

ÍNDICE

ÍNDICE	1
FASE 1	2
REGISTRO WEB	2
PROGRAMACIÓN DEL CHATBOT	2
INTENTS	2
SKILLS	3
FUNCIONAMIENTO DEL CHATBOT	6
LISTA APARTADOS DEL CHATBOT	8
FASE 2	11
CREACIÓN WEB Y ENLACE CON EL CHATBOT	11
FASE 3	15
CONFIGURACIÓN HERRAMIENTAS DE PRUEBAS	15
VARIABLES DE ENTORNO	15
DEPENDENCIAS DE SELENIUM WEB DRIVER Y DE CUCUMBER	17
CREACIÓN DE LAS PRUEBAS AUTOMATIZADAS	18
CASOS DE PRUEBA	22
ESCENARIOS PARA LOS CASOS DE PRUEBA	24
CAMBIOS DE DISEÑO	28
CAMBIOS DE DISEÑO EN EL CHATBOT	28
CAMBIOS DE DISEÑO EN LA WEB	30
SAP.HTML	31
PRUEBAS.HTML	32
MEMORIA.HTML	33
CHATGPT Y LAS INTELIGENCIAS ARTIFICIALES	35
CHATBASE	35

FASE 1

Registro web

El primer paso para poder proceder con la creación del Chatbot es registrarnos en la web de SAP Conversational AI (<https://cai.tools.sap/>). Aquí nos tendremos que registrar, y posteriormente elegir las características principales que va a tener el Chatbot. Vamos a seleccionar que se quiere que el bot ejecute las acciones programadas en un formato de “small talk”. Después se le añade un nombre y una pequeña descripción, seguido de la configuración de tipo de dato que se va a usar, en este caso no personal. Y finalmente, el bot será privado.

Registro

Datos personales

Nombre *

Apellidos *

Correo electrónico *

Establecer contraseña

Contraseña *

Vuelva a introducir la contraseña *

Términos y condiciones

☒ He leído y entendido los Términos y Condiciones de SAP Conversational AI. *

La declaración de privacidad está disponible [aquí](#).

*Obligatorio

[Registrarse](#)

SAP ID Service

 Existing Users | One login for all accounts: Get SAP Universal ID

Figura 1

4 Data Policy

Type of data

Select the most restrictive type of data used by your bot. Note that health-related data is not allowed.

- ☒ Non-personal
- ☐ Personal
- ☐ Sensitive Personal
- ☐ Health

Data retention: unlimited duration

Store conversation data

Select whether you want to store your user's conversations. Learn how this affects monitoring and connect.

- ☒ Store (recommended)
- Store your user's conversations to improve and monitor your bot.
- ☐ Do not store
- Do not store your user's conversations. This will heavily impact monitoring and connect features.

5 Bot visibility

- ☐ Public
- Your bot is open to everyone

- ☒ Private
- Your bot is private and accessible only by you and the developers you decide to share it with

[Continuar](#)

Figura 3

1 What do you want your chatbot to do?

☒ Perform Actions using conversational flows Intent-based

☐ Retrieve Answers from FAQ documents FAQ

2 Select predefined skills for your bot

☐ Greetings

☒ Small Talk

☐ Weather

☐ Customer Satisfaction

☐ Set Alarm

3 Create your bot

Bot name: /

Description (optional):

Topics (optionally categorize your bot with up to 6 topics to improve your bot training):

Try: [Customer support](#) | [Point and booking](#) | [Telecommunications](#)

Default language (your bot is multilingual, you can add more languages later on): Spanish Advanced

Figura 2

Programación del chatbot

Después de esto procedemos a la creación del chatbot. Para crear el bot vamos a separar dos conceptos principales: intents y skills.

Intents: Con esto vamos a definir las posibles frases que el usuario va a introducir al chatbot. Se van a separar por los bloques especificados por el esquema de preguntas que se va a programar. Por

ejemplo, para el apartado de “webs”, un usuario podría preguntar cosas como: “cuál es la web de la UAH”, “dime cual es la pagina web”, o “por favor, llévame a la web de la uah”. Pensando y añadiendo diversas preguntas relacionadas al tema que se quiere responder, podremos entrenar al bot para que sepa reconocer a lo que el usuario le pregunta.

Let's create your intent

mariachantal / tfg / webs

Your intent description (optional)

preguntas sobre la web de la UAH/EPS

Matching Strictness How does it work?

A detected intent with a confidence score strictly greater than the matching strictness is kept and returned in the JSON. Our recommendation is a matching strictness between 50 and 70.

0 70 100

Matching Strictness greater than 70 will decrease detection capacity

Cancel Create Intent

Figura 4

Expression

+ Add Language

Type an expression to add...

Import Expressions

You have 14 expressions suggested to enrich your intent

All			
<input type="checkbox"/>	informacion de la web		
<input type="checkbox"/>	pagina web		
<input type="checkbox"/>	web		
<input type="checkbox"/>	cual es la web de la eps		
<input type="checkbox"/>	cual es la web de la uah		

Figura 5

Cuanto más frases se introduzcan en cada intent, más inteligente se volverá el chatbot, y de esta forma podrá tener más opciones de reconocer la entrada de texto del usuario.

Skills: Dentro del apartado “Build”, se van a poder empezar a definir las “skills”. Una skill va a ser la encargada de reconocer las frases recogidas en los intent y proporcionar una respuesta. Para lograr esto, se deben seguir diferentes pasos, seguiremos centrándonos en el apartado “web”.

- 1) Creamos la skill. Aquí habrá que ponerle un nombre identificativo y elegir el tipo de skill que se quiere probar. Para todo este proyecto, se va a trabajar con el tipo “Floating”, que es la más indicada para las pequeñas conversaciones que vamos a querer tener con el chatbot.

Add Skill

Name

webs-build 245

ex. book-flight, checkout

Type

Business Floating Initialize

Floating skills complement your bot's core business skills (example: small talk).

Activation ☒

Your skill is active. If you deactivate it, your bot will no longer use it when chatting.

Title

Your bot automatically uses this skill title when disambiguating during chat. If you don't specify one, your bot will display: webs-build

Type a short title for this skill's quick reply button...

Figura 6

- 2) Definimos la sección “trigger”. Un trigger actúa como un capturador, el cual será el encargado de analizar el intent que se ha definido previamente y relacionarlo con la skill que estamos definiendo. Por ejemplo, al haber creado antes el intent “webs”, tendremos que decir que esta skill se lanzará si ese intent está presente, como se puede observar en la siguiente figura 7.

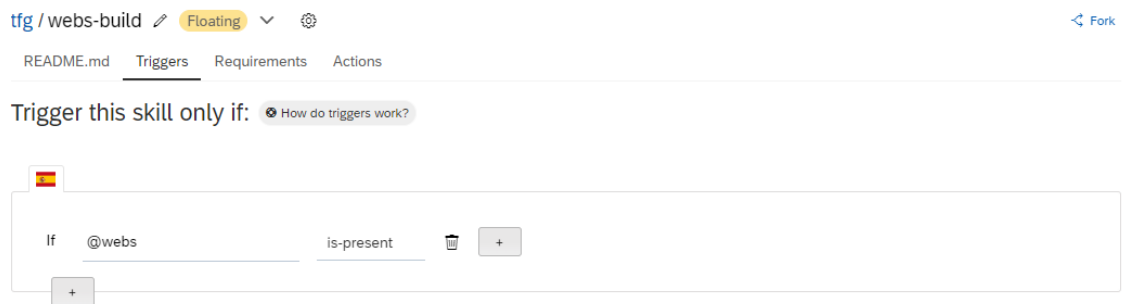


Figura 7

- Definimos la sección "Actions". En esta sección vamos a poder configurar cómo va a ser el mensaje que vea el usuario una vez escriba su pregunta.

En primer lugar, se selecciona la opción "Choose message Type". Dentro de esta opción, podremos elegir la forma de interacción. Este chatbot va a estar configurado mediante botones, para que aparte de proporcionar al usuario la información que busca, pueda acceder a enlaces que complementen esa información. Todas las opciones de respuesta se pueden consultar en la Figura 9.

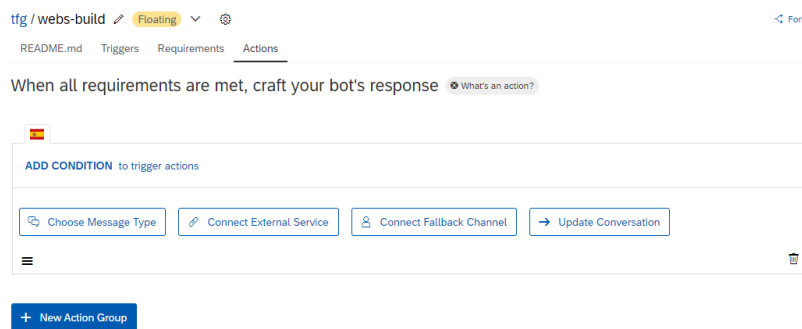


Figura 8

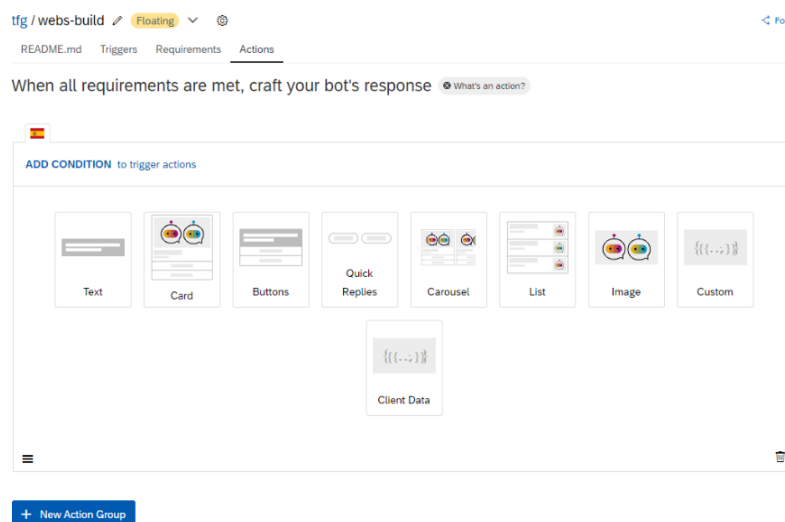


Figura 9

Cuando seleccionamos la opción “Buttons”, nos aparecerá un cuadro para escribir el mensaje que se quiere mostrar por pantalla. Aquí escribiremos la información relacionada a la pregunta hecha por el usuario. En la opción “Add a button”, existen

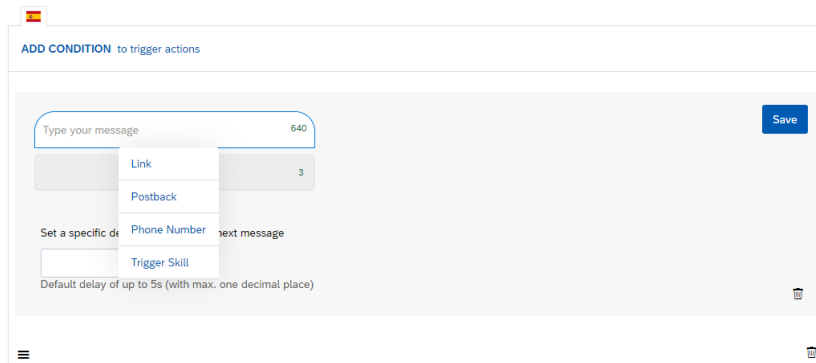


Figura 10

cuatro alternativas para añadir un botón: añadir un enlace a una nueva pestaña, un postback, un número de teléfono, o un enlace directo a otra skill relacionada. En la figura 11 se puede observar una skill rellena con dos botones a enlaces externos, los cuales redirigirán al usuario a las páginas deseadas.

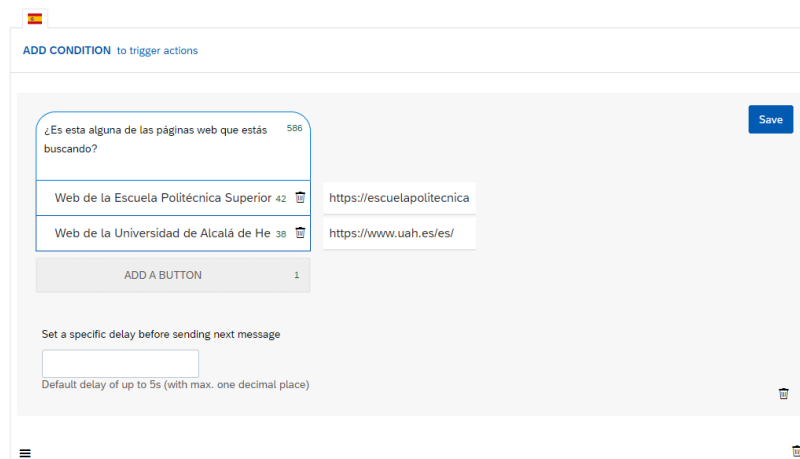


Figura 11

Para este proyecto, se van a utilizar los botones de enlace a una nueva pestaña y de enlace a otra skill relacionada. Un ejemplo de enlace a otra skill relacionada sería el de la figura 12, donde podemos ver una skill de preguntas frecuentes que recoge ciertos apartados que, dependiendo del botón que seleccione el usuario, lo redirigirá a esa skill en concreto. Una vez la navegación entre skills finalice, el usuario encontrará la respuesta a su pregunta o algún enlace para complementar la información ofrecida

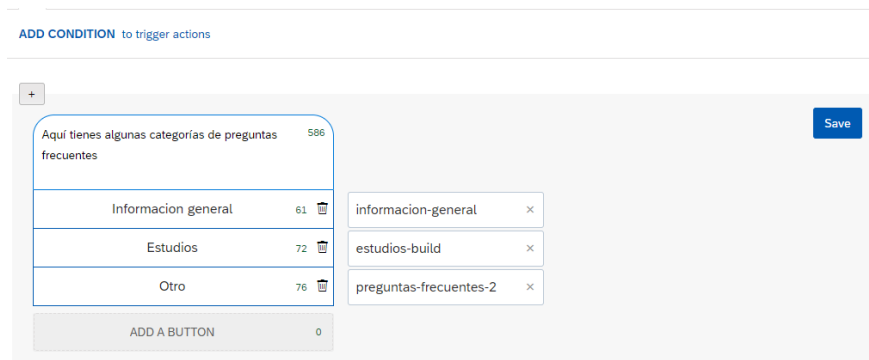


Figura 15

Ahora, hay que probar que el intent de las webs funciona correctamente. Para esto, hay que entrenar al chatbot. Para ello, en la parte superior de la página, encontraremos un botón “Train” (figura 13) con un círculo naranja al lado. El color naranja significa que hay cambios nuevos en el chatbot, y que necesita entrenar. Al pulsar el botón de train, primero veremos una rueda que indica que el entrenamiento está en proceso (figura 14), y finalmente aparecerá un círculo verde al lado, lo cual significa que el chatbot ha sido entrenado y está listo para conversar (figura 15).

