

UNIVERSIDAD POLITÉCNICA DE PUEBLA
Ingeniería en Tecnologías de la Información



**Proyecto de Estancia Práctica en
Desarrollador en Sistemas de Software y
Administrador de Redes**

**“Desarrollo de interfaz visual para rutinas de movilidad para
robot Baxter”**

Área temática del CONACYT: VII
Ingenierías y tecnologías

Presenta:
Nava Dionicio Gerardo

Asesor técnico
Dr. Antonio Benítez Ruiz
Asesor académico
Dr. Antonio Benítez Ruiz

Juan C. Bonilla, Puebla, México.

29/abril/2022

Resumen

En este documento se presentará como se lleva a cabo el proyecto del desarrollo de una interfaz visual de escritorio para la futura manipulación del robot Baxter que se encuentra en la Universidad Politécnica de Puebla.

Se eligió la Universidad Politécnica de Puebla que es una institución educativa de gobierno, la cual ofrece 7 ingenierías, 5 maestrías y 2 especialidades al público en general, este proyecto se enfocará solamente en la ingeniería en tecnologías de la información donde se llevó a cabo el proyecto asignado conforme a los conocimientos adquiridos previamente en asignaturas ya cursadas, de igual manera se asignó un asesor técnico, docente de la misma para su supervisión.

En el capítulo 1 se hará mención del objetivo general y los específicos que se cumplieron para el desarrollo de este proyecto, en el capítulo 2 se describe la metodología de desarrollo para una interfaz visual, utilizada para llevar el control de todas las actividades, utilizando herramientas como Python, Py Qt Designer 5 y usando de base el sistema operativo Windows10.

En el capítulo 3 se muestran las actividades realizadas a lo largo de todo el proceso de estancia 2, para obtener el desarrollo de interfaz visual de escritorio, para el futuro control del robot Baxter a través de ella.

Índice

1. Introducción	4
2. Metodología y herramientas	5
3. Resultados	19
4. Conclusiones y recomendaciones	33
5. Referencias bibliográficas	34
6. Apéndice a - código de programación de la interfaz visual	36
6. Apéndice b – Desarrollo de cuestionario de evaluación	42

1. Introducción

1.1. Descripción del problema o necesidad

En la Universidad Politécnica de Puebla cuenta con un cobot (robot colaborativo llamado Baxter), con el que distintos estudiantes de la universidad pueden realizar proyectos programándolo para que realice ciertas funciones. Las funciones se ejecutan mediante códigos de programación ejecutándolos a través del sistema operativo Ubuntu, el problema radica en tener que teclear cada uno de los comandos a través de la línea de comando

1.2 Justificación

Considerando la problemática mencionada la solución es diseñar una interfaz visual, con distintas opciones que a través de ella se planea poder ejecutar funciones para el robot y tener un manejo más amigable para el usuario.

Al desarrollarse una interfaz visual, será más amigable con los usuarios que lleguen a controlar a Baxter, además que esto simplificará el tiempo en su ejecución cuando se tengan que hacer demostraciones del mismo.

También se planteó esta solución, ya que sirve de refuerzo y práctica para los conocimientos adquiridos previamente en la institución antes mencionada.

1.3 Objetivo General y Específicos

Objetivo General

Desarrollar una interfaz visual de escritorio dirigida al control de movilidad de rutinas para el robot Baxter de la Universidad Politécnica de Puebla.

Objetivos Específicos

- Proponer un diseño de interfaz visual que se acople a las tecnologías en las que se programa el robot Baxter.
- Desarrollo de Interfaz visual que trabaje bajo S.O. Ubuntu

2. Metodología y herramientas

2.1 Herramientas tecnológicas

2.1.1 Python

Python es un lenguaje de programación que le permite trabajar más rápidamente e integrar sus sistemas de manera más efectiva. [1]

Ventajas

Posee diversas ventajas sobre todo porque tiene diversas aplicaciones en las empresas dedicadas al desarrollo de software, como frameworks, aplicaciones web, creación de prototipos, etc.

