International Journal of Computing and Artificial Intelligence

E-ISSN: 2707-658X
P-ISSN: 2707-6571
www.computersciencejournals.com/ijcai
IJCAI 2025; 6(1): 164-170
Received: 19-02-2025
Accepted: 26-03-2025

Manan Buddhadev
Department of Interactive Media, Rochester Institute of Technology, Rochester, New York, USA

Corresponding Author:
Manan Buddhadev
Department of Interactive Media, Rochester Institute of Technology, Rochester, New York, USA

Building YAVIS: An intelligent conversational bot for calendar management using IBM Bluemix

Manan Buddhadev

DOI: https://www.doi.org/10.33545/27076571.2025.v6.i1c.147

Abstract
Natural Language Processing (NLP) and cognitive computing are the next big things in the computing world. Progress in these fields has been significant, and IBM's Watson is one of its biggest successes. The article covers a conversation agent I have made, yet another Very Intelligent System (YAVIS). YAVIS is a utility bot that helps add, update, and search calendar entries using the Google Calendar API. The paper will discuss how to implement such a system using IBM Bluemix.

Keywords: Natural language processing (NLP), conversational agents, cognitive computing, calendar management, IBM Bluemix, Google calendar API

Introduction
The core of YAVIS is IBM Watson Conversation [2], which identifies the user's different needs and intelligently replies to help fulfill those. It consists of three components, these are:
1. Intent
2. Entity
3. Dialog

Intents
For every user input, Watson attempts to identify the intent behind the user's query. Intents represent the different requests a user can make to the bot YAVIS. This process helps the Watson Conversation service determine the appropriate dialog flow for a response [2].

Entities
Once Watson knows of a user's intent, it tries to find the entities in the user's queries, which Watson uses to alter how it replies to a particular intent slightly [2].

Dialog
By identifying the two most essential components of the user's query, specifically intent, and entity, Watson can determine the most logical response to give the user. The dialogs are represented graphically in a tree with different branches for various intents we define [2].
This application uses IBM Watson Conversation with the Google Calendar API. It primarily serves as a utility that allows users to add or update their calendar entries. Additionally, YAVIS can retrieve calendar entries for specific dates and date ranges. For instance, users may inquire, "YAVIS, what is my schedule for tomorrow?" or "YAVIS, what does my day entail on the 10th of March?" Although a comprehensive exploration of application programming interfaces (APIs) has yet to occur, this initiative aims to enhance the understanding of LLMs while gaining insights into API utilization. Furthermore, other components of Watson, such as speech-to-text, text-to-speech, and Watson Conversation, are to be integrated to provide YAVIS with varied means of user interaction.
In addition to assisting users in managing their daily activities, YAVIS is intended to serve as a coding mentor. To facilitate this, Watson will be provided with data from Stack Overflow, a prominent question-and-answer platform encompassing approximately 6.7 million programmers. Users on this platform pose inquiries about diverse programming topics, receiving responses from fellow users. YAVIS will specifically be trained on questions related to Python programming. The relevant dataset, titled "The Python Questions

from Stack Overflow," is available on Kaggle and consists of three structured tables:

Questions table: Consists of the title of the question, the body of the question, when the post was created, the score given to the question, and the ownerId of the question [6].

Tags table: The tags table consists of tagID and the corresponding tag section [6].

Design Considerations
The application will use two major components to create the application. The first component is the Google Calendar API, which will be modified through the application, allowing data to be retrieved, updated, deleted, or inserted according to the user's command. The second component is the Watson Conversation API, which will interact with the user and the application's underlying Calendar API.