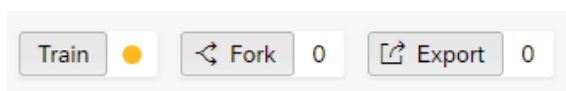


Figura 14

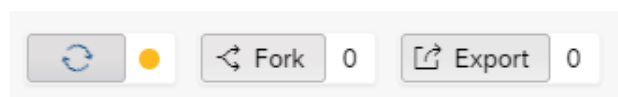


Figura 13

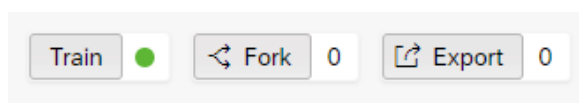


Figura 12

Funcionamiento del chatbot

Cabe recalcar que la interacción con este chatbot tendrá dos variantes:

- 1) **Interacción escrita:** con esto, el chatbot responderá directamente las preguntas que el usuario escriba. Esto está íntegramente ligado con la configuración de los triggers vista en el paso 2 de la configuración de skills.
- 2) **Interacción con botones:** si el usuario no tiene muy claro qué va a preguntar, el chatbot tendrá una configuración de botones mediante la cual el usuario podrá navegar entre los temas seleccionados sin necesidad de escribir nada, siguiendo el sistema redirigir a un enlace de otra skill. Una vez llegue a la opción que considere, podrá consultar la información de igual forma que si escribiera directamente una frase. Para poder iniciar esta interacción con botones, el usuario deberá saludar al chatbot primero.

Para comprobar que la programación de los intents y de las skills funciona correctamente, la web de SAP Conversational AI proporciona una vista previa de cómo quedaría el chatbot, así como un analizador de expresiones para comprobar que la respuesta del chatbot corresponde con lo establecido a la entrada de texto que le pueda dar el usuario.

La vista previa se muestra como un chat normal, donde el usuario procederá a escribir y el chatbot le responderá con la skill programada (figura 16)

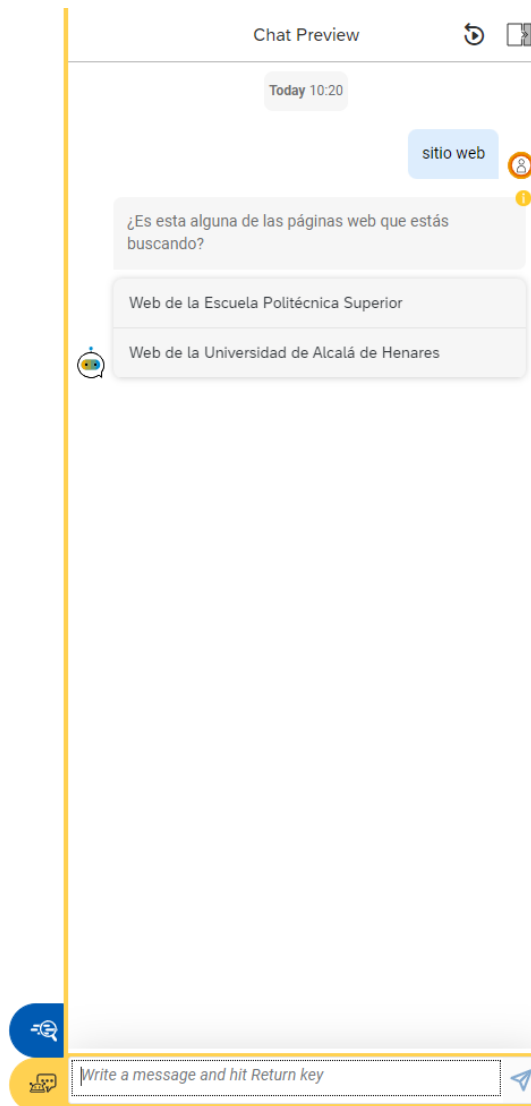


Figura 17

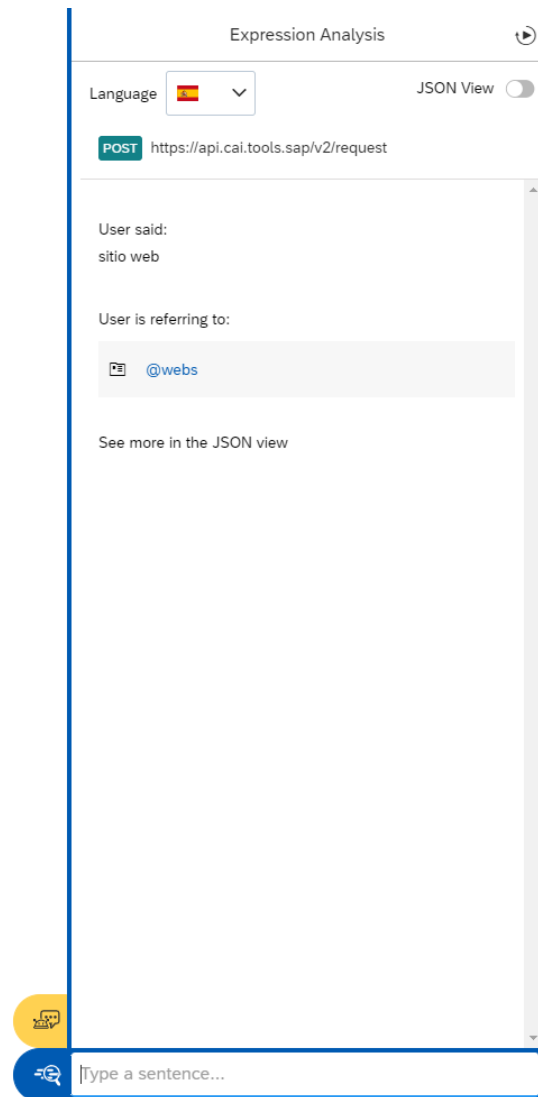


Figura 16

En el analizador de expresiones (figura 17), se introducirá la misma expresión que se ha puesto en la vista previa del chat. Como se puede ver en la imagen, al introducir la misma frase, la configuración del chatbot devuelve que el usuario se está refiriendo al intent concreto que acabamos de programar.

La pregunta ahora es, ¿qué casos se van a recoger para entrenar al chatbot? En la siguiente lista se van a enumerar las situaciones para las cuales se entrenará el chatbot

Lista apartados del Chatbot

- Preguntas frecuentes
 - Información general
 - Webs
 - Qué ofrece
 - Ramas conocimiento
 - Normativas
 - Coordinadores
 - Estudios
 - Qué grados hay
 - Qué masters hay
 - Doctorados
 - Otra educación
 - Estudiantes novatos
 - Primeros pasos
 - TUI
 - Correo electrónico
 - Cuenta de usuario
 - Mi portal
 - Cursos cero
 - Cursos cero informática/químicas/industriales
 - Cursos cero informática y ADE
 - Cursos cero sistemas de la información
 - Panorama universitario
 - Departamentos
 - Asignaturas grado y master
 - Horarios
 - Calendario de actividades
 - Estudiantes veteranos
 - Fechas exámenes
 - Prácticas – enlaza con Prácticas en trámites administrativos
 - Erasmus
 - TFG/TFM
 - Becas
 - Delegación
 - Grupos investigación
 - Servicios
 - cafetería
 - Biblioteca
 - Reprografía
- Trámites administrativos
 - Matrícula
 - Estudiantes nuevos/veteranos
 - Calendario académico
 - Fechas exámenes
 - Solicitudes
 - Solicitudes dirección

- Solicitudes pdi/pas/delegaciones
 - Solicitudes e impresos
- Becas y ayudas
- Pruebas de acceso
 - Evau
 - > 25
 - > 40
 - > 45
- Prácticas
- Cambios de grupo
 - Cambio de grupo grande
 - Solicitud de compensación
 - Solicitud genérica al director
 - Solicitud evaluación final
- Convalidaciones
 - Reconocimiento créditos
 - Acreditación nivel idioma
 - Traslado/reapertura expediente
 - Solicitud de certificado y títulos
 - Otras solicitudes
- Wifi y VPN
 - Red privada virtual
 - Correo electrónico
 - Cuentas y contraseñas de usuario
 - Acceso wifi
- Localizaciones
 - Dónde está
 - Cómo moverme
 - Departamentos
 - Delegación
 - Laboratorios

Para poder comenzar la conversación, la web de SAP Conversational AI proporciona una herramienta que permite poner un mensaje de bienvenida en el chatbot a la hora de enlazarlo a una página web, tal y como se puede observar en la figura 18. Este mensaje solo se podrá ver en la página web donde se va a alojar, no en la vista previa del chatbot.

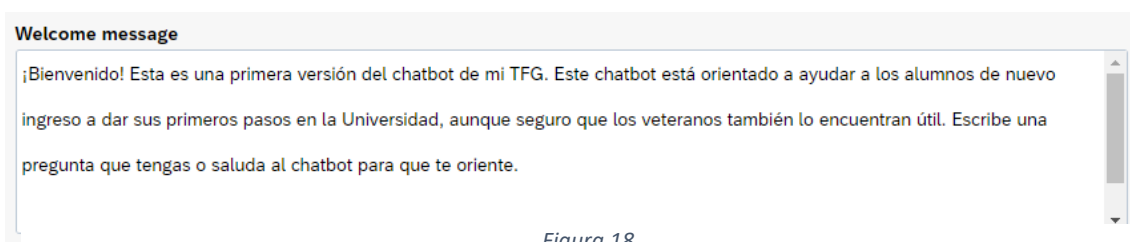


Figura 18

A pesar de esta característica, el usuario siempre va a tener que empezar la conversación con el chatbot. Podrá elegir si preguntar cualquier cosa relacionada con la lista anterior, lo cual iniciará la conversación directamente. Pero si se quiere seguir al dinámica de botones, el usuario tendrá que saludar al chatbot con cualquier mensaje de bienvenida (hola, que tal, quien eres...).

Si el usuario decide seguir la dinámica de botones y saludar al chatbot, este le responderá con un segundo mensaje de bienvenida, donde se enlazarará con los diversos botones a todas las áreas mencionadas antes.

Dado que el chatbot en formato web solo puede soportar 3 botones por cada interacción, se tiene que dividir la respuesta en varias secciones. En las figuras 19 se puede ver el funcionamiento de esta dinámica mediante la vista previa que proporciona el chatbot.



Figura 19

Dentro de cada apartado del índice (de cada botón), el chatbot nos redirigirá a los subíndices que cada apartado tenga asignados. De esta forma, si se quiere llegar a la sección "Fechas de exámenes", el usuario tendrá que pulsar en preguntas frecuentes, buscar el apartado de estudiantes veteranos y posteriormente el apartado de fechas de exámenes.

De nuevo, el árbol de apartados se puede encontrar en la lista de apartados del chatbot.

FASE 2

Creación web y enlace con el Chatbot

Para esta fase, se necesitará tener Java 17 y Maven 3.9.2 instalado en el ordenador. El proyecto se va a crear en el IDE IntelliJ en su versión Community.

Procedemos con la creación del proyecto Maven con el jdk de Java 17.

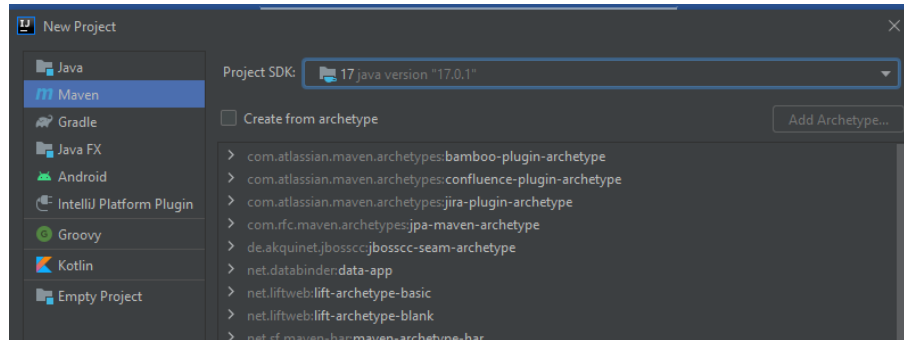
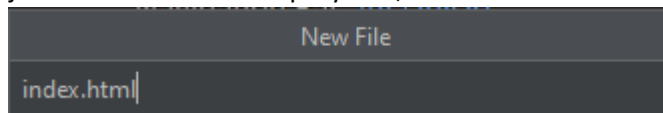


Figura 20

La web se va a programar con html. Para ello, el primer paso es crear en la carpeta principal de java una vez iniciado el proyecto, un nuevo archivo llamado "index.html". En este fichero se



programará la interfaz de la página web y se realizará la configuración del chatbot.

Figura 21

Para proceder con dicha configuración, y de regreso en la web de SAP Conversational AI donde se ha programado el chatbot de la fase 1, nos dirigimos al apartado "Connect", que se puede observar en la figura 22.

Para una primera conexión, se va a dejar el diseño por defecto que nos proporciona la herramienta, únicamente cambiando el nombre del chatbot, el mensaje de bienvenida y los mensajes emergentes que saldrán del bot.