Esto le proporciona a Python una ventaja competitiva frente a otros lenguajes de programación, algunas de sus ventajas son:

- **Lenguaje de alto nivel**
es un lenguaje de alto nivel, por lo que es más fácil de usar que los de bajo nivel, el lenguaje que maneja tiene una sintaxis similar al inglés, por lo que es el más fácil de leer, escribir y aprender.
- **Polivalente y de paradigmas**
Al ser un lenguaje de propósito general se puede usar para diversos propósitos, ya que permite a los desarrolladores utilizar frameworks, además de utilizarse para scripts web, y para el propósito de este documento Desarrollo de Gui de escritorio
- **Bibliotecas y frameworks**
La biblioteca estándar de Python es muy extensa, puesto que contiene muchos módulos integrados. Además, los usuarios de Python también pueden encontrar bibliotecas adicionales disponibles en PyPI (índice de paquetes de Python).
- **Portabilidad**
Python es compatible con todos los sistemas operativos (macOS, Linux, UNIX y Windows), y los programadores solo necesitan escribir código una vez y luego podrá ejecutarse en todas partes.

- **Gratis y de código abierto**
Python es un lenguaje de programación desarrollado bajo la licencia de código abierto aprobada por OSI, todos pueden usarlo y distribuirlo libremente.
- **Baja curva de aprendizaje**
La sencillez de la sintaxis de Python permite escribir programas totalmente funcionales en pocas líneas de código, por lo que su curva de aprendizaje es muy baja.
- **Comunidad fuerte**
Los programadores de Python pueden descargar el código fuente, modificarlo y distribuirlo como deseen, ya que es de código abierto y libre.

Desventajas

- **Lentitud**
La lentitud de Python se debe principalmente a su naturaleza dinámica y versatilidad.
- **Consumo de memoria**
En el caso de que una tarea requiera mucha memoria, Python no es la mejor opción. El consumo de memoria de Python es muy alto, y esto se debe a la flexibilidad de los tipos de datos.
- **Desarrollo móvil**
Python es ideal para plataformas de escritorio y servidor, pero para el desarrollo móvil no es un lenguaje muy adecuado. Por este motivo, apenas vemos aplicaciones móviles desarrolladas con Python. [2]

2.1.2 PyQt5 Designer

Qt Designer le ayuda a crear una GUI (interfaz visual de usuario). Puede cargar una GUI desde Python [3]

Como se menciona PyQt es una biblioteca de Python que, nos permite crear interfaces visuales de manera rápida y sencilla, al ser lenguaje Python provoca que sea sumamente sencillo manipular sus acciones mediante programación. [5]

Ventajas

- Versatilidad de codificación:
la programación GUI con Qt se basa en la idea de señales y ranuras para crear contacto entre objetos. Esto permite versatilidad en el tratamiento de incidentes de GUI, lo que da como resultado una base de código más suave.
- Más que un marco:
Qt utiliza una amplia variedad de API de plataforma nativa para redes, desarrollo de bases de datos y más. Proporciona acceso principal a ellos a través de una API especial.
- Varios componentes de la interfaz de usuario:
Qt proporciona múltiples widgets, como botones o menús, todos diseñados con una interfaz básica para todas las plataformas compatibles.
- Varios recursos de aprendizaje: como PyQt es uno de los sistemas de interfaz de usuario más utilizados para Python, puede acceder cómodamente a una amplia variedad de documentación.

Desventajas

- Falta de documentación específica de Python para clases en PyQt5
- Se necesita mucho tiempo para comprender todos los detalles de PyQt, lo que significa que es una curva de aprendizaje bastante empinada.
- Si la aplicación no es de código abierto, debe pagar una licencia comercial. [4]

¿Por qué se decidió usar esta herramienta?

Se selecciona esta herramienta debido a que es una de las múltiples opciones con las que cuenta Python, de esta manera se garantiza un diseño de interfaz más amigable y entendible al usuario.

2.1.3 Visual Studio Code

Es un editor de programación multiplataforma desarrollado por Microsoft, es un software libre que se distribuye bajo la licencia MIT, aunque los ejecutables se distribuyen bajo una licencia gratuita no libre. [6]

Este software está disponible para plataformas de Windows, Linux y MacOS. Incluye soporte para depuración, control de Git integrado, resaltado de sintaxis, finalización de código inteligente, fragmentos de código y refactorización de código. También es personalizable, de modo que los usuarios pueden cambiar el tema del editor, los métodos abreviados de teclado y las preferencias. Es gratuito y de código abierto. [7]

Ventajas

- Libre.
- Desarrollado por Microsoft.
- Fantástico Debugging.
- Preparado. Solo instalas y a trabajar. Lo básico ya viene preestablecido.
- Plugins para toda la familia. A pesar de la juventud, no te faltará de nada.
- Fácil a la hora de instalar plugins, configurar, cambiar tema.
- Git integrado.
- Aparente sucesor de Sublime Text. Coge todas sus ideas y las mejora, sin miedos.