The Application
PyCharm is designated as the integrated development environment (IDE) for Python, which will be employed in the application's development. Preliminary testing has been conducted on a pre-implemented demonstration of a language translation tool to gain insight into the functionality of individual Watson services. Additionally, the necessary dependencies required to operate Watson services within the development environment have been successfully installed and configured.
Three key Watson services have been established for the project: text-to-speech, speech-to-text, and conversation. Testing the pre-implemented language translation demonstration has provided a comprehensive understanding of how these individual Watson services operate.
The language translator uses text-to-speech, speech-to-text, and conversation as its core components. The code used in the demo application is open-sourced and is being understood to make the YAVIS application. The language translation service can be used to recognize the language in which the text is written. IBM Watson provides language translation for five languages: Arabic, Spanish, Italian, French, and Brazilian Portuguese [3]. The Watson translator service is a powerful application that is also customizable. New words can be added to the Watson dictionary to translate words from the source to the target language.
Speech-to-text is a crucial component of the application we are building. Speech-to-text services can take input in two ways: streamed and recorded. The output of the Speech-to-text service is a text transcription of the audio with recognized words in JSON format [8].
The text-to-speech service can synthesize natural-sounding language from text [10]. The text derived from the computations processed by YAVIS can then be used to generate speech for a better user experience.

Google Calendar API
The Google Calendar API, alongside IBM Watson, is a vital application component. The researcher dedicated considerable time to understanding its fundamentals. The Calendar API provides a wide range of features that enable users to display, create, and modify calendar events and manage various other calendar-related objects [1]. The key functionalities of interest in the Calendar API include updating, inserting, and viewing specific user events.

IBM Watson Conversation
The three primary components used for training IBM Watson are Intents, Entities, and Dialog. Let's examine each component in more detail.

Intents
The application is developed with a set of Intents that users can reference while interacting with the Watson Conversation API. These Intents cover all operations typically performed with a calendar: Add, Update, Delete, View a reminder.

Entities
Along with the user's basic intents, such as adding, updating the calendar can include the following entities:
1. **Event/Reminder:** Set an event or a reminder
2. **List:** To retrieve and show the event-lists from the user's calendar
3. **People:** Represents the different people(if any) involved within the calendar event

Dialog
The dialog begins when YAVIS is greeted, after which YAVIS inquires how it can assist the user. Several dialogue examples are created for each intent to ensure that the Watson API comprehends the different ways a user might request various calendar actions. These examples serve as prompts for the user, all aimed at the exact intent of adding an event to the user's Google Calendar account.
- YAVIS please add Dr. X's appointment for tomorrow at 10 a.m.
- Please put a reminder to call Mom in the morning at 8 a.m.
- Please set up a meeting with John on Monday at 9 a.m.

YAVIS is also trained to tell jokes by integrating a subset of a small-jokes dataset from Kaggle [7].

Implementation
Watson Conversation
The application uses the IBM-provided Watson's Python SDK for development [11].

Intents
The application consists of seven intents described below:

Greeting
This is a start to a conversation with YAVIS. YAVIS is trained to greet the user. If the user inputs a greeting message such as "hi" or "hello." YAVIS would then greet the user by saying, "Hi, how may YAVIS assist you." To train YAVIS to greet, we have to provide a few examples such as "Hi YAVIS," "Hello," and "Hi there." Similarly, YAVIS would respond with "Hi, YAVIS is at your service" or "How may YAVIS assist you?"

Calendar_entry_add
This intent describes the user's desire to add an event or reminder to his Google Calendar. YAVIS is trained on the following example questions: "Add an event for tomorrow at 5 pm," "Create entry" Meeting with Bob" on 26th November 2017 at 5 pm," and "Mark a date on my calendar." This intent helps capture the user's purpose in adding an event or reminder to the calendar.

Calendar_next_event: This intent tells YAVIS to search for the coming events or to look for reminders. Some examples are "Do I have any upcoming events?" "Do I have any future events?" and "What are my next 10 events"