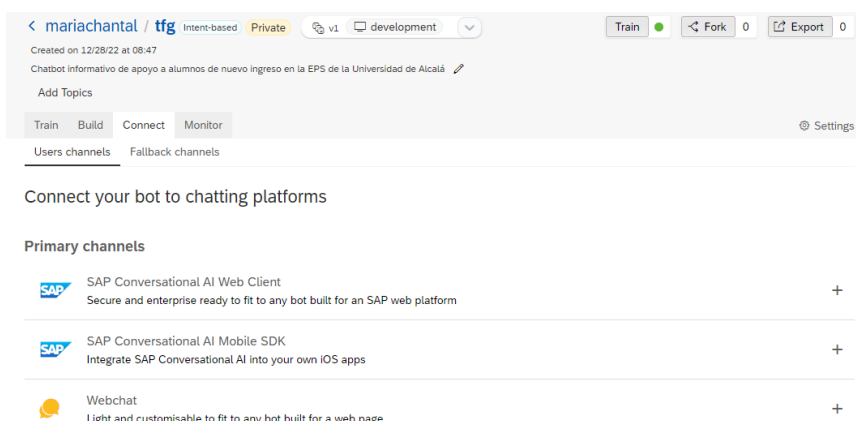
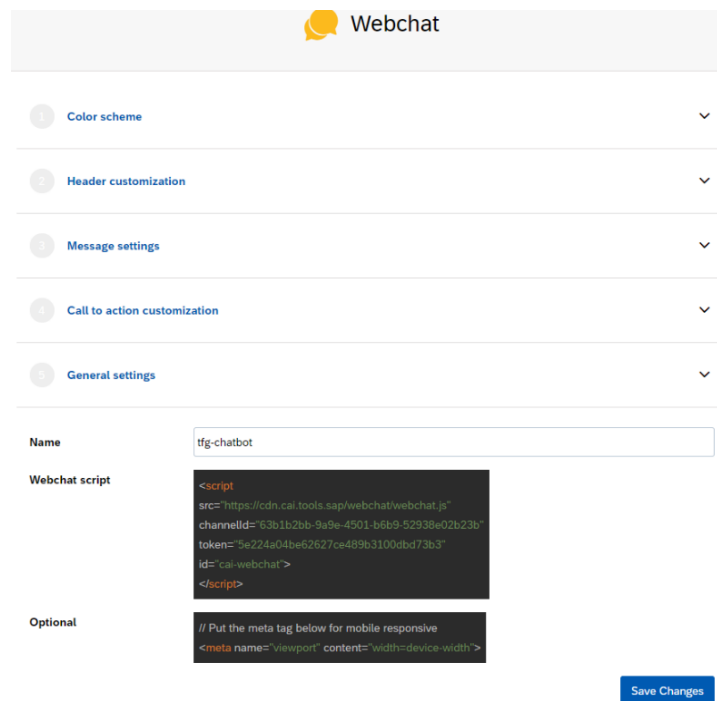


Figura 22

Esta configuración comenzará en la sección “WebChat”. En la figura 23 se puede observar esta interacción.



The screenshot shows the 'Webchat' configuration page. At the top, there's a header with a yellow speech bubble icon and the text 'Webchat'. Below this is a list of settings categories: 1. Color scheme, 2. Header customization, 3. Message settings, 4. Call to action customization, and 5. General settings. The 'General settings' section is expanded, showing fields for 'Name' (tfg-chatbot), 'Webchat script' (a JavaScript code block), and 'Optional' (a meta tag for mobile responsiveness). A 'Save Changes' button is at the bottom right.

```
<script>
src="https://cdn.cai.tools.sap/webchat/webchat.js"
channelId="63b1b2bb-9a9e-4501-b6b9-52938e02b23b"
token="5e224a04be62627ce489b3100dbd73b3"
id="cai-webchat">
</script>
```

```
// Put the meta tag below for mobile responsive
<meta name="viewport" content="width=device-width">
```

Figura 23

Aquí se nos ofrecen diversas opciones de personalización, desde los colores hasta las imágenes de perfil tanto del chatbot como del usuario. De la sección “Message settings”, simplemente se va a modificar, como se ha mencionado antes, el mensaje de bienvenida genérico (figura 24). Más adelante se volverá a este apartado para perfeccionar el diseño del chatbot.



The screenshot shows the 'Message settings' section. It includes fields for 'Bot picture' (https://cdn.cai.tools.sap/webchat/illustration-robot-2.svg), 'User picture' (https://cdn.cai.tools.sap/webchat/illustration-user.svg), 'Onboarding message' (¡El chatbot está aquí abajo!), 'Input placeholder' (Escribe aquí), and 'Welcome message' (¡Bienvenido! Esta es una primera versión del chatbot de mi TFG. Este chatbot está orientado a ayudar a los alumnos de nuevo ingreso a dar sus primeros pasos en la Universidad, aunque seguro que los veteranos también lo encuentran útil. Escribe una pregunta que tengas o saluda al chatbot para que te oriente.).

Figura 24

De nuevo viendo la figura 23, se puede observar el scrip que será necesario introducir en el fichero html previamente declarado. En la figura 25 se observa una programación básica de lo que va a mostrar la página web, con el título, un pequeño encabezado y, de las líneas 32 a la 37, el script del chatbot.

```

1  <html>
2  <head>
3    <title> Maria Chantal TFG </title>
4    <style type="text/css">
5      * {
6        padding:0px;
7        margin:0px;
8      }
9      #header {
10       margin:auto;
11       width:500px;
12     }
13     ul, ol {
14       list-style:none;
15     }
16     .nav li a {
17       background-color:rgb(132, 15, 228);
18       color: #fff;
19       text-decoration:none;
20       padding: 10px 15 px
21     }
22     .nav > li {
23       float:left
24     }
25   </style>
26 </head>
27 <body>
28   <h1 align="center"> Web de prueba en blanco el chatbot del TFG </h1>
29   <hr>
30 </div>
31 <script
32   src="https://cdn.cai.tools.sap/webchat/webchat.js"
33   channelId="63b1b2bb-9a9e-4501-b6b9-52938e02b23b"
34   token="5e224a04be62627ce489b3100dbd73b3"
35   id="cai-webchat">
36 </script>
37 <hr>
38 <body>
39 </html>

```

Figura 26

Finalmente, el resultado de la programación inicial de la web con el chatbot incorporado sería el de las figuras 26 y 27.

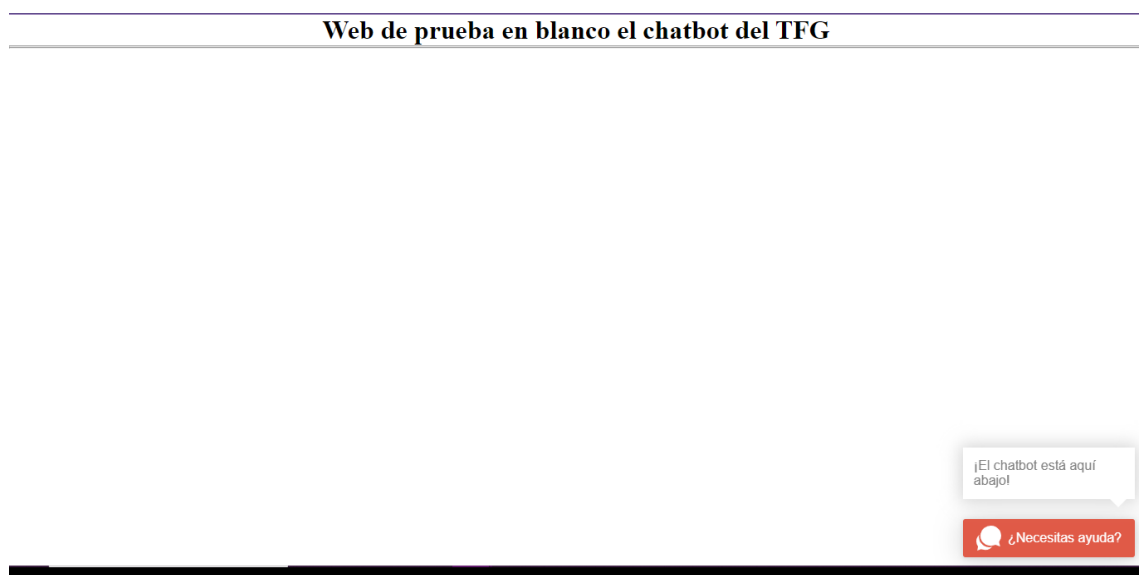


Figura 25

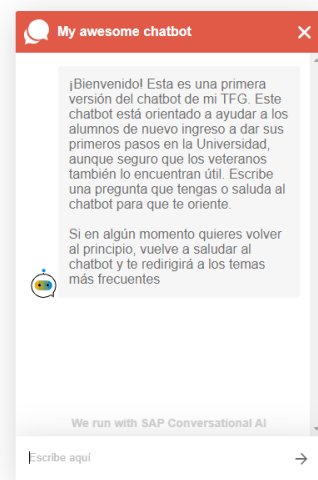


Figura 27

Para comprobar el funcionamiento del chatbot, en la figura 28 se va a iniciar una conversación con un saludo para comprobar la dinámica que se seguiría para una conversación con botones, y en la figura 29 se le va a lanzar la misma pregunta que en la fase 1.

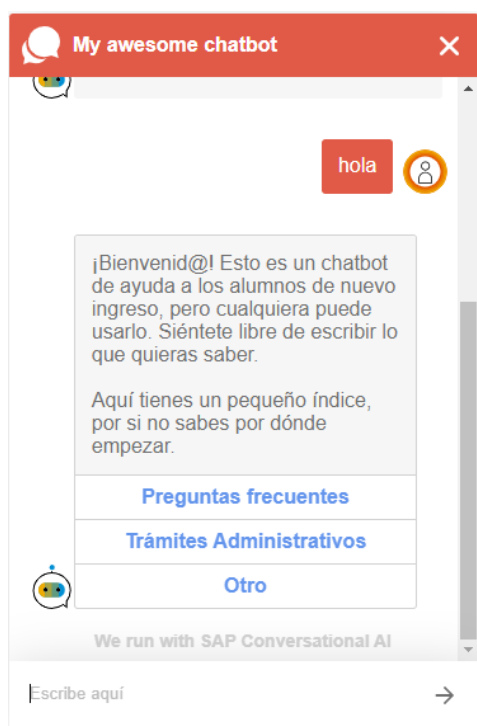


Figura 29



Figura 28

FASE 3

Configuración herramientas de pruebas

Variables de entorno

Para comenzar con la configuración de todas las herramientas necesarias para poder trabajar con Selenium Web Driver y con Cucumber, el primer paso es configurar las variables de entorno de Java 17 y de Maven 3.9.2

Podemos descargar Java 17 en el [siguiente enlace](#). El siguiente paso será acceder a las variables de entorno desde el panel de control.

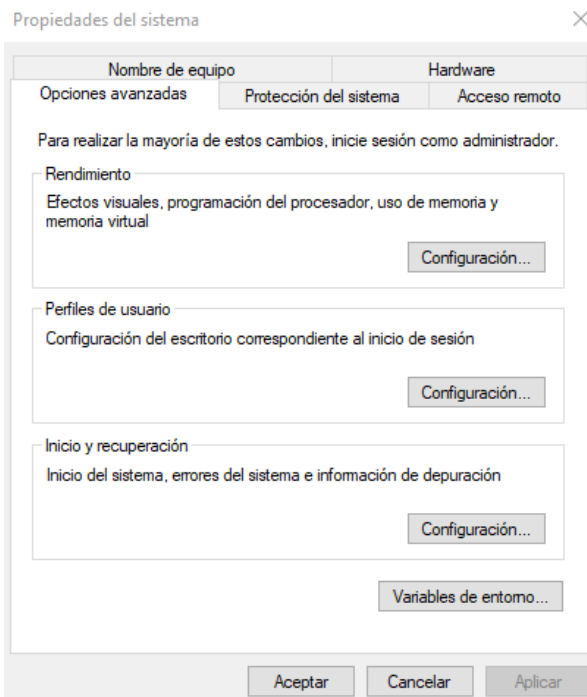


Figura 30

Para comprobar que esta instalación se ha hecho de forma correcta, desde la terminal de Windows, al escribir el comando “java -version”, se puede verificar la versión de java actual instalada en el sistema.

```
Microsoft Windows [Versión 10.0.19045.3086]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Usuario>java -version
java version "17.0.1" 2021-10-19 LTS
Java(TM) SE Runtime Environment (build 17.0.1+12-LTS-39)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.1+12-LTS-39, mixed mode, sharing)

C:\Users\Usuario>
```

Figura 33

Al hacer click en variables de entorno, aparecerán las variables de usuario y de sistema. Para configurar Java 17, será necesario crear una nueva variable de usuario.

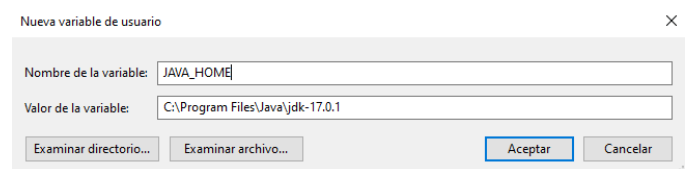


Figura 31

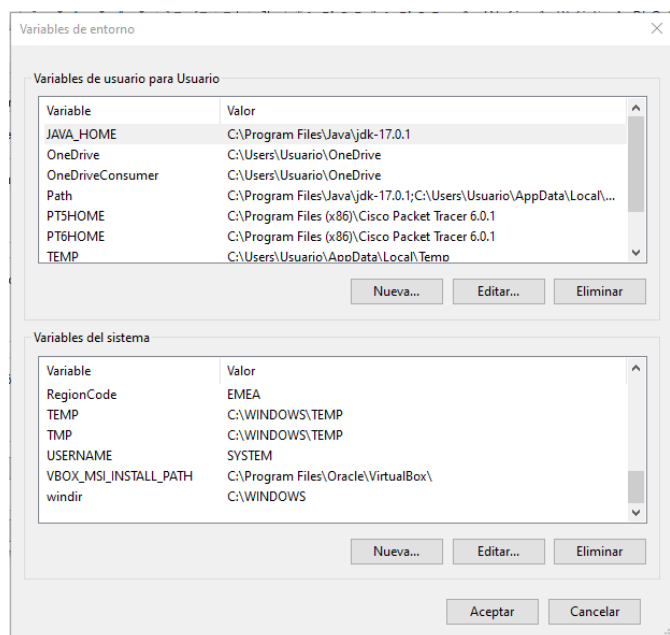


Figura 32

Para la descarga de Maven, hay que acceder al [siguiente enlace](#).