Desventajas

- Pesado. Su base es Electron (Chrome). Prácticamente es como tener un navegador abierto.
- No tan rápido. Notas un ligero retraso cuando abres archivos, te mueves con el scroll o editas ficheros grandes.
- No funciona en terminal, aunque si contiene un terminal.
- Es tan sencillo de utilizar que puede llegar a aburrirte.

¿Por qué se decidió usar esta herramienta?

La decisión de usar esta herramienta es por comodidad propia, ya que no es el único editor que he utilizado, pero si con el que más me siento cómodo y para el proyecto en gestión me resultará más simple llevarlo a cabo debido a las facilidades con las que cuenta.

2.2 Páginas/Aplicaciones Web consultadas

2.2.1 Facebook Oficial de la Universidad Politécnica de Puebla

Es una universidad con el objetivo de formar íntegramente profesionales competentes que atiendan necesidades de los sectores productivo y social, mediante el desarrollo tecnológico, la innovación y la investigación aplicada, promoviendo una cultura ambiental y de equidad de género.

Se consultó esta página para poder obtener el logotipo oficial de la Universidad, así como el logotipo de búhos blancos. [6]

2.2.2 IMAGECOLORPICKER.COM

Es una aplicación web dónde puede cargar una imagen en cualquier formato y detectará los colores que contiene la imagen proporcionándolos en código hexadecimal y RGB.

Se utilizó la siguiente aplicación web para poder obtener los colores oficiales del logotipo de la Universidad Politécnica de Puebla. [7]

2.2.3 Flacticon

Se utilizó la siguiente página web para descargar iconos utilizados en la interface [8]

2.2.4 freepng.es

Es una página web que contiene miles de imágenes en distintos formatos disponibles al público en general.

Se utilizó la siguiente página web para descargar imágenes del cobot Baxter para su utilización en la interface. [9]

2.2.5 ROBOTS

Es una página web que contiene miles de imágenes en distintos formatos disponibles al público en general.

Se utilizó la siguiente página web para descargar imágenes del cobot Baxter para su utilización en la interface [10]

2.3 Metodología de diseño de interfaces UI

El conjunto de pasos a seguir para llevar a cabo el proyecto, son los siguientes:

1. Análisis y diseño de la interfaz visual.
2. Selección de herramientas de programación.
3. Desarrollo de la interfaz.
4. Cuestionario de satisfacción del usuario de la interfaz.

¿En qué consiste el diseño UI?

Como hemos visto anteriormente, el diseño UI está integrado en los engranajes del diseño UX. Situamos el diseño UI en la fase de implementación, esta es la etapa donde las ideas y prototipos de los diseñadores UX cobran vida.

Cuando hablamos de diseño UI (user interface design), nos referimos al diseño de la interfaz de usuario y, por lo tanto, a la experiencia estética que resulta de ella.

¿En qué consiste el trabajo del diseñador UI?

El rol del diseñador UI está enfocado en los aspectos gráficos y visuales de la interfaz. Por lo tanto, los elementos perceptibles por el usuario. Por ejemplo, el color y tamaño de un botón para que sea visible, que llame la atención y que den ganas de clicar en él; o la tipografía de los textos que se muestran en pantalla. [1]

¿Por qué se decidió usar esta herramienta?

Se decidió usar esta metodología, ya que lo que se busca es generar un tipo de experiencia agradable para todo aquel que llegue a interactuar con la interfaz.

2.4 Desarrollo de la interfaz visual

1. Obtención de la Imágenes correspondientes para la interfaz.

En esta sección, las imágenes utilizadas son:



Figura 1. Logo Uppuebla



Figura 2. Logo Uppuebla



Figura 3. Logo Uppuebla



Figura 4. Logo Búhos Blancos

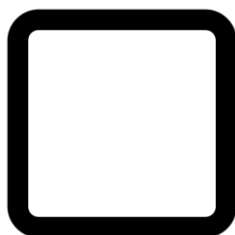


Figura 5. logo Maximizar ventana



Figura 6. Logo restaurar ventana

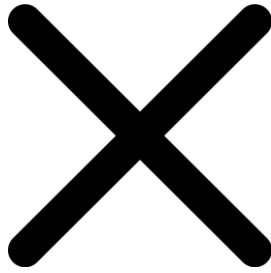


Figura 7. Logo Cerrar ventana



Figura 8. Logo Minimizar Ventana



Figura 9. Logo Git Hub



Figura 10. Baxter brazo



Figura 11. *Baxter tamaño completo*



Figura 12. *Baxter tamaño completo*

2. Objetos de PyQt5 utilizados

a) Frame

QFrame es una clase base que se puede usar directamente; se usa principalmente para controlar algunos estilos de borde, como elevado, empotrado, sombreado, ancho de línea, etc. [11]