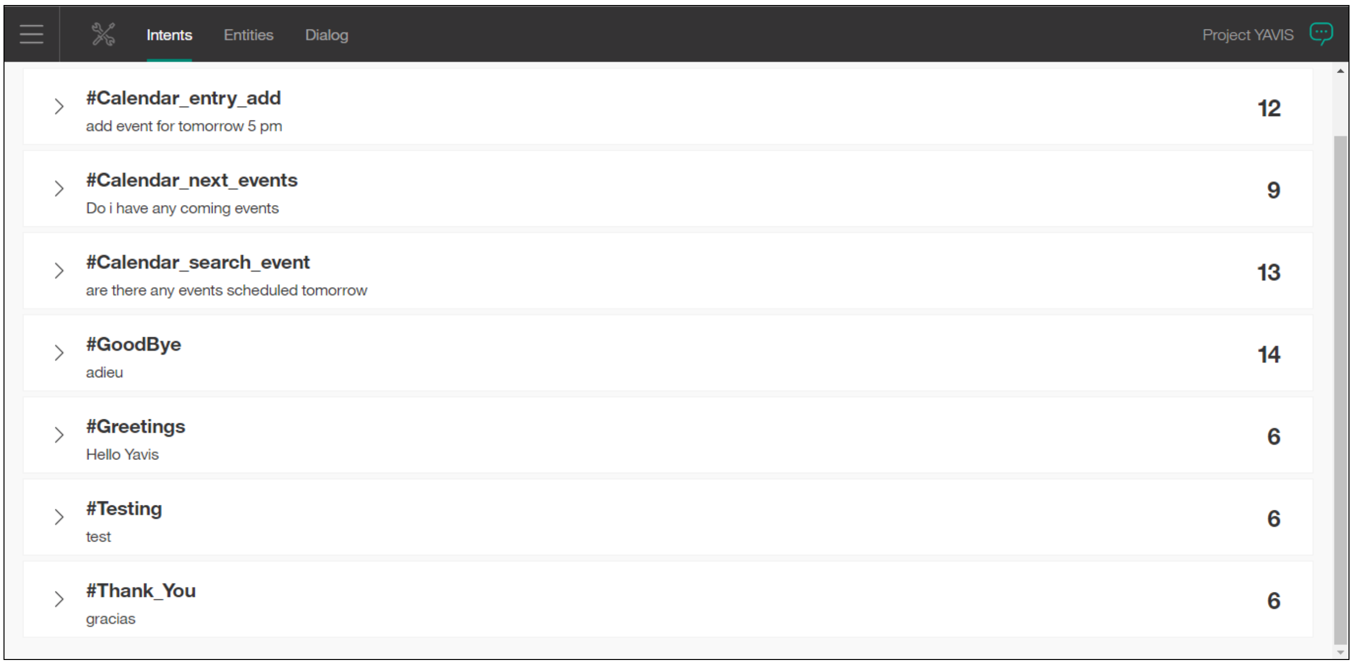
Calendar_search_event: This intent helps YAVIS understand whether the user wants to search calendar entries based on the time or date of an event.

Jokes: This intent describes that the user wants to hear a joke. We have added 100 jokes to YAVIS. Whenever this intent is understood by YAVIS, YAVIS responds by telling the user a joke. For example, if the user inputs, " Tell me a joke," YAVIS responds with a joke.

Thank_You: Thank_You describes the intent of acknowledging YAVIS. Whenever YAVIS understands this intent, it responds with a Thank-You response. Some examples are "Thanks for your help, YAVIS," "Thank you," and "Thank you, YAVIS." One sample response by YAVIS would be, "You are welcome."

GoodBye
The intent of "Good Bye" describes the user's desire to end the conversation with YAVIS. Some Examples on which YAVIS is trained are "farewell," "Thanks. Bye," and "Adios."

The intents created on Bluemix are shown below.



#Calendar_entry_add add event for tomorrow 5 pm	12
#Calendar_next_events Do i have any coming events	9
#Calendar_search_event are there any events scheduled tomorrow	13
#GoodBye adieu	14
#Greetings Hello Yavis	6
#Testing test	6
#Thank_You gracias	6

Fig 1: Intents for YAVIS

Entities
Entities describe objects the user can interact with within the chat application system. YAVIS can detect entities whenever they are mentioned in a conversation. Some of the entities created by us are described below:

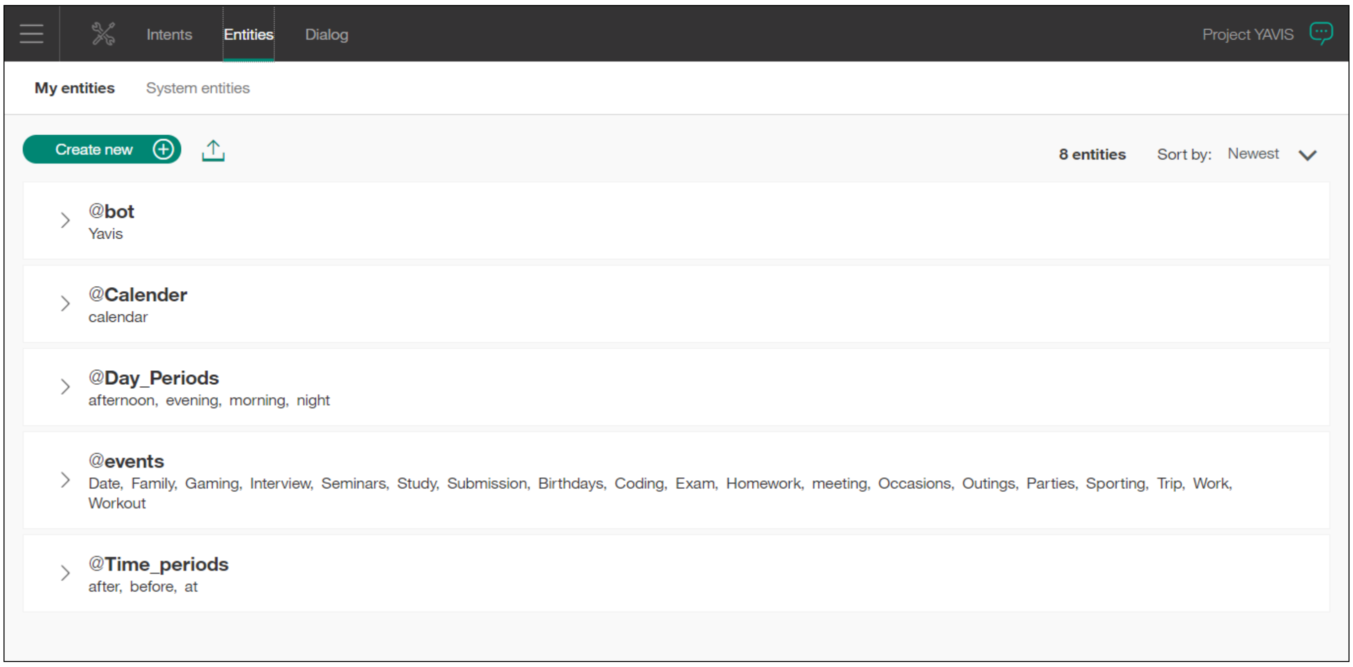
Bot
The bot describes YAVIS. Hence, whenever someone mentions YAVIS in the input, we are referring to the Watson system.YAVIS understands it being mentioned in a conversation.

Calendar
The calendar is an entity used by the user to set reminders and events and to look for events. It allows YAVIS to understand which entity to manipulate when a user asks a question.

Day_Periods
Day_periods help us describe the day intervals. A day is divided into four intervals: Morning, Evening, Night, and Afternoon.

Events
YAVIS uses this to register which events to set or look for in the calendar.

Time_Period
This helps us to understand if the event has occurred before, after, or in a given time frame. It helps YAVIS to refine searches based on this entity
In addition to these entities, we have used Watson's built-in system entities, such as sys-date, sys-time, and sys-numeric, to recognize date, time, and numbers when interacting with users.



@bot
Yavis

@Calender
calendar

@Day_Periods
afternoon, evening, morning, night

@events
Date, Family, Gaming, Interview, Seminars, Study, Submission, Birthdays, Coding, Exam, Homework, meeting, Occasions, Outings, Parties, Sporting, Trip, Work, Workout

@Time_periods
after, before, at

Fig 2: Entities for YAVIS

Dialogs

Besides intents and entities, dialogs are important in how the conversation will continue. For YAVIS, the dialog begins with a greeting, adding an event, and searching calendar events. The conversation is carried on based on the context of the conversation. A dialog will first take input from the user and be used to generate a response from YAVIS. Based on the response, the user inputs new information. YAVIS processes this to create a new response again. This dialogue continues until it reaches an end. The dialogue in Bluemix is defined as a linked data structure. A node is created every time the user inputs some information to YAVIS. The node is then linked to a new node based on the results produced by YAVIS. The chain of nodes then creates a dialog between the user and YAVIS.

