Gaman

Live Chat

##### Document history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Person | Version no. | Description |
| 2022-09-15 | Gaman | 1.0 | First version of the document |
| 2022-09-18 | Gaman | 1.1 | Review of the document |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Table of Contents

[Table of Contents 1](#_Toc114077270)

[Introduction 2](#_Toc114077271)

[Solution Overview 2](#_Toc114077272)

[How it works 4](#_Toc114077273)

[Chat features 5](#_Toc114077274)

[Future upgrades 6](#_Toc114077275)

# Introduction

One of the trends observed in the customer services is that interaction starts from digital channels. Searching for contact usually takes place via corporate website. Many organizations are looking for a chat solution that can be easy to deploy and has capabilities to integrate with the bot.

At Gaman we also got to a point where we started to review the market and see what we can use to have a chat functionality in our organization. Before we started our research, we defined some requirements that needed be fulfilled. Below you may find major requirements that our chat needed to meet our needs:

* Chat solution should be free of charge solution or use already licensed tools – we don’t want to pay additional fees
* Chat should fit the tools that we have in our organization, internally we use Webex Teams for communication
* Solution should have the capability to integrate with bot

Our team reviewed the integrations available on the market and most of them are either paid solutions or require usage of specially designed applications for chat purposes. Because we didn’t get expected results, we have reviewed previous solutions created by our team and we found:

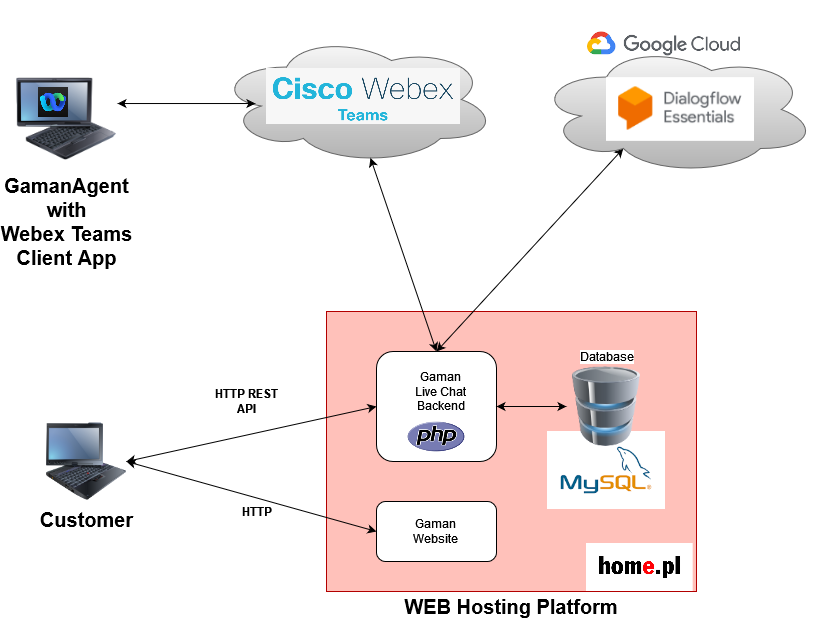
* UCCX Bubble chat form can be used on the frontend side, some small adjustments would be needed
* In our organization we use Webex Teams as our internal communicator. Chat conversations should be processed as internal chats
* In the past we built bot integration with WebEx Teams. From this part we can use or rewrite the code for both integration
* We have a hosting platform that can run PHP or Python code for the backend. We also have a database to store chat data

Looking at those points we decided to create our own custom solution. The question is why? Most of the major components for chat were created by Gaman team in the past. Some components needed code refactoring and some components needed to be created from scratch. The time needed to create the solution should be quick and only required our time.

In the next chapters we would like to describe effects of our work. You will find the description of the solution that we created, and which we are using on our website.