El uso en el diseño de esta interfaz, es para poder separar los distintos objetos en diversas pantallas movibles.

b) Label

Una etiqueta es un elemento de control gráfico que muestra texto en un formulario. Suele ser un control estático; no tener interactividad. Una etiqueta se utiliza generalmente para identificar un cuadro de texto cercano u otro widget. [12]

El uso en el diseño de esta interfaz, es para mostrar imágenes y textos.

c) Button

El button de comando, es quizás el widget más utilizado en cualquier interfaz visual de usuario. Presione (haga clic) en un botón para ordenar a la computadora que realice alguna acción o que responda a una pregunta. [13]

El uso en el diseño de esta interfaz, es para que el usuario, al dar clic pueda realizar determinadas funcionalidades en Baxter.

d) Tool box

Toolbox) es un widget contenedor en PyQt. El widget puede mostrar grupos de elementos separados por pestañas. Si hay muchos elementos para una barra de herramientas, es posible que desee una caja de herramientas. [14]

El uso en el diseño de esta interfaz, es para incrustar buttons y ejecute diversas instrucciones

e) stackedWidget

proporciona una pila de widgets donde solo un widget es visible a la vez.

El uso en el diseño de esta interfaz, es para incrustar labels y button, se esta manera mostrará imágenes de las acciones correspondientes.

3. Desarrollo de la interfaz.

A continuación, se describirá a grandes rasgos el desarrollo de la interfaz visual.

1. Como primer paso, se incrustan 3 frame en la pantalla inicial, y se acomodarán de manera horizontal, haciendo que el frame superior e inferior sean de menor tamaño al central.

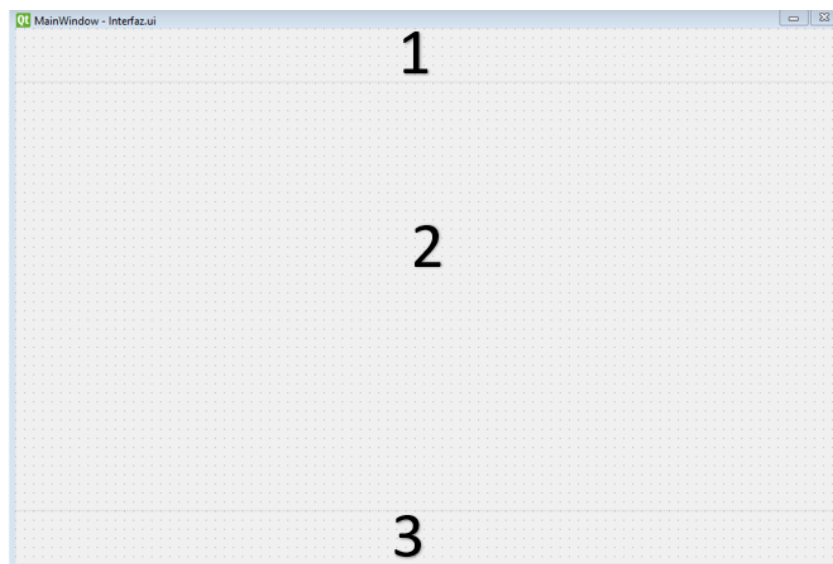


Figura 13. interfaz parte 1

2. Cambiar los colores de cada frame de la siguiente manera.
Los colores utilizados, se obtuvieron del logo de la UPPUE con la página [IMAGECOLORPICKER.COM](https://imagecolorpicker.com), quedando de la siguiente manera