Figure 3 shows some of the start nodes of the dialog between YAVIS and the user. One of these nodes is selected based on the user's input. After choosing the initial node, the path is chosen based on the user's responses. Once the dialog ends, the user can again instantiate YAVIS by providing input to YAVIS. Again, based on the user's input, a node is selected from the figure below.

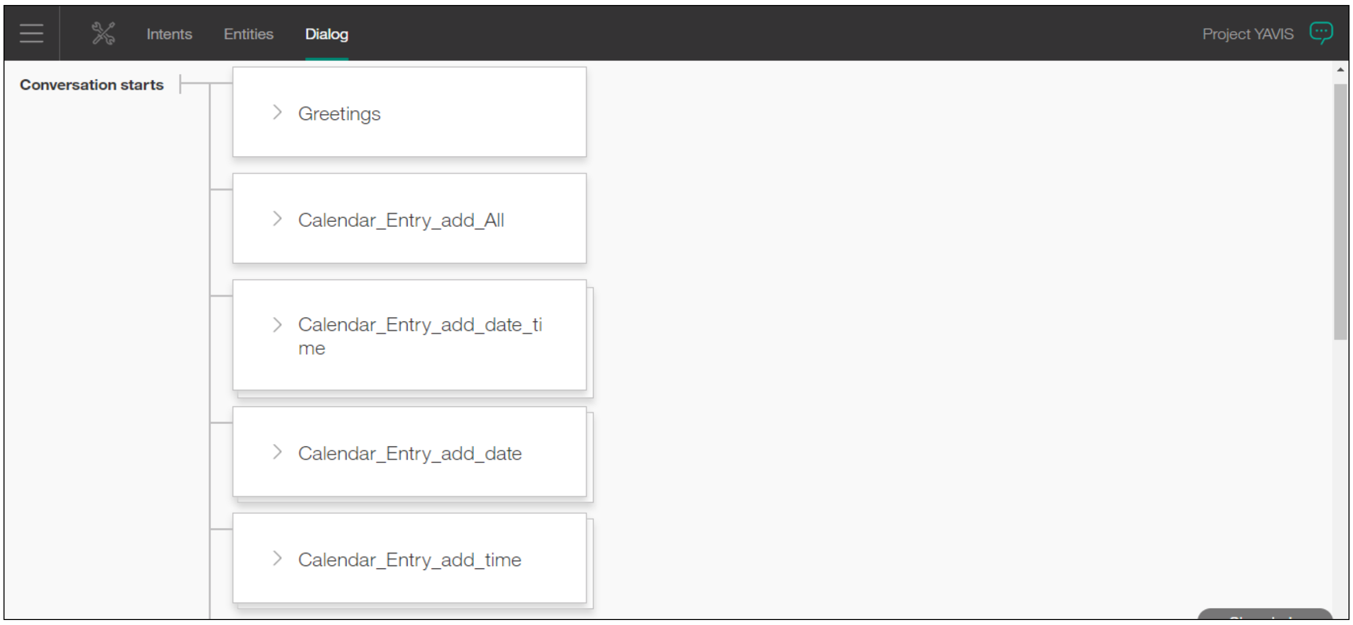
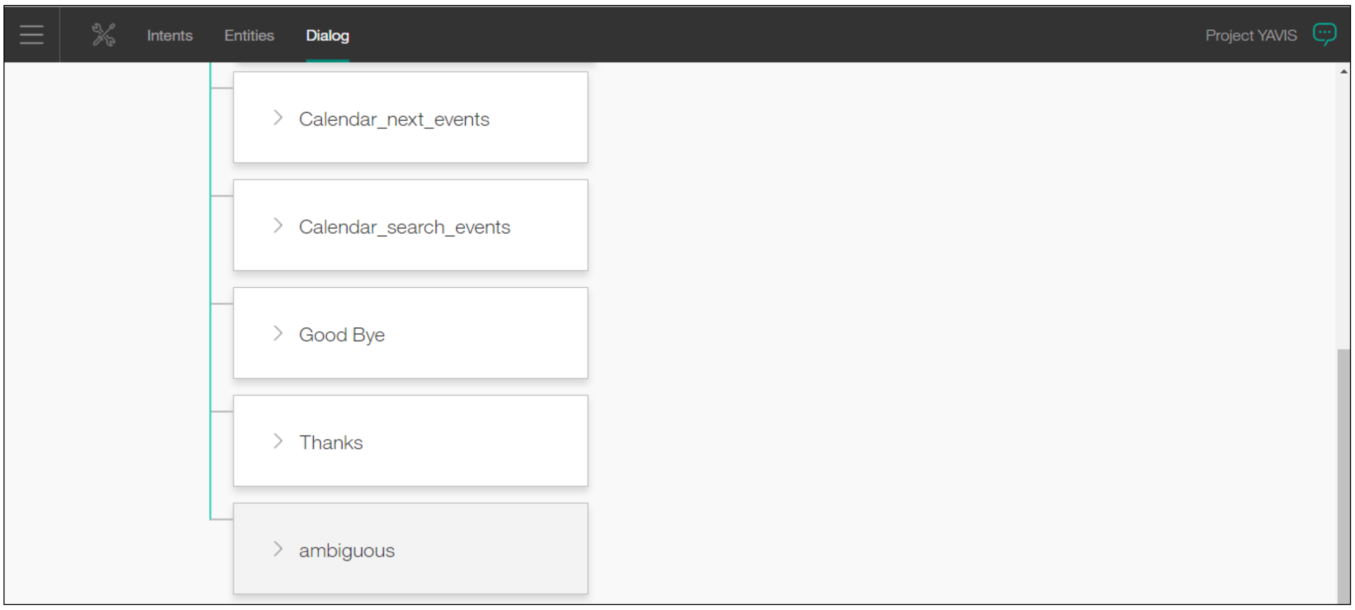


