**ZoomInfo Chat**

ZoomInfo Chat is a conversational marketing platform. ZoomInfo Chat allows customers to segment, target and activate custom conversations for various audience segments on their website and engage them in real-time

**Application Flow Diagram**

START

Phone Number

Choose Option

Email

END

Support team Online

No

Yes

Support communication

**USER FLOW**

1. The Chatbot greets with a welcome message and provides you with few options to choose from. A User ID and a communication channel are established.
2. The User ID is authenticated (/pusher/presence/auth/visitor)
3. A series of API’s are triggered (delivered/read) to confirm Chatbot messages are visible on the app and viewed by the User.
4. If an option is chosen the User is asked for a valid business email address and once the email is valid someone from the support team is added to the chat.
5. An API (pageVisit/spentTime) is called in intervals and during user input to track the events
6. If support team is not available users phone number is requested.
7. Chat ends with a Thank you message once the Phone number is provided.
8. User has the option to Refresh the Chat to start a new Conversation at any point of time.
9. User has the option to view all the communications made previously.

**API FLOW**

1. Initial channel, User ID and other settings is provided by the API when the page is opened

<https://insentstaging.api.insent.ai/getuser?url=staging0.web-test.insent.ai%2Ffe-assignment>

authorization key must be provided to access the API and configured URL needs to be sent in the query param to get a channel ID.

1. On click on the chat bot icon a API is triggered to fetch the previous messages and current message that has to be shown to the user.

https://insentstaging.api.insent.ai/user/channels/private-LZpg36QFSEByPbXpY16574404494631657440452136

1. A instance of Pusher client is created with the provided User ID
2. {"event":"pusher:connection\_established"} confirms the connection.
3. During Pusher client creation authorization URL has to be provided which takes userid as query parameter.
4. Need to subscribe to the channel provide in first step.
5. {"event":"pusher:subscribe"} is triggered to subscribe to the cannel
6. {"event":"pusher\_internal:subscription\_succeeded} is triggered once the subscription is complete.
7. {"event":"client-widget-message"} is triggered when a message is sent from the client side
8. {"event":"server-message"} gives the response for the clients message