La web nos mostrará distintas opciones de descarga. Para este proyecto enfocado en Windows, habrá que descargarse el archivo “apache-maven-3.9.2-bin.zip”

Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the [installation instructions](#). Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the signature](#) of the release bundles against the public [KEYS](#) used by the Apache Maven developers.

	Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.9.2-bin.tar.gz	apache-maven-3.9.2-bin.tar.gz.sha512	apache-maven-3.9.2-bin.tar.gz.asc
Binary zip archive	apache-maven-3.9.2-bin.zip	apache-maven-3.9.2-bin.zip.sha512	apache-maven-3.9.2-bin.zip.asc
Source tar.gz archive	apache-maven-3.9.2-src.tar.gz	apache-maven-3.9.2-src.tar.gz.sha512	apache-maven-3.9.2-src.tar.gz.asc
Source zip archive	apache-maven-3.9.2-src.zip	apache-maven-3.9.2-src.zip.sha512	apache-maven-3.9.2-src.zip.asc

Figura 35

Se descomprime el archivo en el disco C del ordenador y se añade la ruta a las variables del sistema, al igual que se hizo con Java.

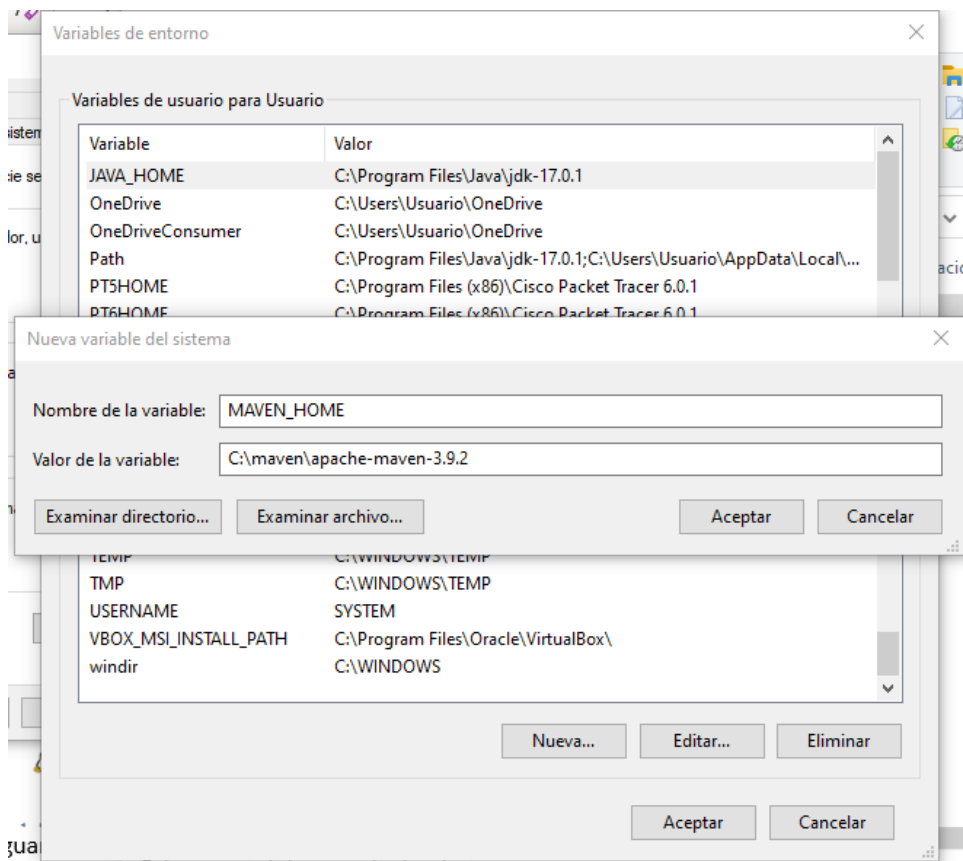


Figura 34

Y de nuevo, se comprueba que la instalación ha sido exitosa con el comando “mvn -v” en la terminal.

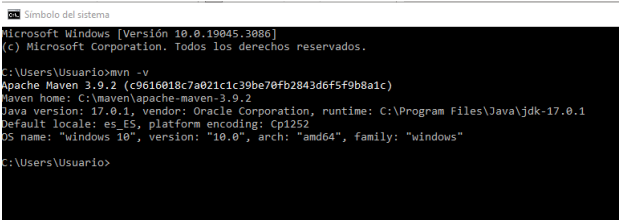


Figura 36

Dependencias de Selenium Web Driver y de Cucumber

Selenium Web Driver y Cucumber no tiene que ser instaladas directamente en el sistema. Al contrario que Java y Maven, estas dos herramientas se instalarán directamente en el proyecto de IntelliJ que se ha creado anteriormente mediante dependencias.

Las dependencias de [Selenium](#) y de [Cucumber](#) se instalarán en la clase “pom.xml”, y es tan sencillo como se muestra en la figura 37.

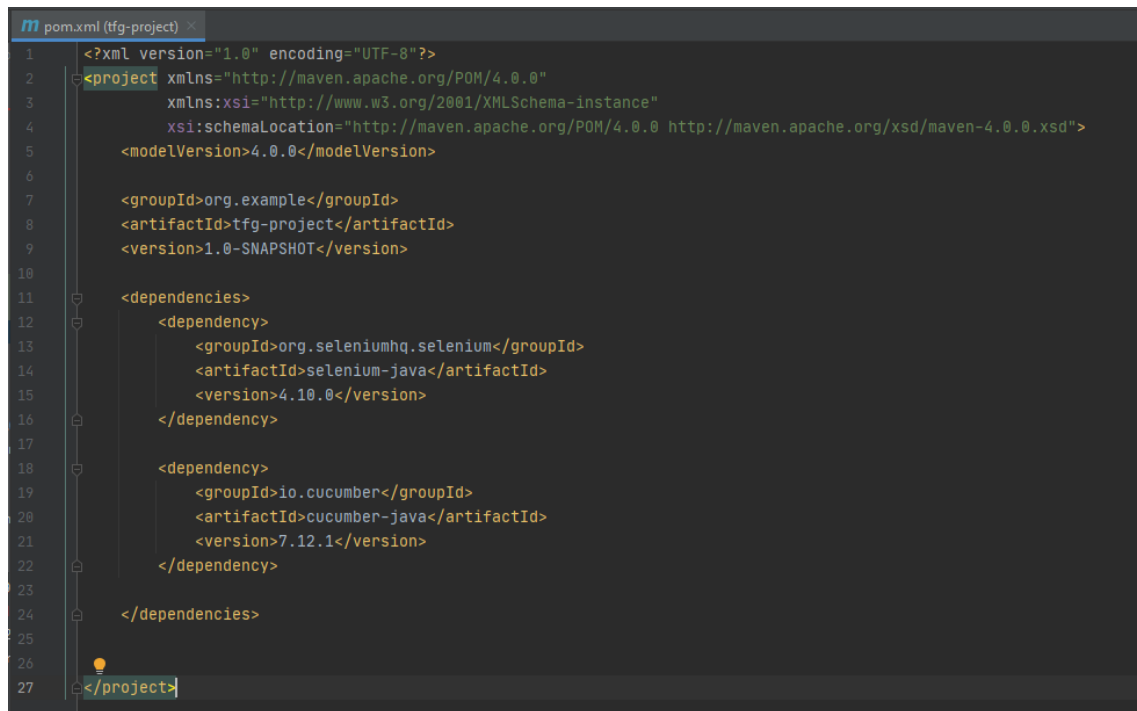


Figura 37

Cucumber tiene una interfaz un tanto característica, por lo que para facilitar la interacción del usuario con la herramienta, es recomendable instalar los plugins de *Gherkin* y de *Cucumber for Java* en IntelliJ.

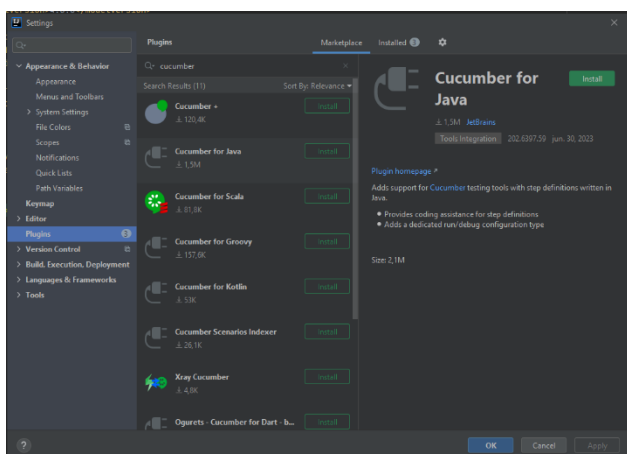


Figura 39

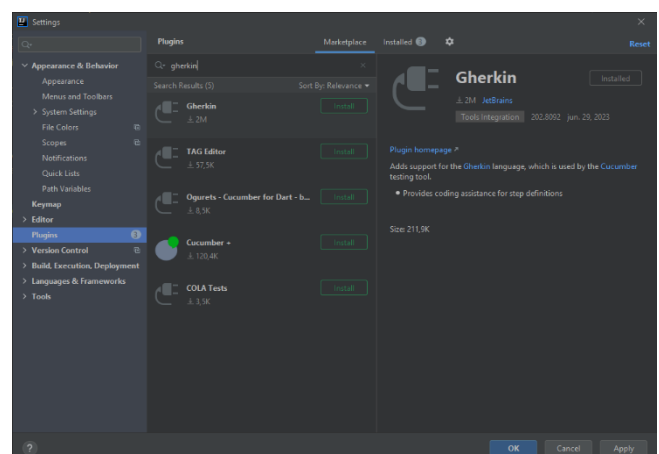


Figura 38

Gherkin es el lenguaje que se va a usar para poder comprender Cucumber. Con él, se podrán definir los escenarios y las funcionalidades de forma muy comprensible para el usuario.

Cucumber es una de las herramientas que se pueden usar para automatizar las pruebas.

Creación de las pruebas automatizadas

Antes de comenzar, es necesario aclarar qué son las pruebas automatizadas y cómo vamos a trabajar con ellas.

A pesar de que su nombre pueda hacer pensar que todo es un proceso automático, estas pruebas necesitan de la interacción humana para garantizar su funcionamiento. El programador deberá saber cuál es el resultado deseado de la web, aplicación o, en este caso, del chatbot. Si el programador sabe que al hacer click en cierto botón se debe redirigir al usuario a cierta web, esto se va a plasmar en las pruebas. Para ello, va a ser necesario acceder a las herramientas de desarrollador de la web o de la aplicación en la que estemos trabajando, el proceso se explicará más adelante.

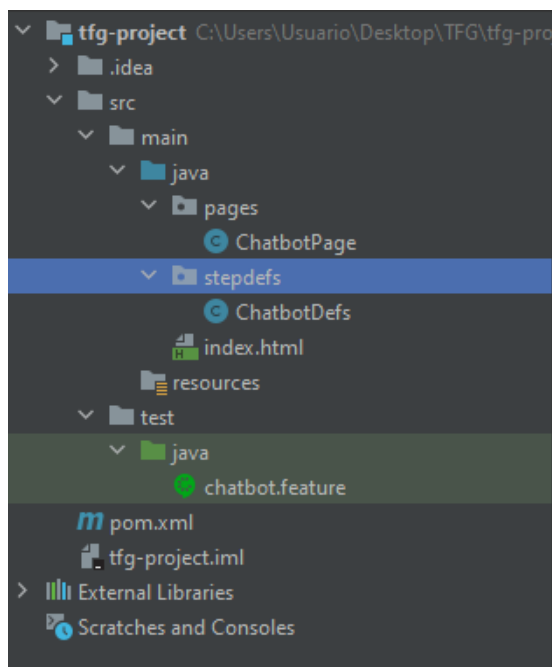


Figura 40

Una vez hecha la configuración de las herramientas, se deben crear tres clases para poder programar el funcionamiento de las pruebas, siguiendo la estructura de la figura 40

La clase “ChatbotPage” servirá como clase para poder almacenar los identificadores de los elementos de la web para poder hacer las llamadas en las pruebas automatizadas.

La clase “ChatbotDefs” será en la que se programe la funcionalidad de las pruebas.

Finalmente, la clase chatbot.feature será la clase Cucumber que se encargará de definir los escenarios de cada uno de los casos de prueba que se quieran llevar a cabo. Para ver un ejemplo práctico de cómo funcionan los escenarios en Cucumber, podemos observar la figura 41.

```
Feature: Maria Chantal TFG

Scenario: First greetings
  Given I click on the bot
  When I write 'hola'
  Then the bot gives me a welcome greeting
```

Figura 41

Una clase en Cucumber siempre va a llevar la extensión .feature como distintivo. Estas clases funcionan mediante la configuración de distintos escenarios. Cada escenario es un conjunto de instrucciones que seguirán cierta lógica para poder ejecutar las pruebas. Cada escenario, va a tener las palabras reservadas Given, When, Then y And. Cada una de ellas se utilizará para saber

en qué orden se va a ejecutar cada paso del escenario. Se puede traducir a español directamente para aclarar su funcionamiento. Con base en la imagen de arriba, las sentencias serían “dado que hago click en el bot, cuando escriba hola, entonces el bot me va a dar un mensaje de bienvenida”. Estas palabras reservadas se pueden intercalar y son únicamente un recurso para que el usuario que programa las pruebas tenga una mejor perspectiva de lo que está haciendo.

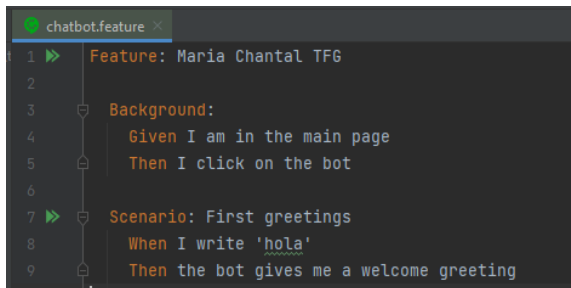


Figura 42

Se pueden programar tantos escenarios como se deseen. Por ello, también una opción llamada “Background”, cuya función es ejecutar las sentencias que estén en su interior siempre antes de cada escenario. Todos los pasos que estén dentro del background, se ejecutarán con todos los escenarios, independientemente de si se lanza uno de forma individual o todos los que agrupe la clase .feature en la que esté declarado.