Fig 3: Dialogs for YAVIS-1



Calendar_next_events

Calendar_search_events

Good Bye

Thanks

ambiguous

Fig 4: Dialogs for YAVIS-2

Text-to-Speech

The application employs the Text-to-Speech Watson API to convert YAVIS's text responses into speech, enhancing the interface's user-friendliness and appeal. Additionally, the system allows for gender assignment to YAVIS by selecting either a male or female voice.

Using the Google Calendar API

Google Calendar API helps communicate with users' Google calendars. We can use it to add new reminders and search for events in the calendar. Leveraging the power of Watson conversation API, Google Calendar understands the event, its day and time, and whether the user wants to search for or add an event.

Google Calendar relies heavily on clues from Watson's conversation to decipher the appropriate action it needs to take. As a user's reply, for example, searching for an event can have different details such as date, time, number of events, or even a mixture of those. Because of this, many different scenarios were considered to make calendar API take proper action. Even when provided with the date and time, the user's requirement changes with words such as "before 5 pm", "after 5 am", or "on 23rd June," and hence the search space for events in the user's calendar changes appropriately. The application worked on all such natural language nuances for the same intent: adding or searching for an event.

To enable the application's Google Calendar API, the developers must set up a project in the Google Developer's console and activate the Google Calendar API. After creating the project, they need to generate credentials using the OAuth client ID, which produces a JSON file named client_secret.json. The developers should place this JSON file in the directory where the YAVIS project resides to facilitate communication with the calendar API. Since they developed the source code using Python, they must install the google-api-python-client library to access the various functions provided by the API. With the client_secret JSON file in place and the Google Calendar library installed, the API can assist users with various calendar requests.

