**Table of Contents**

**1.0 Introduction**

**1.1 What is Webex?**

Webex is a cloud-based suite of tools that enables online meetings, video conferencing, cloud calling, and collaboration. It provides users with secure, high-quality communication and collaboration capabilities, making it a popular choice for both businesses and individuals.

**1.2 What is the Webex API?**The Webex API allows developers to integrate Webex functionalities into third-party applications. This API enables developers to manage Webex rooms, participants, messages, and user information programmatically. It provides RESTful endpoints that facilitate access to various features of the Webex platform.

**1.3 Developing Third-Party Apps Using Webex API**

Third-party apps can be built using the Webex API to extend Webex’s capabilities and integrate them with external systems or custom applications. This involves using RESTful calls to interact with Webex services such as managing rooms, sending messages, and retrieving user information.

**1.4 Steps for Webex API Application Development**

To build an application using the Webex API, developers need to:

* **Obtain an OAuth token** for authenticating API calls.
* Use **REST API endpoints** to interact with Webex services.
* Develop the user interface and backend logic to handle API requests and responses.
* Ensure proper testing and deployment of the application.

**2.0 Objectives**

The primary objective of this project is to develop a problem-solving tool that leverages the Webex API to interact with Webex's conferencing and collaboration services. This tool is designed to help users efficiently manage and troubleshoot various aspects of their Webex environment. Specifically, the tool aims to provide the following key functionalities:

1. **Server Connectivity Check**: Verifying the accessibility of the Webex API by ensuring a stable connection to the Webex servers.
2. **User Information Retrieval:** Displaying detailed user information, including full name, username, and email address, to give users a comprehensive view of their Webex account.
3. **Room Information Display:** Listing the rooms the user is part of, with relevant details such as room ID, title, creation date, and last activity, providing insight into recent collaboration activities.
4. **Room Creation:** Allowing users to create new Webex rooms for meetings or team collaboration, streamlining the process of initiating group interactions.
5. **Message Communication:** Enabling users to send messages to Webex rooms, facilitating quick and efficient communication within their organization.

The application will undergo rigorous testing to ensure all functionalities work as intended. The development process will be thoroughly documented to ensure the tool’s reliability and usability in various scenarios. Ultimately, this project aims to create a valuable resource for Webex users while offering insights into API-based application development.

**3.0 Development of the Application**

**3.1 Tools and Technologies**

* **Programming Language**: Python , HTML, Javascript
* **API Documentation**: Webex API documentation for reference and guidance.
* **Version Control**: GitHub for source code management and version control.

**3.2 Application Development Process**

The following steps were followed to develop the Webex API Troubleshooting Tool:

**3.2.1 Token Authentication**

**A computer screen shot of text

Description automatically generated**

**A screenshot of a computer screen

Description automatically generated**

The application prompts the user to input their OAuth token, which is required for authenticating subsequent API requests to Webex. The token allows secure access to Webex resources associated with the authenticated user.

**3.2.2 Test Connection**

**A computer screen with text

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

The application tests the connection to Webex servers. Upon successful connection, the user receives an acknowledgment that confirms the communication with Webex API is established.

**3.2.3 Display User Information**

**A screen shot of a computer code

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

The tool retrieves and displays the user's profile information from Webex, such as:

* Name
* Org ID
* Email address This is accomplished by calling the Webex API endpoint responsible for fetching user details.

**3.2.4 Display Rooms**

**A computer screen with text

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

The application lists five rooms the user has recently participated in, along with the following information:

* Room Name
* Room ID
* Date of creation
* Date of last activity

**3.2.5 Create a Room**

**A computer screen shot of text

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

The tool provides an option to create a new Webex room. The user inputs a room title, and the application uses the Webex API to create the room. A confirmation message is displayed upon successful creation.

**3.2.6 Send Message**

**A computer screen shot of text

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

This option allows the user to select a room from the list of available rooms and send a message to that room. The application interacts with the Webex API to send the user's message to the selected room.

**3.3 Code Repository**

The full source code for the Webex API Troubleshooting Tool is hosted on GitHub at the following link: [Insert GitHub repository link here].

**4.0 Testing of the Application**

**4.1 Test Plan**

The testing phase of the application involved verifying each feature to ensure it worked as expected. The following features were tested:

* **Token Authentication**: Verifying that the OAuth token allows secure API access.
* **Server Connection**: Ensuring the application can establish a connection to Webex servers.
* **User Information Retrieval**: Confirming that user details are accurately retrieved from Webex.
* **Room Listing**: Checking the correct display of recently accessed rooms.
* **Room Creation**: Verifying the successful creation of new rooms.
* **Message Sending**: Ensuring that messages are sent to the correct rooms.

**4.2 Test Results**

Below are the results of the testing:

* **Token Authentication**: Successfully authenticated using the provided OAuth token.

A computer screen shot of a computer screen

Description automatically generated

* **Server Connection**: The connection test to Webex servers passed without any issues.

A screenshot of a computer

Description automatically generated

A black rectangular box with white text

Description automatically generated

* **User Information Retrieval**: User data (name, org id, nickname) was correctly fetched and displayed.

A screenshot of a computer

Description automatically generated

* **Room Listing**: Recently accessed rooms were displayed, showing correct information about room ID, title, creation date, and last activity.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* **Room Creation**: A new room was successfully created, and a confirmation message was displayed.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* **Message Sending**: Messages were sent successfully to the selected rooms.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**4.3 Testing Conclusion**

All core functionalities of the Webex API Troubleshooting Tool worked as intended. No critical bugs were encountered, and the application was able to successfully interact with Webex API endpoints. Minor improvements could be made to enhance user experience and error handling.

**Conclusion**

This project effectively created a problem-solving tool with the Webex API, offering essential features for handling Webex resources. The tool enables users to check server connection, fetch user details, control rooms, and communicate via API integration. These characteristics make it easier to programmatically interact with Webex services, providing a more efficient option for troubleshooting and handling conference environments.

Throughout the development process, various difficulties were faced, especially in managing OAuth token authentication and guaranteeing consistent communication with the Webex API endpoints. Careful implementation is needed for token management to handle expiration and ensure secure communication. Furthermore, overcoming API rate limits and grasping Webex's API response formats were important technical challenges. Nevertheless, by conducting tests and making incremental enhancements, these obstacles were effectively tackled, resulting in a reliable and operational application.

This project gave important insight into integrating APIs, emphasizing the need to understand technical needs and real-world applications of APIs. Collaborating with Webex API has further improved the comprehension of efficiently managing cloud-based services through automation, thus improving effectiveness in a practical collaboration environment. In general, the project met its goals and produced a useful tool that could help Webex users with troubleshooting and management duties.

**References**