Estos escenarios por su cuenta no son capaces de ejecutar ningún tipo de prueba. Para completar su funcionalidad, es necesario comenzar con la parte de Selenium.

Selenium Web Driver es la herramienta que nos va a permitir acceder a los elementos de la web que servirán como guía para saber si el funcionamiento del Chatbot es el esperado. Estas pruebas, al ser configuradas de forma manual, tienen el precedente de que el programador debe analizar cuál es el resultado deseado. Aquí es donde entran en juego las herramientas de desarrollador de, en este caso, Google Chrome.

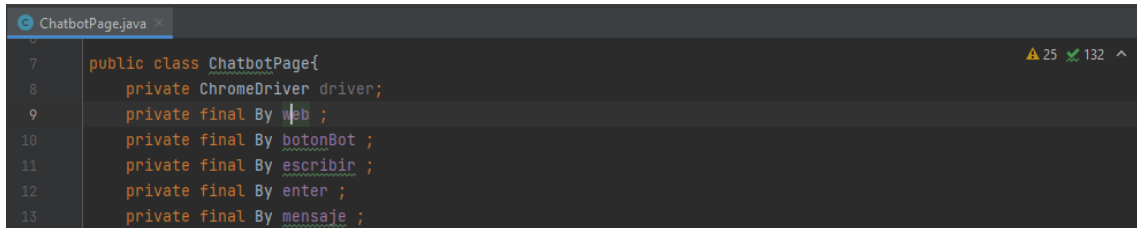


Figura 43

En la web que se ha creado para almacenar el chatbot, haremos click derecho sobre el botón que abre el bot. Después de hacer click en la sección “inspeccionar”, nos aparecerán las herramientas de desarrollador tal y como se aprecia en la figura 43. Se marcará en azul el elemento que hemos seleccionado, y de nuevo con el click derecho, iremos hasta el desplegable “copy” para poder copiar la dirección xpath.

Es importante recalcar que para toda la configuración de pruebas del chatbot, se van a usar identificadores xpath para poder jugar con las secuencias numéricas.

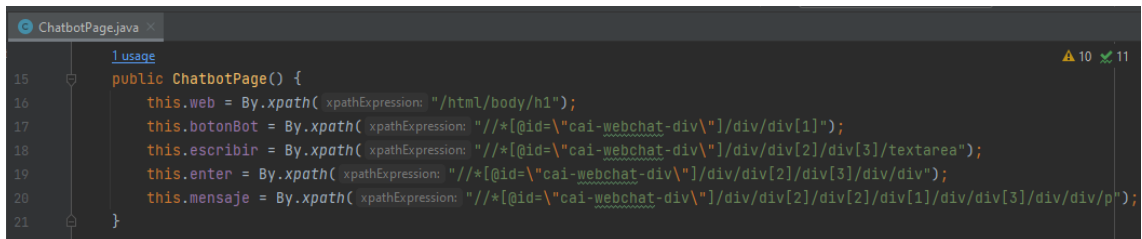
Será necesario hacer esto con todos los elementos que se deseen incluir en las pruebas. En este caso, para el escenario de prueba que se ha visto previamente en la figura 41, será necesario el botón de abrir el chatbot, hacer click en el panel de texto, escribir la pregunta deseada, hacer click en el botón de enviar, y comprobar que el mensaje que recibimos es el indicado. Cada uno de estos pasos tiene su propio elemento asociado. Cogiendo el xpath de cada uno de ellos, se comienza a programar la clase ChatbotPage.



```
7 public class ChatbotPage{
8     private ChromeDriver driver;
9     private final By web ;
10    private final By botonBot ;
11    private final By escribir ;
12    private final By enter ;
13    private final By mensaje ;
```

Figura 44

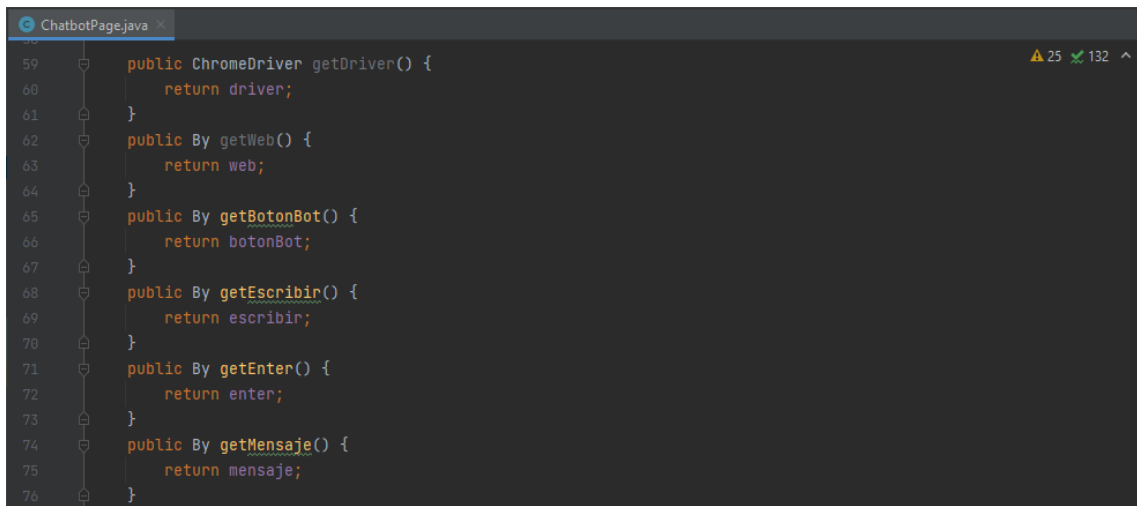
En primer lugar, se declara el nombre de las variables que se van a utilizar. Serán de tipo “By”, dado que Selenium utiliza este tipo de datos para recoger los elementos de identificación de la web. También es necesario declarar el driver del navegador que se vaya a emplear para realizar las pruebas, como se ha mencionado anteriormente, en este caso se usará Google Chrome.



```
15 public ChatbotPage() {
16     this.web = By.xpath( xpathExpression: "/html/body/h1");
17     this.botonBot = By.xpath( xpathExpression: "//*[id=\"cai-webchat-div\"]/div/div[1]");
18     this.escribir = By.xpath( xpathExpression: "//*[id=\"cai-webchat-div\"]/div/div[2]/div[3]/textarea");
19     this.enter = By.xpath( xpathExpression: "//*[id=\"cai-webchat-div\"]/div/div[2]/div[3]/div/div");
20     this.mensaje = By.xpath( xpathExpression: "//*[id=\"cai-webchat-div\"]/div/div[2]/div[2]/div[1]/div/div[3]/div/div/p");
21 }
```

Figura 45

A continuación, dentro del constructor, se van a inicializar estas variables a los valores xpath que se han recogido anteriormente del inspector de elementos de la web.



```
59 public ChromeDriver getDriver() {
60     return driver;
61 }
62 public By getWeb() {
63     return web;
64 }
65 public By getBotonBot() {
66     return botonBot;
67 }
68 public By getEscribir() {
69     return escribir;
70 }
71 public By getEnter() {
72     return enter;
73 }
74 public By getMensaje() {
75     return mensaje;
76 }
```

Figura 46

Finalmente, para poder utilizar estas variables fuera de seta clase, se crearán métodos get para poder recuperar los valores.

Una vez recogidos los identificadores de cada elemento, se pueden programar los escenarios en la clase ChatbotDefs.java.

Para poder definir cada paso de cada escenario previamente declarados en la clase .feature, se tienen que crear métodos que llamen a estos pasos de forma individual. La forma de hacer esto es mediante etiquetas. Las etiquetas de Cucumber permiten especificar el tipo de palabra reservada que se ha empleado para cada paso (Given, When, Then y And). Junto con esta etiqueta, debe ir el nombre exacto que se le haya dado al paso específico dentro, con la estructura: @When("^I say hola to the chatbot\$"). De esta forma, se realizará la llamada al paso con ese nombre.

```
@Given("^I am in the main page$")
public void mainpage() {
    System.setProperty("webdriver.chrome.driver",
        "C:\\Users\\Usuario\\Desktop\\TFG\\tfq-project\\src\\main\\resources\\drivers\\chromedriver.exe");
    this.driver = new ChromeDriver();
    driver.get("http://www.mariachantaltfq.es/");
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(time: 10, TimeUnit.SECONDS);
    this.chp = new ChatbotPage();
}

@Then("^I click on the bot$")
public void clickonbot() throws InterruptedException {
    driver.manage().timeouts().implicitlyWait(time: 2, TimeUnit.SECONDS);
    driver.findElement(chp.getBotonBot()).click();
}

@When("^I say hola to the chatbot$")
public void iSayHi(){
    driver.findElement(chp.getEscribir()).click();
    driver.findElement(chp.getEscribir()).sendKeys(...keysToSend: "Hola");
    driver.findElement(chp.getEnter()).click();
}

@Then("^the bot gives me a welcome greeting$")
public void welcomeGreeting(){
    driver.findElement(chp.getMensaje()).toString().substring(0, 31)
        .equals(("¡Bienvenid@! Esto es un chatbot"));
}
```

Figura 47

Estos son los cuatro pasos que forman el “background” y el escenario de prueba diseñado para ejemplificar el funcionamiento de Cucumber y Selenium. La estructura general sería crear un método con la etiqueta del nombre del paso correspondiente.

En el primer método mainpage(), se puede observar la construcción del driver que va a ejecutar las pruebas. Se le debe pasar la ruta donde está guardado el driver del navegador deseado, e inicializarlo. Se le dice al driver que se ponga a pantalla completa y se le asigna una espera implícita para garantizar que la página ha cargado de forma satisfactoria.

En el siguiente método, se hará click en el botón del chatbot para abrirlo. Llamando al driver, se hará click en el botón declarado mediante el xpath en la clase ChatbotPage. Este método y el anterior se ejecutarán siempre, dado que han sido configurados como background.

El siguiente “When” ya es específico del escenario que estamos construyendo. En primer lugar, será necesario hacer click en el recuadro de escritura del chatbot, para posteriormente escribir el mensaje deseado y luego enviarlo, pulsando nuevamente un botón.

Para finalizar, comprobaremos que el mensaje que nos devuelve el chatbot es el que hemos configurado previamente. Para ello (y para no meter todo el mensaje), se selecciona una cantidad corta de caracteres y se comprueba que esos caracteres son iguales que lo que está recibiendo el driver como respuesta.

Con esto, se ha realizado un primer escenario básico de una prueba automatizada del chatbot de ayuda a los estudiantes de nuevo ingreso.

Casos de prueba

Como se explicó previamente en la Fase 1, en el apartado de “Funcionamiento del chatbot”, la interacción con el chatbot se dividirá en dos secciones, una interacción escrita y una interacción con botones. Se puede ver en la siguiente lista los casos de prueba seleccionados para cada tipo de interacción. Subrayado en morado, los casos de interacción escrita, y subrayado en amarillo, los casos de interacción con botones.

- Preguntas frecuentes
 - Información general
 - Webs
 - Qué ofrece
 - Ramas conocimiento – enlazado con estudios
 - Normativas
 - Coordinadores
 - Estudios
 - Qué grados hay
 - Qué masters hay
 - Doctorados
 - Otra educación
 - Estudiantes novatos
 - Primeros pasos
 - TUI
 - Correo electrónico – enlazado con Wifi y VPN
 - Cuenta de usuario
 - Mi portal
 - Cursos cero
 - Cursos cero informática/químicas/industriales
 - Cursos cero informática y ADE
 - Cursos cero sistemas de la información
 - Panorama universitario
 - Departamentos
 - Asignaturas grado y master
 - Horarios
 - Calendario de actividades
 - Estudiantes veteranos
 - Fechas exámenes – enlazado con trámites administrativos
 - Prácticas – enlaza con Prácticas en trámites administrativos
 - Erasmus
 - TFG/TFM
 - Becas – enlazado con trámites administrativos
 - Delegación

- Grupos investigación
- Servicios
 - cafetería
 - Biblioteca
 - Reprografía
- Trámites administrativos
 - Matrícula
 - Estudiantes nuevos/veteranos
 - Calendario académico
 - Fechas exámenes
 - Solicitudes
 - Solicitudes dirección
 - Solicitudes pdi/pas/delegaciones
 - Solicitudes e impresos
 - Becas y ayudas
 - Pruebas de acceso
 - Evau
 - > 25
 - > 40
 - > 45
 - Prácticas
- Cambios de grupo
 - Cambio de grupo grande
 - Solicitud de compensación
 - Solicitud genérica al director
 - Solicitud evaluación final
- Convalidaciones
 - Reconocimiento créditos
 - Acreditación nivel idioma
 - Traslado/reapertura expediente
 - Solicitud de certificado y títulos
 - Otras solicitudes
- Wifi y VPN
 - Red privada virtual
 - Correo electrónico
 - Cuentas y contraseñas de usuario
 - Acceso wifi
- Localizaciones
 - Dónde está
 - Cómo moverme
 - Departamentos
 - Delegación
 - Laboratorios

Escenarios para los casos de prueba

Anteriormente se ha explicado el funcionamiento de un escenario básico de Cucumber y su configuración. Para poder optimizar la creación de escenarios para los casos de prueba, Cucumber proporciona otro tipo de escenarios, llamados “Scenarios Outline”. Este tipo de escenarios permitirá hacer iteraciones entre diversos ejemplos que se le proporcionen.

Dejando el “Background” que se ha configurado para el ejemplo, procedemos a crear el “Scenario Outline” para la interacción escrita con el chatbot

```
Scenario Outline: seleccion escrita
  Then I write <seleccion>
  And I receive the <seleccion> response
  Examples:
  |seleccion|
  |que ofrece|
  |masters|
  |doctorados|
  |cursos cero|
  |departamentos|
  |fechas exámenes|
  |erasmus|
  |becas|
  |cafeteria|
  |matricula|
  |cambio de grupo grande|
  |wifi|
  |laboratorios|
```

Figura 48

La principal diferencia que se observa en relación a un escenario simple, es la existencia de ejemplos. Estos ejemplos funcionan como un bucle for: en las sentencias de los pasos a realizar (Then y And en este caso) se puede observar que se está llamando a una variable <selección>. Esta variable irá llamando a cada ejemplo de forma escalonada, de forma que en primer lugar se ejecutará todo el escenario para el ejemplo “que ofrece”, después, para el ejemplo “masters”, y así hasta llegar al ejemplo “laboratorios”.