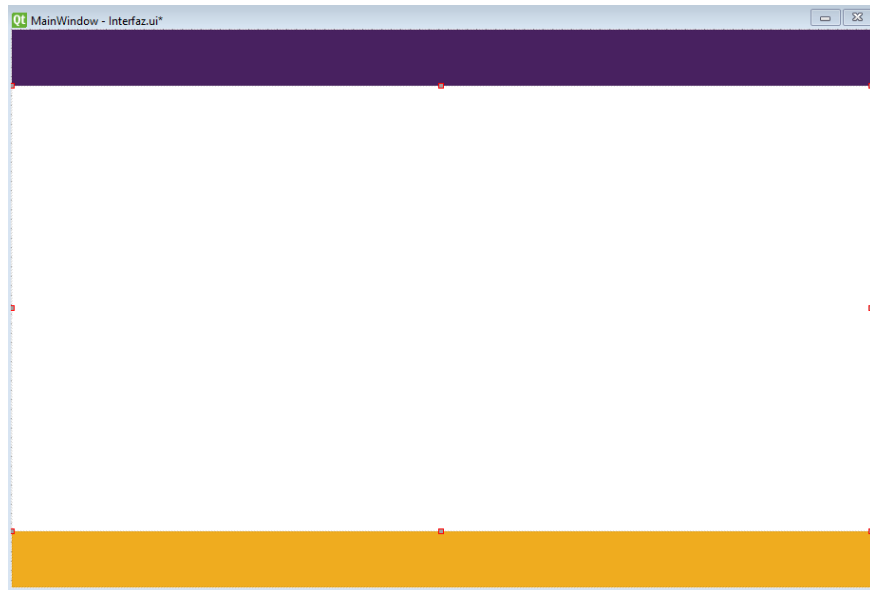


Figura 14. interfaz parte 2

3. Incrustar 5 buttons, una label y un separador en el frame superior y una label y un button en la parte inferior, amos con sus respectivos iconos.

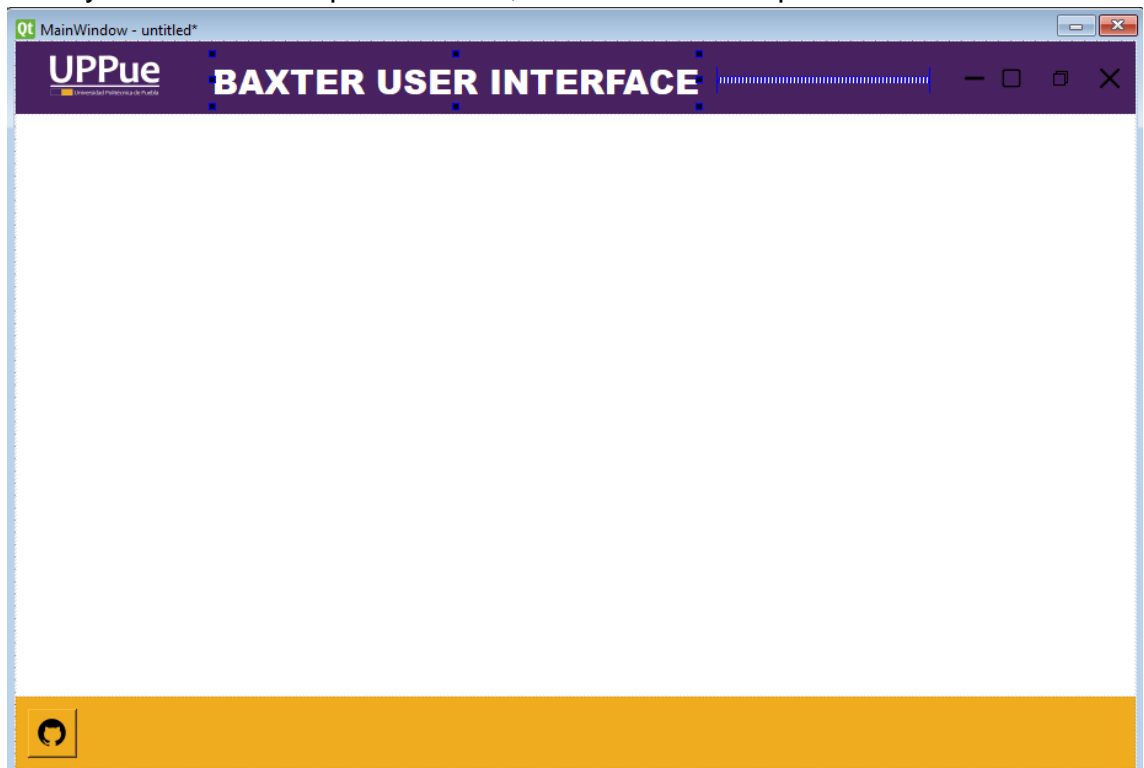


Figura 15. interfaz parte 3

4. Ahora se añaden 3 frame más en el frame central y se alinean de manera vertical y se rellena el fondo con el color de su preferencia.