# Solution Overview

Below is the diagram that shows all the elements that were used to build the described chat functionality:



At the diagram we see the following components:

* Web hosting platform that holds the following elements:
  + Web page – customer website
  + JavaScript chat library – component added to website that implements APIs delivered by Live Chat backend
  + Live Chat Backend – written in PHP, component created by Gaman responsible for:
    - Publishing API for chat form placed on the website
    - Delivering integration features with Cisco Webex Teams and Google Dialogflow
    - Delivering integration with internal database
  + Database – to store data needed for integration
* Webex Teams – main platform used for conversation. For the integration purposes single account in Webex Teams platform is needed. The account needs to have to options enabled:
  + Integration – needed for issuing Webex Teams API requests and for delivering customer messages to Webex Teams rooms
  + Bot – needed in the case of integration with the Bot – identify messages created by bot.
* (Optional) Dialogflow ES bot – needed for non-agent conversations

|  |  |
| --- | --- |
| Note | In the Gaman deployment we used Web hosting platform called hold home.pl that delivers:   * Code executions – in the presented solution the backend is written in PHP but can be replaced with other programming language * Database – in the presented solution the MySQL database was used. It can be replaced by any other relational database   This platform can be replaced by any other platform available on the market, but it must be able to deploy database and backend components.  It is also possible to move database and backend component outside the hosting platform. |

# How it works

All begins when the customer clicks the start chat button and selects conversation topic. At the same time chat form communicates with the backend and performs the following tasks:

* Creates new Webex Teams room for the conversation
* Adds participants to newly created room (if bot is enabled, then bot is also added to room)
* Adds webhook to newly created room for notification that the new message was added to room
* Ask external Web Service for geolocation of the clients IP address

If all steps are successful, the chat session is created, and conversation can begin.

To deliver treatment agents from Gaman side can use any Webex Teams client which is available on multiple platforms ex. PC, mobile, etc.

During the session messages from backend are retrieved using pooling mechanism. This means that chat form periodically calls API and checks if agent from Gaman side has replied to message. If the messages are available, then they are presented to customer.

The backed has also two mechanisms that track customer inactivity:

* Backend checks the time from when the last message has been sent to Gaman representative. If the time exceeds the configured threshold, then the session is treated as inactive, and chat automatically closes the session
* Chat form from the client side periodically checks if new messages are available for clients. Those requests are treated as heartbeats that update the session lifetime. If the backend detects loss of the messages, the session is treated as inactive.

Here is the sample conversation session visible from Webex Teams side.

Obraz zawierający tekst

Opis wygenerowany automatycznie

# Chat features

Below is the list of features that the chat gives to the owner:

* Chat form supports 2 types of chat:
  + Anonymous – chat starts when users click’s “start” button
  + Chat with required identification – chat where user needs to provide input data to start chat session: name, email etc.
* Time control
  + Ability to set open/closed hours
  + Ability to define holidays
  + Ability to define special days – days where chat is open for shorter periods of time.
* Bot integration – chat is currently connected to Google Dialogflow ES bot solution. This integration can be replaced with other AI available on the market (integration via API).
* Topic selection support – before chat can be initiated user needs to select conversation topic. Based on the user selection, participants are added to the new created Webex Teams room.
* Each chat session opens new room on Webex Teams. The room can be:
  + Personalized – 1:1 conversation – company representative can talk to customer
  + Group – 1:N conversation – multiple company representatives talk or track conversation to customer at the same. Client doesn’t know who is providing the answer to the question
  + Mixed – conversation starts as a 1:1 chat and then during the conversation it’s upgraded to group conversation – customer representative can add additional participants to the conversation in real time
* Every room has entire chat conversation (including bot conversation) between customer and company representative. It is easy review the history – history is kept up to 6 months
* Detection of customer inactivity and customer has left chat event

# Future upgrades

Presented solution has some place of improvements that can be added. Here is the list of features that can be added:

* Multilanguage support – the chat is designed to support single language. Additional languages can be added as static, configuration messages. To add flexibility the Google Translate service can be added so the static messages will be translated on fly. Also, conversation can be translated to native language of customer representatives
* Admin portal – some of the features available on page are configured using JavaScript files – work hours, holidays, special days etc. To add flexibility, those parameters should be moved to database and be changed via admin portal
* Ability to integrate with other chat communicators – to add flexibility to the solution and be able support platform the chat backend should integrate not only with the Webex Teams but also with other communicators like Slack