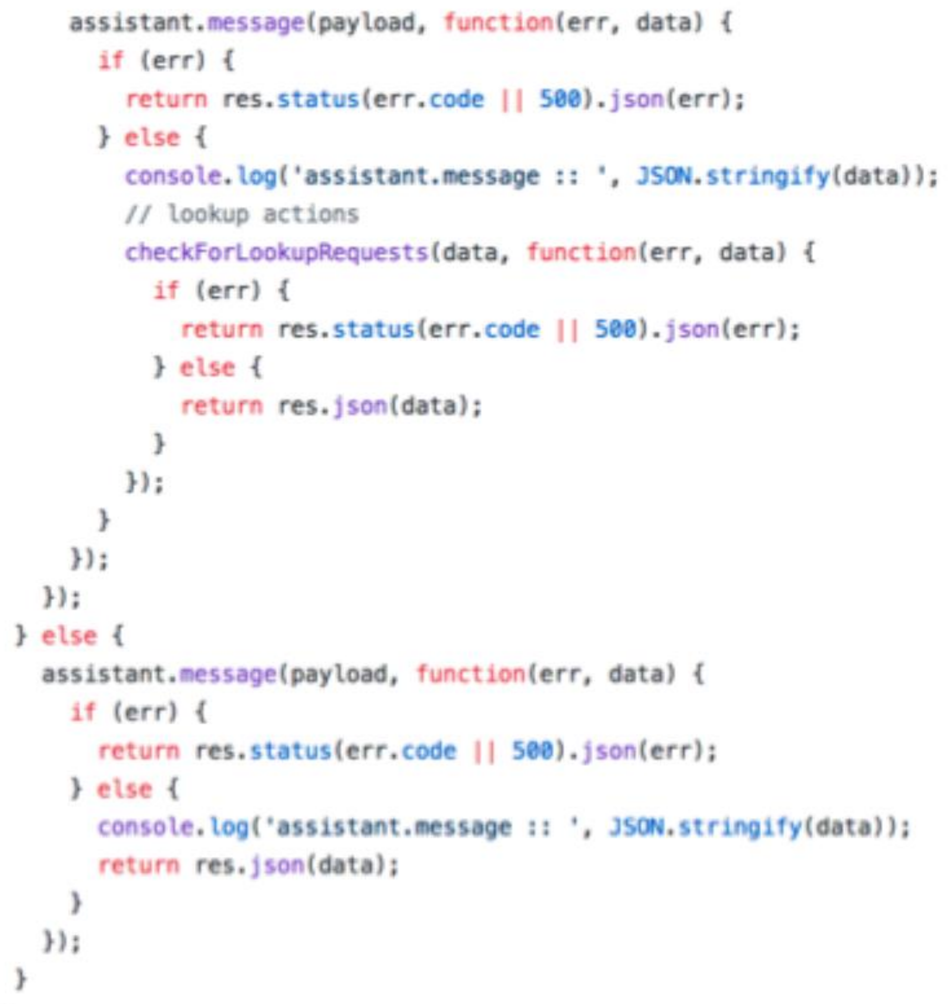
```
nlu.analyze(parameters, function(err, response) {
  if (err) {
    console.log('error:', err);
  } else {
    const nluOutput = response;

    payload.context['nlu_output'] = nluOutput;
    // console.log('NLU = ', nlu_output);
    // identify location
    const entities = nluOutput.entities;
    let location = entities.map(function(entry) {
      if (entry.type == 'Location') {
        return entry.text;
      }
    });
    location = location.filter(function(entry) {
      if (entry != null) {
        return entry;
      }
    });
    if (location.length > 0) {
      payload.context['Location'] = location[0];
      console.log('Location = ', payload.context['Location']);
    } else {
      payload.context['Location'] = '';
    }
```