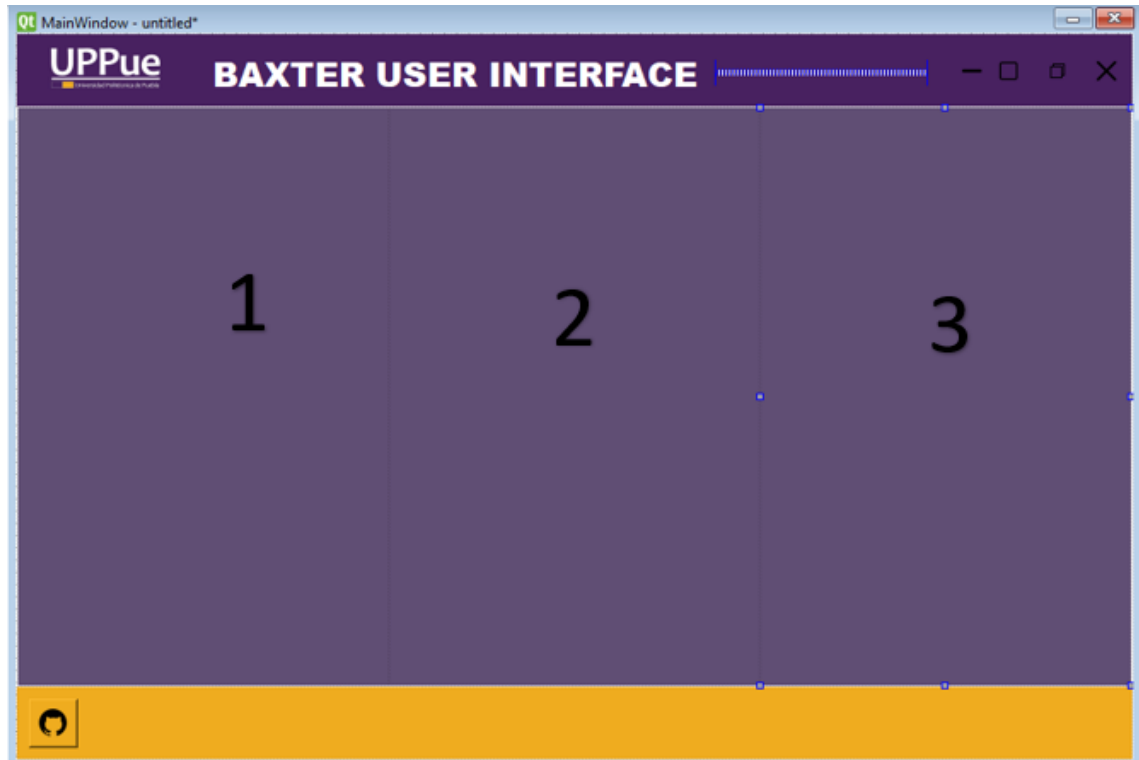


Figura 16. interfaz parte 4

5. Inserción de Objetos para las funciones

- En el frame lateral izquierdo se insertan 4 buttons con su respectivo nombre que ejecutará
- El frame del centro se le insertaran 2 frame, en el inferior se agregará un tool box con su respectivo button y funcionalidad.
- En el frame lateral derecho contendrá un stackwidget, con el número de páginas iguales a los buttons declarados en el frame lateral derecho, además de que cada página constará con cierto número de buttons para cada funcionalidad, además de una imagen ilustrativa.