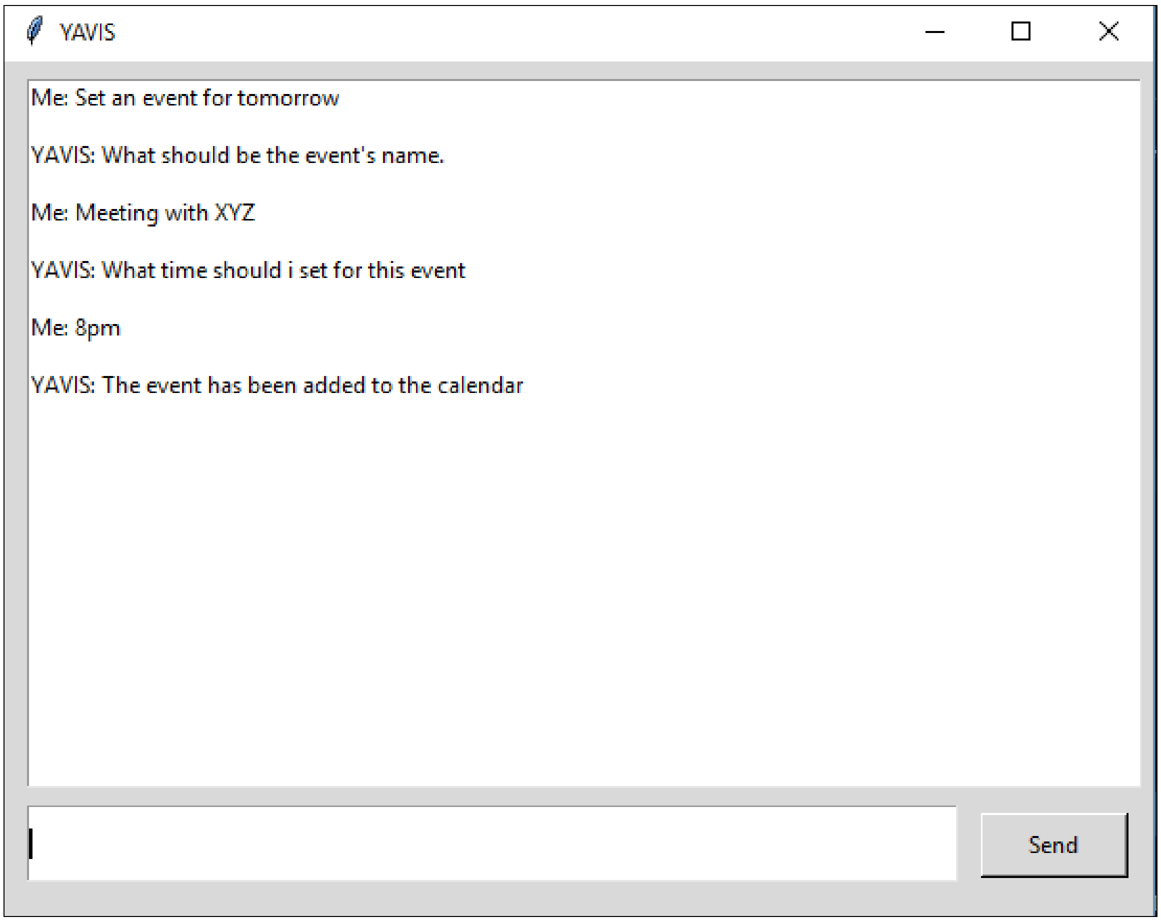
A couple of issues were faced while using calendar API; the biggest was the timezone issue. Using the DateTime class from Python to work with DateTime objects, it works on UTC schedule, and since the user for testing the application was in Eastern time, it created and searched for events with 4 hours offset. Additionally, even Watson conversation will work by default on UTC. Hence, extra care was taken to convert the timezone from UTC to Eastern time. The other problem, but a tiny one, was changing the initial calendar API to read-only access for the calendar to read-write access.

Graphic User Interface (GUI)

For the front end, we have developed a graphic user interface (GUI) in Python using a tool called PAGE [4, 5]. PAGE is a simple drag-and-drop UI creation tool for Python. While using PAGE, a user must create a new project and a new top-level box that will act as a canvas with all the components. The application uses the scrolled text box to display the user input and YAVIS's response to that input. Additionally, the application uses the EntryBox to read the user input and a Button to send the input to Watson.

After establishing the basic framework, the developer read the input from the EntryBox. After several attempts, they successfully retrieved the user input. Subsequently, the developer needed to post the same response in the Scrolled Textbox and learn how to use the Insert method. Once both the TextBox and Entry functioned properly, they integrated the back-end code with the front-end.

After integrating the search functionalities into the code, the 'Return' key was bound with the 'Submit' button to increase ease of use.