La programación de estos ejemplos se ve de forma muy intuitiva si se emplea un sistema switch-case a la hora de programar cada paso, pues así se podrá ir iterando entre cada ejemplo de forma muy visual.

Para poder llamar a estas variables desde la programación de los pasos, habrá que hacer referencia a la variable <selección>. Esta llamada se puede realizar escribiendo los caracteres reservados ([^"]*) en el lugar donde iría esta variable en los pasos de Cucumber, tal y como se muestra en la figura 49.


```

@Then("^I write ([^\"]*)$")
public void queOfrece(String sel){
    waitMethod();
    driver.findElement(chp.getEscribir()).click();
    switch (sel) {
        case "que ofrece":
            driver.findElement(chp.getEscribir()).sendKeys( ...keysToSend: "quiero saber que ofrece la eps");
            driver.findElement(chp.getEnter()).click();
            break;
        case "masters":
            driver.findElement(chp.getEscribir()).sendKeys( ...keysToSend: "por favor dime el listado de masters");
            driver.findElement(chp.getEnter()).click();
            break;
        case "doctorados":
            driver.findElement(chp.getEscribir()).sendKeys( ...keysToSend: "quiero cursar un doctorado");
            driver.findElement(chp.getEnter()).click();
            break;
        case "cursos cero":
            driver.findElement(chp.getEscribir()).sendKeys( ...keysToSend: "Hay algun curso de introduccion?");
            driver.findElement(chp.getEnter()).click();
            break;
        case "departamentos":
            driver.findElement(chp.getEscribir()).sendKeys( ...keysToSend: "departamento de ciencias de la computacion");
            driver.findElement(chp.getEnter()).click();
            break;
    }
}

```

Figura 49

Este método hará que el chatbot escriba la frase que se ve reflejada en la figura anterior y que la envíe haciendo click en el botón enter.

En la siguiente figura se puede observar un funcionamiento similar, poniendo los caracteres reservados ([^\"]*) para hacer referencia a la variable selección que se quiere ir sustituyendo. La diferencia con este método es que este será el encargado de recuperar la respuesta del bot y de asegurarse de que la contestación prevista concuerda con lo que se está recibiendo en realidad

Siguiendo con la interacción mediante botones, aquí se podrán encontrar dos “Scenario Outline” y dos escenarios normales.

```

Scenario Outline: tramites administrativos
When I say hola to the chatbot
Then I click on the tramites administrativos button
And I search until I find my destiny for tramites in <seleccion>
Then I go to the <seleccion> web for tramites
Examples:
|seleccion|
|calendario academico|
|solicitudes|
|evau|

Scenario Outline: convalidaciones
When I say hola to the chatbot
Then I click on the convalidaciones button
And I search until I find my destiny for convalidaciones in <seleccion>
Then I go to the <seleccion> web for convalidaciones
Examples:
|seleccion|
|reconocimiento creditos|
|nivel idioma|
|traslado expediente|

```

Figura 50

Se van a dividir por secciones dentro de los casos de prueba. En la figura 50 se puede observar que, como en los casos de prueba seleccionados para interacción con botones en la sección de “trámites administrativos” hay tres opciones que se van a llevar a cabo mediante botones, conviene más configurar un Scenario Outline para que la iteración sea más dinámica. Lo mismo ocurre con la sección de convalidaciones.

Esta dinámica es muy útil sobre todo para el apartado de preguntas frecuentes, el cual cuenta con numerosas interacciones con botones programadas.

```

Scenario Outline: preguntas frecuentes
  When I say hola to the chatbot
  Then I click on the preguntas frecuentes button
  And I search until I find my destiny in <seleccion>
  Then I go to the <seleccion> web
Examples:
  |seleccion|
  |informacion general|
  |normativa|
  |correo electronico|
  |cuenta usuario|
  |panorama universitario|
  |asignaturas|
  |horario|
  |practicas|
  |tfg|
  |biblioteca|

```

Figura 51

```

Scenario: vpn
  When I say hola to the chatbot
  Then I click on the VPN button
  And I search until I find my destiny for vpn
  Then I go to the web for vpn

Scenario: localizaciones
  When I say hola to the chatbot
  Then I click on the como moverme button
  And I search until I find my destiny for como moverme
  Then I receive the como moverme response message

```

Por otro lado, para las secciones de VPN y de Localizaciones, únicamente se ha seleccionado un caso de prueba, por lo que configurar un escenario simple es la opción más conveniente en estos casos. La dinámica de programación será igual que como se ha visto en el escenario de ejemplo de explicación.

Figura 52

La interacción con botones tiene una característica peculiar que no tenía la interacción escrita. Como es posible que el usuario tenga que pulsar varios botones para llegar hasta la consulta deseada, los enlaces de los xpath van a variar, de forma que dependiendo del número de mensaje, de la cantidad de botones y de la cantidad de links, estos identificadores van a cambiar.

Como crear una variable para cada uno de estos casos sería poco óptimo para el código, se van a seleccionar los distintos casos que se pueden dar y se irán modificando mediante variables de método.

```
public WebElement getMessage(int i) {
    WebElement message = driver.findElement(By.xpath("//*[@id=\"cai-webchat-div\"]/div/div[2]/div[2]/div[1]/div/div[\"+i+\"]/div/div"));
    return message;
}

public WebElement getSingleMessage(int i) {
    WebElement message = driver.findElement(By.xpath("//*[@id=\"cai-webchat-div\"]/div/div[2]/div[2]/div[1]/div/div[\"+i+\"]/div/div/div/div"));
    return message;
}

public WebElement getButton(int i, int j){
    WebElement button = driver.findElement(By.xpath("//*[@id=\"cai-webchat-div\"]/div/div[2]/div[2]/div[1]/div/div[\"+i+\"]/div/div/div/div[\"+j+\"]"));
    return button;
}

public WebElement getLinkMessage(int i, int j) {
    WebElement message = driver.findElement(By.xpath("//*[@id=\"cai-webchat-div\"]/div/div[2]/div[2]/div[1]/div/div[\"+i+\"]/div/div/div/a[\"+j+\"]"));
    return message;
}

public WebElement getSingleLinkMessage(int i) {
    WebElement message = driver.findElement(By.xpath("//*[@id=\"cai-webchat-div\"]/div/div[2]/div[2]/div[1]/div/div[\"+i+\"]/div/div/div/a"));
    return message;
}
```

Figura 53

- `getMessage()`: este método recuperará el identificador xpath para un mensaje. Es decir, este será el mensaje que o bien escribe el usuario o bien el que responde el chatbot, sin meterse en los botones que puede llevar asociados esa respuesta. La variable `i` se referirá al número de mensaje en el que se encuentra este identificador.
- `getSingleMessage()`: este método se usará para hacer click en un botón de respuesta. La característica de este tipo de botón es que es específico para el único botón que no redirija a una web externa. Por ejemplo, si hay dos respuestas que redirigen al usuario a una web externa y uno que redirige a otra skill para seguir navegando por botones, se usará este método para esta última opción. La variable `i` se referirá al número de mensaje en el que se encuentra este identificador.
- `getButton()`: este método se encarga de pulsar cualquier botón que redirija a otra skill. La variable `i` se referirá al número de mensaje en el que se encuentra este identificador, mientras que la variable `j` se referirá al número de botón que se quiere pulsar.
- `getLinkMessage()`: este método se encargará de redirigir al usuario a una web externa. Esta redirección estará siempre dentro de uno de los botones disponibles. La variable `i` se referirá al número de mensaje en el que se encuentra este identificador, mientras que la variable `j` se referirá al número de botón que se quiere pulsar.
- `getSingleLinkMessage()`: este método, al igual que `getSingleMessage()`, tiene la peculiaridad de que se usará en el caso de que el botón a pulsar sea el único que tiene un link a una web externa. Por ejemplo, si hay dos botones que redirigen a otra skill, y un botón que te lleva a una web externa, para este último caso se usará este método. La variable `i` se referirá al número de mensaje en el que se encuentra este identificador.

Cambios de diseño

Cambios de diseño en el chatbot

Para personalizar un poco el chatbot que se ve en la web, SAP Conversational AI ofrece la posibilidad de editar la apariencia del bot. Para ello, hay que dirigirse al apartado “Connect” y posteriormente hacer click sobre “Webchat”. Aquí estará almacenada la configuración inicial que se le dio al chatbot

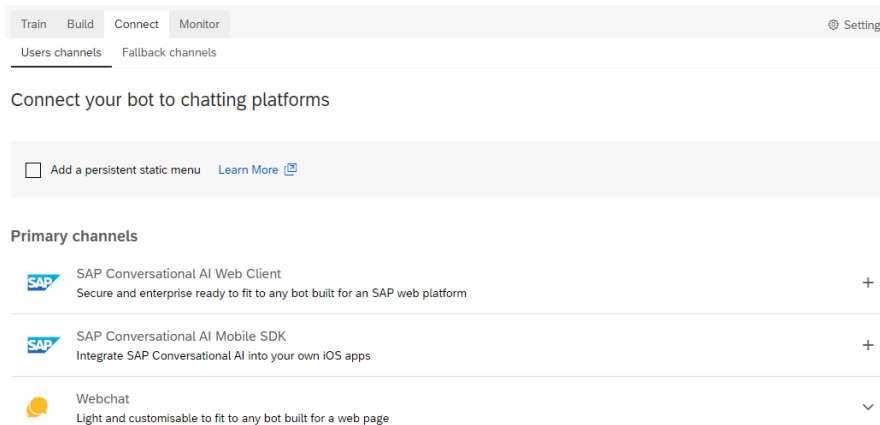


Figura 54

Aquí se abrirá un desplegable. En la sección “color scheme” se podrán elegir los colores que se mostrarán en el chatbot

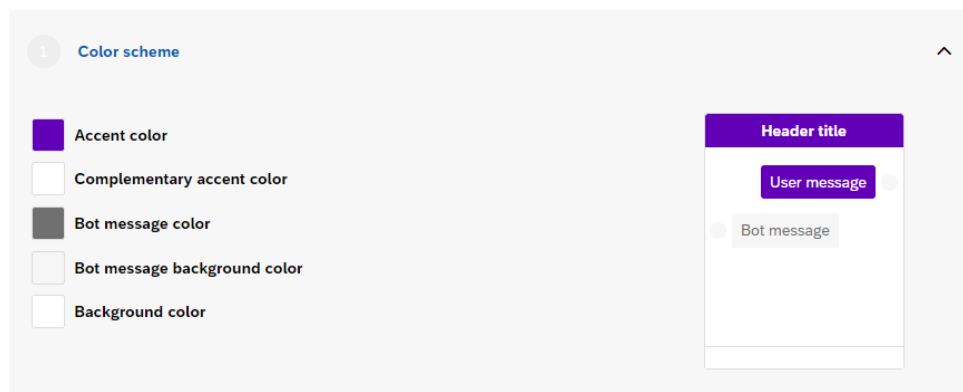


Figura 55

Para la cabecera, simplemente se le pondrá el nombre de “Chatbot TFG” como algo identificativo”



Figura 56

También se pueden personalizar las imágenes de icono que se ven en el bot. Para este caso, se va a elegir un icono general para el usuario y el icono de la Universidad de Alcalá para mostrar como icono del bot.



3 Message settings

Bot picture

<https://lcelula.es/wp-content/uploads/2017/12/logo-uah.jpg>

User picture

<https://cdn-icons-png.flaticon.com/512/1077/1077114.png>

Onboarding message

¡El chatbot está aquí abajo!

Input placeholder

Escribe aquí 23

Welcome message

¡Bienvenido! Esta es una primera versión del chatbot de mi TFG. Este chatbot está orientado a ayudar a los alumnos de nuevo ingreso a dar sus primeros pasos en la Universidad, aunque seguro que los veteranos también lo encuentran útil. Escribe una pregunta que tengas o saluda al chatbot para que te oriente.

Figura 57

El resultado final sería el mostrado en la figura 58.

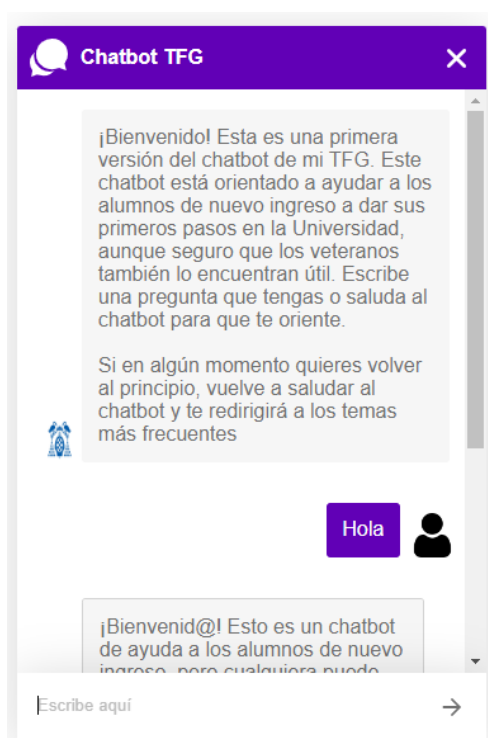


Figura 58

Cambios de diseño en la web

Después de haber trabajado con un diseño muy básico en la web hasta ahora, es momento de retocarla para que sea más visible y tenga más funcionalidades. Vamos a introducir aquí, a parte de HTML, el lenguaje CSS para dar estilo a la web.