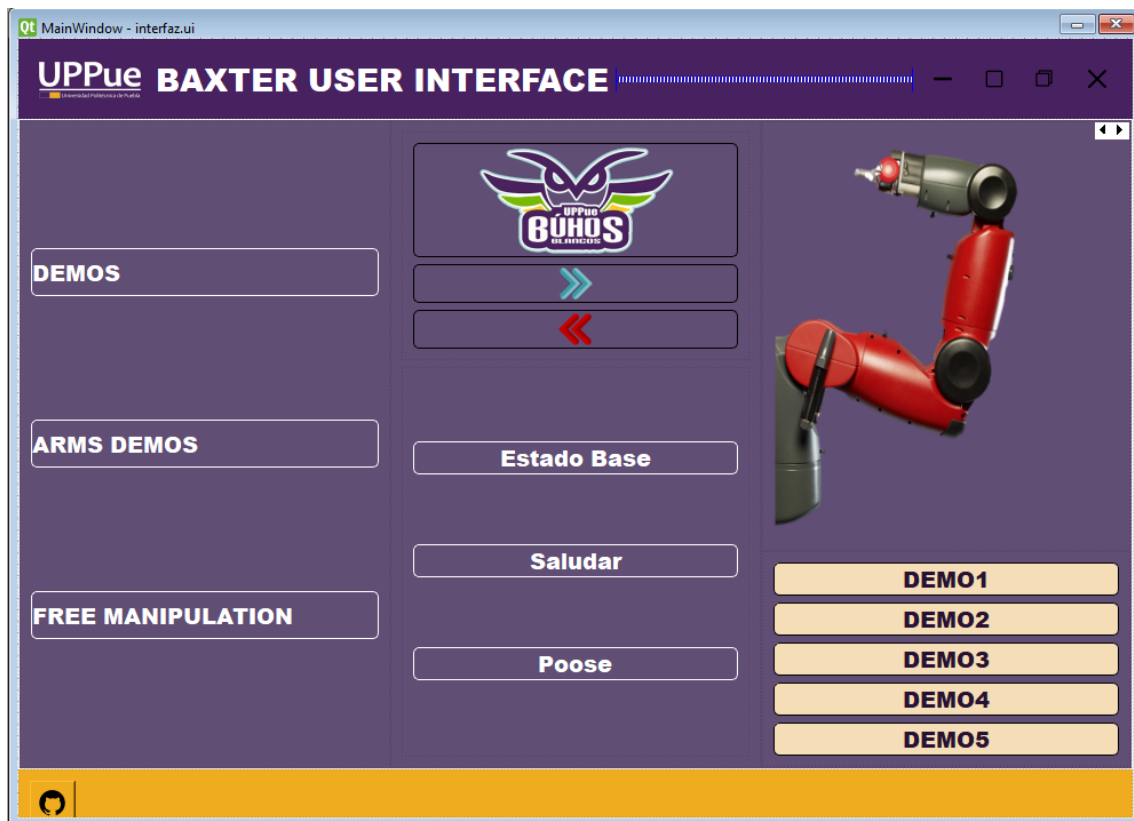


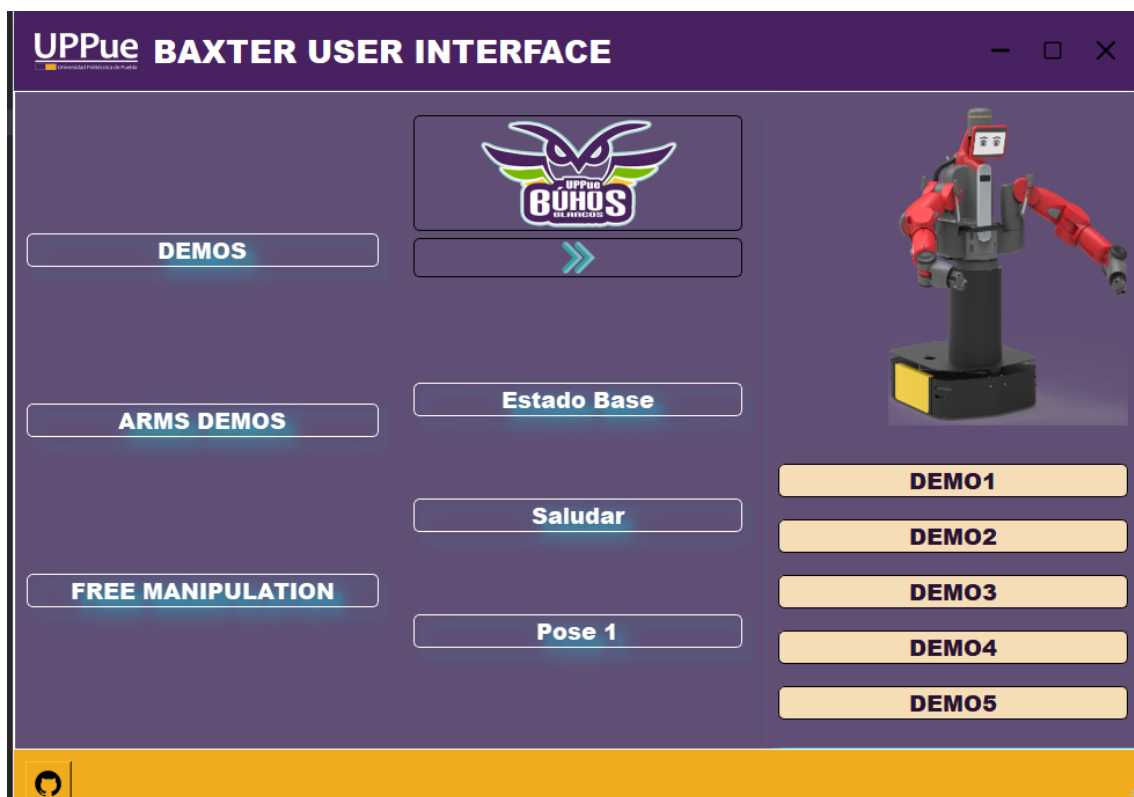
Figura 17. interfaz demos parte 5

3. Resultados

3.1 Descripción de Interfaz visual

A continuación, se presenta el proyecto finalizado, es importante mencionar que el código de programación en lenguaje Python a través del editor de programación visual studio code se describe en el apéndice a.

Como resultado final se obtuvo la siguiente interfaz visual



Descripción de las funcionalidades de la interfaz