Invoking Discovery Service



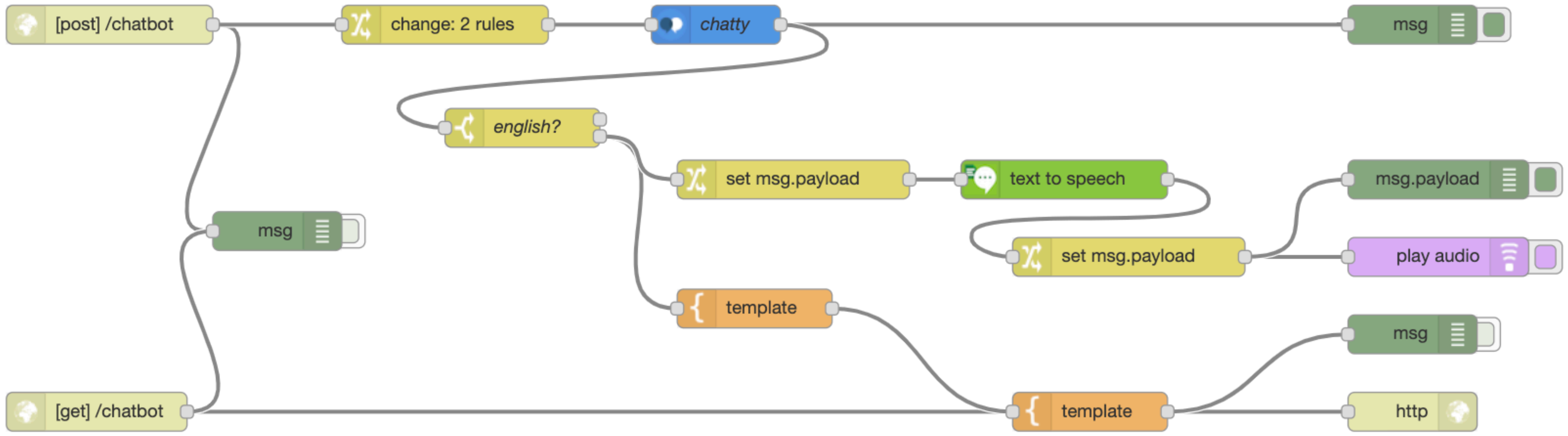
```
    });
} else if (data.context.action.lookup === DISCOVERY_ACTION) {
  console.log('************** Discovery *************** InputText : ' + payload.input.text);
  let discoveryResponse = '';
  if (!discoveryParams) {
    console.log('Discovery is not ready for query.');
    discoveryResponse = 'Sorry, currently I do not have a response. Discovery initialization is in progress. Please try aga
    if (data.output.text) {
      data.output.text.push(discoveryResponse);
    }
    // Clear the context's action since the lookup and append was attempted.
    data.context.action = {};
    callback(null, data);
    // Clear the context's action since the lookup was attempted.
    payload.context.action = {};
  } else {
    const queryParams = {
      natural_language_query: payload.input.text,
      passages: true
    };
    Object.assign(queryParams, discoveryParams);
    discovery.query(queryParams, (err, searchResponse) => {
      discoveryResponse = 'Sorry, currently I do not have a response. Our Customer representative will get in touch with yo
      if (err) {
        console.error('Error searching for documents: ' + err);
      } else if (searchResponse.passages.length > 0) {
        const bestPassage = searchResponse.passages[0];
        console.log('Passage score: ', bestPassage.passage_score);
        console.log('Passage text: ', bestPassage.passage_text);

        // Trim the passage to try to get just the answer part of it.
        const lines = bestPassage.passage_text.split('\n');
```

Calling Watson Assistant service



```
  assistant.message(payload, function(err, data) {
   if (err) {
    return res.status(err.code || 500).json(err);
   } else {
    console.log('assistant.message :: ', JSON.stringify(data));
    // lookup actions
    checkForLookupRequests(data, function(err, data) {
     if (err) {
      return res.status(err.code || 500).json(err);
     } else {
      return res.json(data);
     }
    });
   }
  });
 });
} else {
 assistant.message(payload, function(err, data) {
  if (err) {
   return res.status(err.code || 500).json(err);
  } else {
   console.log('assistant.message :: ', JSON.stringify(data));
   return res.json(data);
  }
 });
}
```

Calling Watson Assistant service



THANK YOU!