BAMBA Samuel
CORDAT-AUCLAIR Julien
Referent lecturer: DONSEZ Didier

An Accessible ChatBot

RICM4 Project - Polytech Grenoble

Academic Year 2017-2018

Introduction **How advantageous are Chatbots for services ?**

Definition

A Chatbot is a program capable to lead a conversation with a human being while providing information and/or automating a task. In today's world where messaging apps have become the most popular form of communication, Chatbots are one of the main tool business can use to provide an efficient and nimble customer experience.

Brief History

The Chatbot story began in 1996 with a bot named ELIZA. "She" was able to act as a psychotherapist, creating questions from the users answers in order to conduct an humorous discussion. It was only later in 2001 that a true assistant capable of performing tasks was created, it was SmarterChild. Available on AOL Instant and MSN Messengers, it provided a quick access to web information (news, weather, sports data,...) and many tools (calculator, translator, ...). SmarterChild is considered today as a main precursor of the famous Apple's Siri and Sanmsung's S Voice.

The Facebook Messenger App

In 2011, Facebook launched Messenger, a fast, free and internet based messaging tool. In a context where analysts were rightfully concerned that instant messaging could hit the operator's text and voice revenues, Messenger appeared as a promising contestant for WhatsApp and BlackBerry Messenger.

It was in April 2016, when Facebook decided to open its API[1] to developers, one of the biggest milestone for Chatbot had been set. The API allowed people to receive messages on any http url webhook[2] and send response messages or actions part of Messenger's RCS Messaging[3]. Subsequently, the creation of thousands of Messenger Bots; everywhere for any task.

The Handicap Reception Service

The Handicap Reception Service of the Grenoble University has the mission to ensure students in an handicap situation access to formation and student life participation. Located on the Saint-Martin d'Hères campus, the service receives people in need as well as volunteers.

Problems Encountered by the Service:

- With a very limited reception surface, the service tends to get crowded;
- The Service provides information forms on their website that people can print and pre-fill before coming to the building, but a very small portion of students actually do it.

By reducing the amount of people coming to the Service and easing the pre-filling form process, the aim of our Chatbot is to solve both these issues.

[1] In computer programming, an application programming interface (API) is a set of subroutine definitions, protocols, and tools for building application software.

[2] A webhook in web development is a method of augmenting or altering the behaviour of a web page, or web application, with custom callbacks.

[3] Rich Communication Services is a communication protocol between mobile-telephone carriers and between phone and carrier.

Chatbot Components **Breaking down of the different parts of our bot.**

Platforms

Input Channel - Messenger

A Chatbot require a channel to get the inputs messages from the user, as we introduced it above, we decided to choose Messenger because it is very welcoming to Chatbot development. Facebook provides developer account functions, allowing to listen for events on a Page's Messenger inbox. Once the page receives a message, a request will be generated towards our server.

Hosting - Glitch

Glitch.com is a free hosting open source platform for Node.js servers. As the free hosting options are very limited, Glitch.com is a very interesting site, with a vivid community and great options for in-real-time synchronous team development.

Database - MongoDB

When someone starts a conversation with our bot, we need to collect and save information about the user. In our database, we also save the advancement of the conversation so it can be put aside and continued whenever wanted. Again, we went for one of the most efficient options we could get for free (0.5 GB of storage).

API's MVC Architecture

Our API receives requests from the Messenger App and sends a post request to the Messenger Send API to answer accordingly.

Model

Our model is represented in the model.js file, it contains a schema for all the field that will be authorised for one user in our database. We used the mongoose npm package[4] to easily create our schema.

Controller

Our controller, controller.js, contains all functions related to database management. Looking for a contact with a specific id in the database, modifying one field, deleting a contact, ect..

View

Our View, app.vue uses the Frontend technology of Vuejs[5] to display our database in a table format. This webhook is our principal address meant for an usage by the HRS secretariat. We protect the database content with a password and allow the administrator to modify and see contacts who have talked with the bot.

[4]Mongoose is a MongoDB object modelling tool designed to work in an asynchronous environment.
[5]Vue.js is an open-source progressive JavaScript framework for building user interfaces.

Core of the Bot **Conversational Schema and State Graph explanation.**

Conversational Schema

One of the main job that has to be done during this project was to build a conversation tree that our bot would follow with any kind of user. To do so, two main paths have been used : the first one is about providing information about the Handicap Reception Service and the second one consists on filling in forms that the service would use to register students (either disabled or not). In addition, we decided to use buttons in order to make it easier for the user to chat with the bot, but we also had to implement some text fields, namely for the forms, where the user could type in information about him that the service would store. However, the bot never analyses these texts : he moves on to the next section of the tree.

State Graph

In our main code, app.js, the state graph is represented by a switch[6]. We firstly get the previous state of the user in the database then we call our function which will add the received text to the database if the state require it, then we call the send API to send the messages and buttons for next state and advance the state according to the user's answer.

[6]In computer programming languages, a switch statement is a type of selection control mechanism used to allow the value of a variable or expression to change the control flow of program execution via a multiway branch.

Accessibility

User specification

Since we created the Chatbot for the Handicap Reception Service, we can expect a user with any handicap.
Mainly physical, visual, hearing and mental health disabilities.

Tools and Choices

The Messenger platform has its own tools for people with disabilities.

- Keyboards Shortcuts allows to navigate the whole website;
- Screen readers compatibility;
- Online assistance through report forms;
- Text size persistence and contrast adjustment;
- A speech-to-text feature is currently being developed.

We made specific choices for our bot to provide a truly accessible experience. As a Keep-it-short-and-simple approach.

- Limitations of the user's possible inputs;
- Buttons for users to click;
- Online platform for computer and smartphones compatibility;

Conclusion

Thanks to monthly meetings with Ms. Marie Ballicco, director of the HRS Grenoble, we created a Chatbot that corresponds to the service. It was a very rewarding project in term of knowledge as we had to tackles a language, technologies and overall a field that we weren't familiar with. Building a customer based product in a team of 2 was a unique experience, it made us realised the importance of understanding the needs of the user and how we could meet them.

Links

- The Source code on Gitlab: https://gricad-gitlab.univ-grenoble-alpes.fr/Projet-RICM4/17-18/14/Chatbot
- Remixing the project on Glitch: https://glitch.com/edit/#!/melted-umbrella?path=app.js:1:0
- Secretariat App: https://melted-umbrella.glitch.me/
- Facebook Page: https://www.facebook.com/SAHGrenoble/

Bibliography

[1] Infographic on ChatBots History by Futurism.
https://futurism.com/images/the-history-of-chatbots-infographic/

[2] Financial Times Article by Tim Bradshaw, *Facebook launches Messenger app*, August 10, 2011.
https://www.ft.com/content/c1975e8a-c372-11e0-b163-00144feabdc0

[3] Handicap Reception Service website (French).
https://handicap.univ-grenoble-alpes.fr/fr/ressources/documents-et-formulaires/

[4] The Facebook Developers Platform, *Getting started*.
https://developers.facebook.com/docs/messenger-platform/getting-started/quick-start

[5] The Glitch Community Forum.
https://support.glitch.com

[6] The LinkedIn Learning Center, *Learning Vue js* by Michael Sullivan.
https://www.linkedin.com/learning/learning-vue-js

[7] The LinkedIn Learning Center, *Building RESTful web APIs with Node js and Express* by Emmanuel Henri.
https://www.linkedin.com/learning/building-restful-web-apis-with-node-js-and-express/

[8] Accessibility for People with Disabilities by Facebook.
https://www.facebook.com/help/141636465971794