Partiendo de la clase ya existente index.html, en la cabecera se le tendrán que insertar las hojas de estilo. Esto se compone desde la clase “main.css”, que será la clase donde definamos las clases que definirán el diseño y la distribución de la página, hasta el tipo de letra que se va a emplear.

```
<head>
  <meta charset="UTF-8">
  <title>Maria Chantal TFG</title>

  <link rel="stylesheet" href="main.css" type="text/css">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Montserrat">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKrQN+PtmoHDEXuppvnDJzQIu9" crossorigin="anonymous">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
    crossorigin="anonymous"></script>
</head>
```

Figura 59

La forma de trabajar con .css es mediante clases. Las clases se añaden en el .html a través de los sectores de división (<div>), como se puede ver en la figura 60, que correspondería al cuerpo de la clase index.html. Los <div> son divisiones en “cajas” que permitirán dividir la página en secciones para poder tener cierto control sobre ellas. También haremos divisiones por párrafos (<p>).

```
<body>

  <div class="pagina">
    <div class="menu">
      <div class="logoMenu">
        <a href="index.html"><i class="bi bi-c-circle"></i>Página principal</a>
      </div>
      <div class="accionesMenu">
        <div class="cajaAccionesMenu"><a href="SAP.html">SAP Conversational AI</a></div>
        <div class="cajaAccionesMenu"><a href="Pruebas.html">Pruebas automatizadas</a></div>
        <div class="cajaAccionesMenu"><a href="Memoria.html">Memoria explicativa</a></div>
      </div>
      <div class="derechaMenu"></div>
    </div>
    <div class="rowCaja">
      <div class="cajaFlexTexto">
        <div class="cajaTexto">
          <H2>Bienvenido a la web de prueba del chatbot</H2>
        </div>
        <div class="cajaTexto">
          <p>Esta es la página desde las que se harán las pruebas automatizadas con Cucumber y Selenium para el chatbot. Puedes encontrar el chatbot en un desplegable abajo a la derecha. Haciendo click sobre él, podrás empezar una conversación. Para más información sobre cómo se ha creado el chatbot, puedes consultar los menús que encontrarás en la parte superior de la página
        </div>
      </div>
    </div>
  </div>
```

Figura 60

Este nuevo diseño de la web va a tener un menú en la zona superior y un botón para regresar a la página principal. En el menú, se podrá acceder a diferentes .html codificados con la información relativa:

- SAP.html: aquí habrá una breve explicación sobre la herramienta que se ha usado para desarrollar el chatbot así como un enlace para acceder.
- Pruebas.html: aquí se podrá encontrar información sobre las pruebas automatizadas que se han usado para el proyecto.
- Memoria.html: aquí aparecerá el .pdf de la memoria de este TFG, para poder acceder a ella de forma dinámica en la web en cualquier momento.

En todas estas páginas, aparecerá en la zona inferior derecha el chatbot en todo momento y la configuración del menú será la misma.

SAP.html

En esta clase, a parte de las divisiones <div> y los párrafos <p> previamente mencionadas, se va a trabajar también con puntos de lista. Estos se van a diferenciar por estar en la división e indicados con .

```
<div class="rowCaja">
  <div class="cajaFlexTexto">
    <div class="cajaTextoTitulo">
      <H2>¿Qué es SAP Conversational AI?</H2>
    </div>
    <div class="cajaTexto">
      <p>
        SAP Conversational AI es una herramienta que ofrece una interfaz para poder desarrollar,
        entrenar,
        probar y conectar chatbots creados por el usuario.
        Permite simplificar información convirtiéndola en una sencilla charla entre el chatbot y el
        usuario.
      </p>
      <p>
        Esta herramienta incluye todas las características necesarias para poder construir tu propio
        chatbot
        desde cero.
      </p>
    </div>
  </div>
```

Figura 61

```
<div class="cajaTextoLista">
  <ul>
    <li>En primer lugar, te permite, de forma muy intuitiva, crear tu propio chatbot. Desde las
    expresiones que el usuario introducirá como preguntas hasta las respuestas que se esperan
    del
    bot.</li>
    <li>Un paso fundamental a la hora de construir este chatbot es su entrenamiento. Gracias al
    entrenamiento, el chatbot podrá asimilar todas las preguntas y respuestas que se le han
    asignado
    para poder mantener una conversación por si mismo</li>
    <li>En este caso hemos conectado el chatbot a una web, pero SAP Conversational AI también
    permite
    conectarlo a canales de mensajería, como Slack o Telegram.</li>
  </ul>
  <p>En la esquina inferior derecha, podrás observar el desplegable del chatbot. Haciendo click
  sobre
  él, podrás iniciar una conversación. Es muy importante recordar que, para activarlo, es muy
  importante saludarle o hacerle directamente una pregunta.</p>
</div>
<div class="cajaBoton">
  <button type="button" class="btn btn-outline-primary btn-lg" onclick="">Visita la página de SAP
```

Figura 62

Pruebas.html

Aquí se van a encontrar muchas divisiones de párrafos ya que es una clase puramente explicativa. Habrá dos subtítulos para explicar Cucumber y Selenium, y un botón con enlace a cada una de las webs oficiales.

```
<div class="contenido">
  <div class="cajaFlexTexto">
    <div class="cajaTextoTitulo">
      <H2>¿Qué son las pruebas automatizadas?</H2>
    </div>
    <div class="cajaTexto">
      <p>
        Las pruebas automatizadas consisten en emplear una o varias herramientas para probar algún tipo de software.
      </p>
      <p>
        El objetivo de las pruebas automatizadas es comprobar que la ejecución del software está funcionando de manera esperada.
        El programador debe tener conocimiento de cómo debería funcionar su software, por lo que deberá plasmar en las pruebas automatizadas las órdenes necesarias para garantizar que el código funciona correctamente
      </p>
      <p>
        Existen numerosas herramientas para realizar pruebas de este estilo, pero para este TFG se ha decidido emplear Cucumber y Selenium
      </p>
    </div>
  </div>
```

Figura 63

```
<div class="cajaTitulo">
  <H3> ¿Qué es Cucumber?</H3>
</div>
<div class="cajaTitulo">
  <p>
    A pesar de que en el proyecto se está hablando de Cucumber, el lenguaje de programación utilizado es Gherkin, el cual se utiliza para desarrollar pruebas automatizadas. Cucumber no es más que una de las herramientas que proporciona Gherkin para poder desarrollar estas pruebas. Cucumber funciona a base de escenarios. Cuando se crea un escenario bajo la extensión ".feature", se deben tener en cuenta los siguientes pasos:
  </p>
</div>
<div class="cajaTextolista">
  <ul>
    <li style="...">Identificar el escenario de la prueba: nos referimos a escenario como el conjunto de acciones que se van a llevar a cabo para que la prueba siga las indicaciones que le da el programador. Cada escenario debe llevar un nombre y diferentes pasos a seguir. Existen dos tipos de escenarios:</li>
    <ul style="...">
      <li>
        Sceanario: es un escenario clásico. Se construye con los indicadores de los pasos
      </li>
      <li>
        Scenario Outline: funciona igual que un escenario clásico, salvo el detalle de que permite incluir variables en los pasos a ejecutar. Este tipo de escenarios se utiliza en ocasiones donde se quiere repetir un paso con diferentes valores.
      </li>
    </ul>
    <li style="...">Pasos a seguir: para que un escenario funcione, cada "paso" está definido por un indicador. De esta forma, podemos asemejar el lenguaje humano a lo que va a hacer cada escenario. Para esto, el escenario seguirá un orden ejecución basado en una lista de acciones: "Cuando tengo esta situación dado este ejemplo, entonces ejecutaré cierto comando y comprobaré el resultado. Los indicadores se marcan con las siguientes palabras reservadas
    <ul style="...">
      <li>Given</li>
      <li>When</li>
      <li>Then</li>
      <li>And</li>
    </ul>
  </li>
  </ul>
</div>
```

Figura 64


```

<div class="cajaFlexTexto">
  <div class="cajaTitulo">
    <H3> ¿Qué es Selenium Web Driver?</H3>
  </div>
  <div class="cajaTitulo">
    <p>
      Selenium Web Driver es una interfaz que simula las interacciones del usuario con cualquier navegador. En el caso de este proyecto, servirá de intermediario para recoger las órdenes escenificadas con Cucumber y posteriormente interactuar con el chatbot alojado en la web.
    </p>
    <p>
      La conexión que tienen Cucumber y Selenium es muy sencilla. Gracias a los escenarios definidos con Cucumber, Selenium cogerá el nombre de cada uno de los pasos definidos con los identificadores y se le asignarán órdenes para proceder con las pruebas. Por ejemplo, si la primera orden de un escenario en Cucumber es "Given I am on the main page", la programación de ese paso con Selenium permitirá crear la conexión con la página principal, abriendo el driver y mostrando la dirección web establecida.
    </p>
  </div>
</div>
<div class="cajaTitulo">
  <H5> ¿Cómo funciona la interacción de Selenium con el chatbot?</H5>
</div>
<div class="cajaTitulo">
  <p>
    Para poder hacer las pruebas automatizadas en el chatbot, se van a tener que recoger de forma manual los elementos "xpath" que componen el chatbot en cada mensaje. De esta forma, Selenium se encargará de leer el xpath encargado de darle al botón de abrir el desplegable del bot, para posteriormente ir leyendo los xpath de cada mensaje para comprobar que los textos, botones e interacciones son los esperados
  </p>
</div>
</div>
<div class="rowBoton">
  <div class="cajaBoton">
    <a class="btn btn-outline-primary btn-lg" href="">Visita la página oficial de Selenium Web Drive</a>
  </div>
</div>

```

Figura 65

Memoria.html

En esta clase se va a usar la división <embed>, la cual se va a utilizar para adjuntar el pdf de la memoria de este TFG.

```

<div class="rowCaja">
  <div class="cajaFlexTexto">
    <div class="cajaTextoTitulo">
      <H2>Memoria Explicativa del TFG</H2>
    </div>
    <div class="cajaVisorPdf">
      <embed src="docs/TFG.pdf" type="application/pdf" width="100%" height="100%"/>
    </div>
  </div>
</div>
/body>

```

Figura 66

En las siguientes figuras se mostrarán todas las clases creadas en main.css. En la mayoría de figuras se van a observar codificaciones bastante similares entre sí. El sentido de esto es que cada "caja" dividida en el .html va a tener una flexibilidad y un comportamiento diferente. Por ejemplo, el menú que va a aparecer en la parte superior, necesita unos márgenes distintos a los que tendrán los títulos de texto de cada página. De esta forma, se puede dinamizar el movimiento de las cajas y de los textos.

```

.pagina {
  background-image: repeating-radial-gradient
(circle at 0 0, transparent 0, #ffffff 50px),
repeating-linear-gradient(#faf6fd, #faf6fd);
background-color: #ffffff;

  width: 100%;
  height: 100%;
  display: flex;
  flex-direction: column;
  justify-content: start;
  font-family: "Montserrat";
}

.contenido{
  width: 100%;
  height: wrap-content;
  display: flex;
  flex-direction: column;
  font-size: 1.5vw;
  justify-content: start;
  align-items: center;
}

.menu {
  width: 100%;
  height: wrap-content;
  display: flex;
  flex-direction: row;
  font-size: 1.5vw;
  justify-content: space-between;
  align-items: center;
}

```

Figura 71

```

.logoMenu {
  width: 20%;
  height: wrap-content;
  margin: 1%;
}

.accionesMenu {
  width: 50%;
  height: wrap-content;
  margin: 1%;
  display: flex;
  flex-direction: row;
  justify-content: center;
}

.logoMenu a {
  text-decoration: none;
  color: black;
}

.accionesMenu a {
  text-decoration: none;
  color: black;
}

.derechaMenu {
  width: 20%;
  height: 100%;
  margin: 1%;
}

```

Figura 70

```

.accionesMenu a {
  text-decoration: none;
  color: black;
}

.derechaMenu {
  width: 20%;
  height: 100%;
  margin: 1%;
}

.cajaAccionesMenu {
  width: fit-content;
  height: 100%;
  margin: 1%;
  margin-left: 3%;
  margin-right: 3%;
  font-size: 1.1vw;
  text-align: center;
}

.rowCaja {
  width: 100%;
  height: wrap-content;
  display: flex;
  flex-direction: row;
  justify-content: space-between;
  align-items: start;
}

.cajaTextoTitulo{
  width: wrap-content;
  height: wrap-content;
  margin: 1%;
  margin-top: 10%;
  margin-left: 17%;
  margin-right: 17%;
  text-align: center;
}

```

Figura 69

```

.cajaTexto{
  width: wrap-content;
  height: wrap-content;
  margin: 1%;
  margin-left: 17%;
  margin-right: 17%;
  text-align: center;
}

.cajaTextoLista{
  width: wrap-content;
  height: wrap-content;
  margin: 1%;
  margin-left: 17%;
  margin-right: 17%;
}

.cajaFlexTexto {
  width: 100%;
  height: wrap-content;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  align-items: center;
  font-size: 1.2vw;
}

.cajaFlexTexto2 {
  width: 100%;
  height: 100%;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  align-items: center;
  font-size: 1.2vw;
  margin-right: 17%;
  margin-left: 17%;
}

```

Figura 68

```

.cajaTitulo {
  width: wrap-content;
  height: wrap-content;
  margin: 1%;
  margin-left: 17%;
  margin-right: 17%;
  text-align: center;
}

.rowBoton {
  width: 100%;
  height: 100%;
  display: flex;
  flex-direction: row;
  font-size: 1vw;
  font-weight: 50;
  justify-content: center;
  align-items: center;
}

.cajaBoton {
  width: wrap-content;
  height: wrap-content;
  margin: 1%;
  margin-bottom: 3%;
}

.cajaFlexTexto h2 {
  font-size: 1.9vw;
  font-weight: 1000;
}

.cajaVisorPdf{
  width: 80%;
  height: 800px;
  margin: 2%;
}

```

Figura 67

ChatGPT y las inteligencias artificiales

No podemos negar el impacto gigantesco que han tenido las Inteligencias Artificiales en el ámbito de los chatbot en el último año. Desde que empecé a trabajar en este proyecto, CHATGPT ha crecido de forma exponencial hasta ser conocido por prácticamente todo el mundo.

ChatGPT es una aplicación desarrollada en 2022 por OpenAI programado con Python. La característica de este chatbot es que utiliza técnicas de aprendizaje reforzado y supervisado para aprender sobre los temas que se le preguntan. Gracias a este aprendizaje, es capaz de dar respuestas muy concretas y con bastante veracidad, basándose en todo lo que ha ido aprendiendo de los usuarios que lo han usado.

ChatGPT ofrece la posibilidad de crear tu propio chatbot, una aplicación llamada Chatbase. Para hacer una comparación de lo que puede hacer un chatbot creado con inteligencia artificial contra el chatbot que se ha creado en este proyecto desde cero de forma manual, voy a crear un prototipo de chatbot con Chatbase para hablar de las principales diferencias.

Chatbase

En primer lugar, es obligatorio registrarse en el [sitio web](#) de Chatbase para poder configurar el chatbot.