Me: Set an event for tomorrow

YAVIS: What should be the event's name.

Me: Meeting with XYZ

YAVIS: What time should i set for this event

Me: 8pm

YAVIS: The event has been added to the calendar

Fig 5: Conversation with YAVIS

Result & Conclusion

Based on the conversation with YAVIS, it can add events to the calendar and search for current events in the Google Calendar. There are instances where YAVIS fails to capture the intent or gets confused with the entities. For example, if the user inputs "Show me five events after 6 pm". Then, YAVIS cannot understand the difference between numeric values 5 and 6. It returns 5 and 6 as numeric values, whereas it should return 6 pm as time and five events as entities. This does not happen with date. Watson cannot recognize differences between date and time since it lacks the necessary training in the initial model.

Furthermore, there are cases where YAVIS is unable to understand the user's intent and is unable to respond correctly. For these reasons, we need to add more forms of questions for which Watson would train YAVIS to understand the intent of questions posed by users.

References

1. Google Developers. Google Calendar API overview [Internet]. Mountain View (CA): Google LLC; 2025 Apr 1 [cited 2025 Apr 20]. Available from: https://developers.google.com/workspace/calendar/api/guides/overview
2. IBM Corporation. System entities - IBM Watson Assistant [Internet]. Armonk (NY): IBM Corporation; 2025 Jan 13 [cited 2025 Apr 20]. Available from: https://cloud.ibm.com/docs/assistant?topic=assistant-system-entities
3. IBM Corporation. Supported languages - IBM Watson Assistant [Internet]. Armonk (NY): IBM Corporation; 2024 [cited 2025 Apr 20]. Available from: https://cloud.ibm.com/docs/assistant?topic=assistant-language-support
4. PAGE Developers. PAGE - Python Automatic GUI Generator [Internet]. Version 8.0. SourceForge; 2025 [cited 2025 Apr 20]. Available from: http://page.sourceforge.net/
5. SourceForge PAGE. PAGE - Python Automatic GUI Generator [Internet]. Version 8.0. SourceForge; 2024 Dec 9 [cited 2025 Apr 20]. Available from: https://sourceforge.net/projects/page/
6. Stack Overflow. Python Questions from Stack Overflow [Internet]. Kaggle; [cited 2025 Apr 20]. Available from: https://www.kaggle.com/stackoverflow/pythonquestions
7. Moudgil A. Short Jokes [Internet]. Kaggle; 2017 [cited 2025 Apr 20]. Available from: https://www.kaggle.com/abhinavmoudgil95/short-jokes
8. IBM Corporation. IBM Watson Speech to Text [Internet]. Armonk (NY): IBM Corporation; [cited 2025 Apr 20]. Available from: https://www.ibm.com/products/speech-to-text
9. Stack Overflow. Stack Overflow [Internet]. [cited 2025 Apr 20]. Available from: http://stackoverflow.com/
10. IBM Corporation. IBM Watson Text to Speech [Internet]. Armonk (NY): IBM Corporation; [cited 2025 Apr 20]. Available from: https://www.ibm.com/products/text-to-speech
11. IBM Corporation. watson-developer-cloud/python-sdk [Internet]. GitHub; [cited 2025 Apr 20]. Available from: https://github.com/watson-developer-cloud/python-sdk

12. Beal V. API (Application Program Interface) [Internet]. Webopedia; 1996 Sep 1 [cited 2025 Apr 20]. Available from: https://www.webopedia.com/definitions/api/Webopedia +3
13. Basu K, Abdelaziz I, Chaudhury S, Dan S, Crouse M, Munawar A, Kumaravel S, Muthusamy V, Kapanipathi P, Lastras LA. Api-blend: A comprehensive corpora for training and benchmarking API LLMs. arXiv preprint arXiv:2402.15491. 2024 Feb 23.
14. Sharp RD, Bocchieri E, Castillo C, Parthasarathy S, Rath C, Riley M, Rowland J. The Watson speech recognition engine. In: 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing; 1997 Apr 21; Munich, Germany. Vol. 5. Piscataway (NJ): IEEE; 1997. p. 4065-4068.