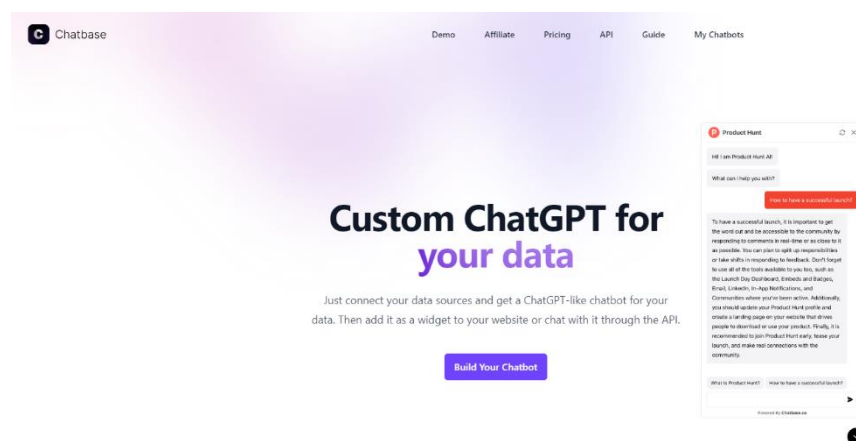
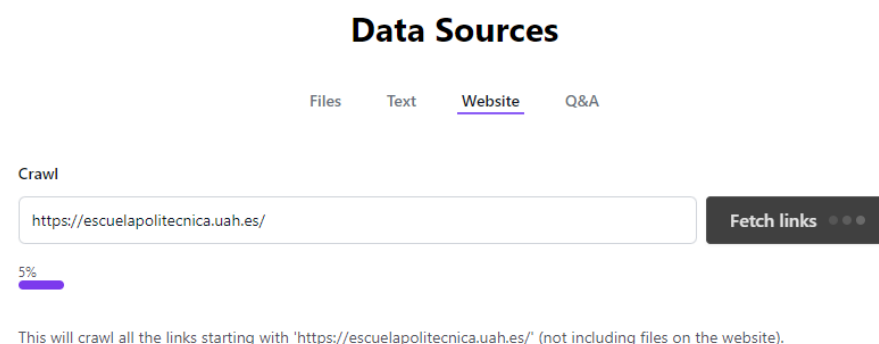


Figura 72

Nada más registrarse, al ir a crear el chatbot, he optado por configurarlo via web. Con esto, se introduce la url de la Escuela Politécnica Superior y de esta forma el chatbot recogerá todos los enlaces relacionados con esa web.



El problema de esto es que, para la versión gratuita, únicamente permite reconocer un total de 400.000 caracteres entre todos los enlaces que recoja del enlace principal que se le ofrece. La web de la Escuela Politécnica Superior suma más de un millón de caracteres en total.

Para poder proseguir con esta prueba, he quitado los enlaces que, aunque son necesarios, veía algo menos relevantes para los alumnos de nuevo ingreso, como información sobre la cafetería o las cátedras. También he borrado enlaces a los dobles grados, así como el grado de Matemáticas y Computación (por ser un nuevo grado), así como las asignaturas de los dobles grados, del grado de Telemática y del grado de Electrónica de comunicaciones.

La selección final de los enlaces seleccionados es la que se muestra en la figura 56 y 57

Included Links		Delete all
https://escuelapolitecnica.uah.es/	10880	
https://escuelapolitecnica.uah.es/estudiantes/practicas-externas.asp	10502	
https://escuelapolitecnica.uah.es/estudiantes/trabajo-fin-grado.asp	11127	
https://escuelapolitecnica.uah.es/estudiantes/nuevos-alumnos.asp	9173	
https://escuelapolitecnica.uah.es/estudios/grados.asp	9642	
https://escuelapolitecnica.uah.es/estudiantes/horarios.asp	9528	
https://escuelapolitecnica.uah.es/estudiantes/fechas-examenes.asp	9293	
https://escuelapolitecnica.uah.es/inicio.asp	4256	
https://escuelapolitecnica.uah.es/estudiantes/asignaturas.asp	8644	
https://escuelapolitecnica.uah.es/escuela/normativa.asp	8150	
https://escuelapolitecnica.uah.es/tramites/formularios.asp	12955	
https://escuelapolitecnica.uah.es/estudiantes/programas-internacionales.asp	11612	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G781	16093	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G390	14031	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G581	17859	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G370	13448	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G350	14364	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G380	13550	

Figura 74

https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G610	12745	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G60	13604	
https://escuelapolitecnica.uah.es/estudios/grado-int.asp?plan=G591	19221	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G781	20149	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G610	18131	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G60	19509	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G581	18842	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G591	20447	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G350	25620	
https://escuelapolitecnica.uah.es/estudios/1stAsignaturas-v3.asp?CodPlan=G390	22767	

396,142 detected characters

Add

Included sources:
28 Links (396,142 detected chars)

Total detected characters: **396,142** / 400,000 limit

Create Chatbot

Figura 76

El siguiente paso sería directamente probar el chatbot. Según la interfaz principal, parece que solo se permiten 30 preguntas diarias al chatbot.

escuelapolitecnica.uah.es

Chatbot Settings Dashboard Sources Integrations **New** Embed on site Share Delete

Still training your chatbot on the provided links
This will take a few minutes

Hi! What can I help you with?

30 message credits left

Hi! I am ChatbaseAI, ask me anything about Chatbase!

By the way, you can create a chatbot like me for your website!

C

Figura 75

Para probar este chatbot, se van a introducir algunas frases definidas manualmente en el chatbot del proyecto.



Figura 77

En la figura 59 se le pregunta al chatbot de Chatbase por el sitio web general. Mientras que en el chatbot configurado en el proyecto esta pregunta redirige directamente a la web principal de la Escuela Politécnica Superior, aquí interpreta la pregunta de forma más general, y hay que especificarle qué web se quiere buscar. Con la segunda pregunta, proporciona la respuesta adecuada.

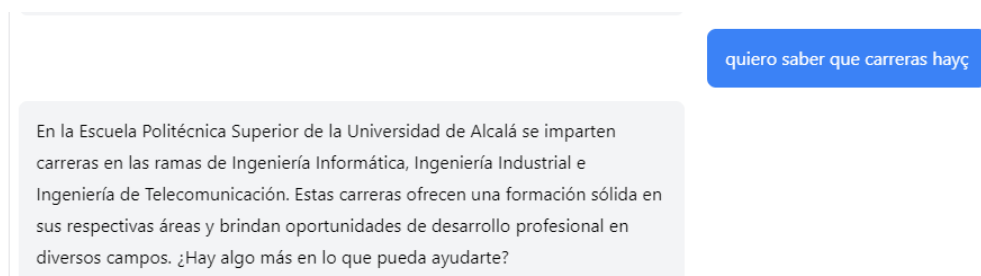


Figura 78

Con esta pregunta, mi intención era que el chatbot me devolviera una lista con todos los grados disponibles en la EPS. Sin embargo, parece que únicamente está reconociendo Ingeniería Informática, Ingeniería Industrial e Ingeniería de Telecomunicación, que no son grados, sino las ramas principales que engloban a todos los grados que tiene la escuela. Para profundizar un poco más, he reformulado la pregunta:

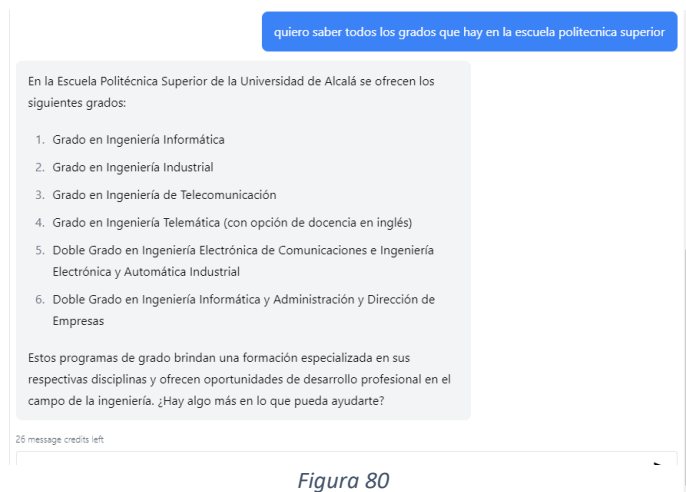


Figura 80

Con esto se reafirma lo anterior. Está seleccionando los posibles campos de estudio, pero no todos los grados disponibles. Sin embargo, sí que asocia el grado de Ingeniería Telemática y los Dobles Grados como carreras aparte de los campos de estudio. Se puede comprobar en la figura 62 (sacada de la web de la EPS) que la lista mostrada en la figura 61 no es idéntica.




Estudios de Informática

- [Grado en Ingeniería de Computadores](#) Opción docencia inglés
- [Grado en Ingeniería en Sistemas de Información](#)
- [Grado en Ingeniería Informática](#) Opción docencia inglés
- [Grado en Matemáticas y Computación](#) **Guadalajara. ¡Nuevo!**

Estudios de Industriales

- [Grado en Ingeniería en Electrónica y Automática Industrial](#) Opción docencia inglés 
- [Grado en Ingeniería en Tecnologías Industriales](#)

Estudios de Telecomunicación

- [Grado en Ingeniería Electrónica de Comunicaciones](#) Opción docencia en inglés 
- [Grado en Ingeniería en Sistemas de Telecomunicación](#) Opción docencia en inglés 
- [Grado en Ingeniería en Tecnologías de Telecomunicación](#) Opción docencia en inglés
- [Grado en Ingeniería Telemática](#) Opción docencia en inglés 

Dobles Grados


- [Doble Grado en Ingeniería Electrónica de Comunicaciones e Ingeniería Electrónica y Automática Industrial](#) 
- [Doble Grado en Ingeniería Informática y Administración y Dirección de Empresas](#)

Figura 79

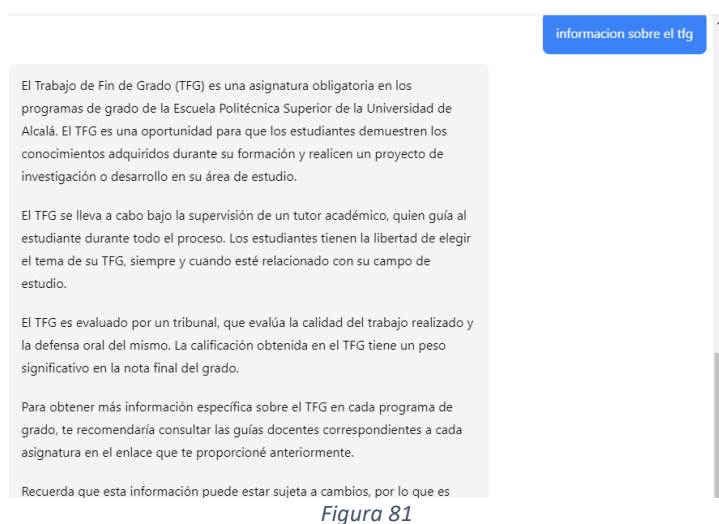


Figura 81

También se le ha preguntado sobre los trabajos de fin de grado, aunque su respuesta ha sido más bien una descripción general que el enlace donde se puede encontrar toda la información concreta necesaria para poder afrontar el trabajo. La respuesta deriva al usuario a las guías docentes de cada asignatura, pero no muestra ningún enlace. Cabe decir que es posible que no tenga acceso a los enlaces por la limitación de caracteres, donde se tuvieron que eliminar diversos enlaces de acceso.

Como último ejemplo, se le ha preguntado por las fechas de exámenes de un grado en concreto. El chatbot, como se puede observar en la figura 64, dice que no tiene acceso a los datos en tiempo real ni a las fechas de exámenes. Esto puede ser debido a que los ficheros que se suben a la web indicando fechas de exámenes son dinámicos y cambian cada año. La respuesta proporciona las opciones de buscar en los documentos oficiales de la Universidad o contactar con el departamento académico.

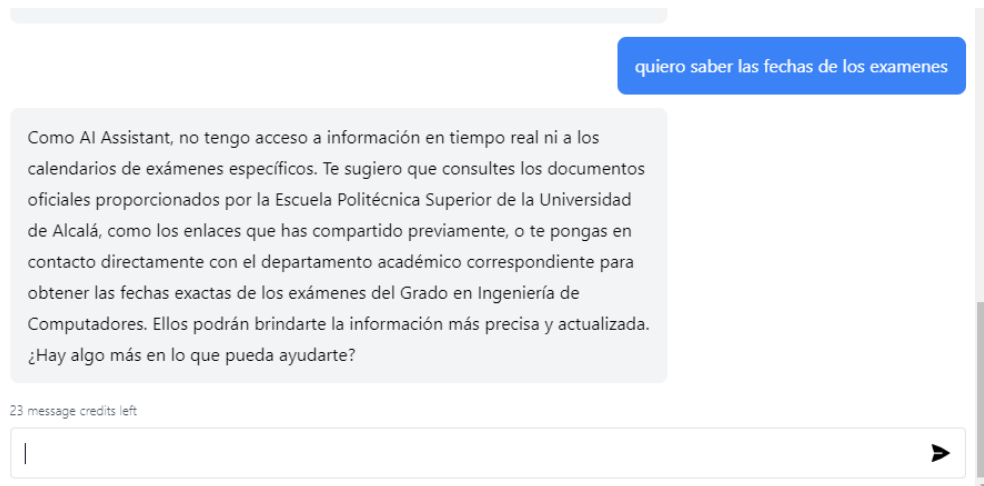


Figura 82

Como conclusión, quizá una versión de pago con más capacidad para entrenar al chatbot podría resultar óptima en el sentido de que es mucho más fácil de configurar y requiere mucho menos tiempo y esfuerzo. La inteligencia artificial avanza a pasos agigantados y no se puede negar el impacto que tiene en nuestra sociedad. Pero en este caso en particular, la programación manual del chatbot ofrece la garantía de que la respuesta que el chatbot le da al usuario va a ser la que el programador desea. Un trato manual con este tipo de asistentes proporciona una estabilidad objetiva a la hora de responder. Con una inteligencia artificial, no se podrían hacer pruebas automatizadas, dado que la IA cada vez responderá con una frase diferente, y no se puede controlar el camino que va a tomar esa respuesta.