En el lado izquierdo se cuentan con 3 funcionalidades que pueden ejecutarse para el control de Baxter, al hacer clic en cada una se mostrará una página del lado derecho con bonotes asignados a una función de movilidad de Baxter.

En esta primera vista, se muestra les lado derecho cinco buttons que posteriormente ejecutará funcionalidades de demostración de Baxter.

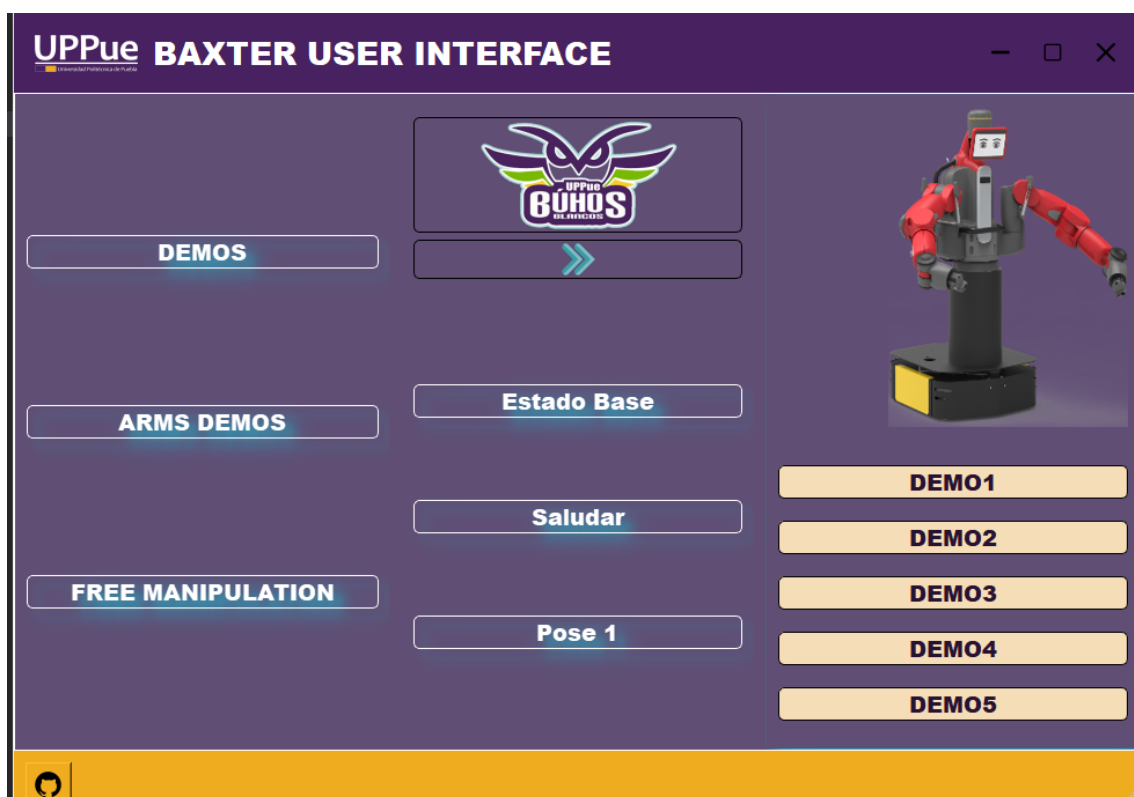


Figura 18 button y página de Demos de Baxter

En esta segunda vista, se muestra los lado derecho cinco buttons que posteriormente ejecutará funcionalidades de movimientos de brazos de Baxter.

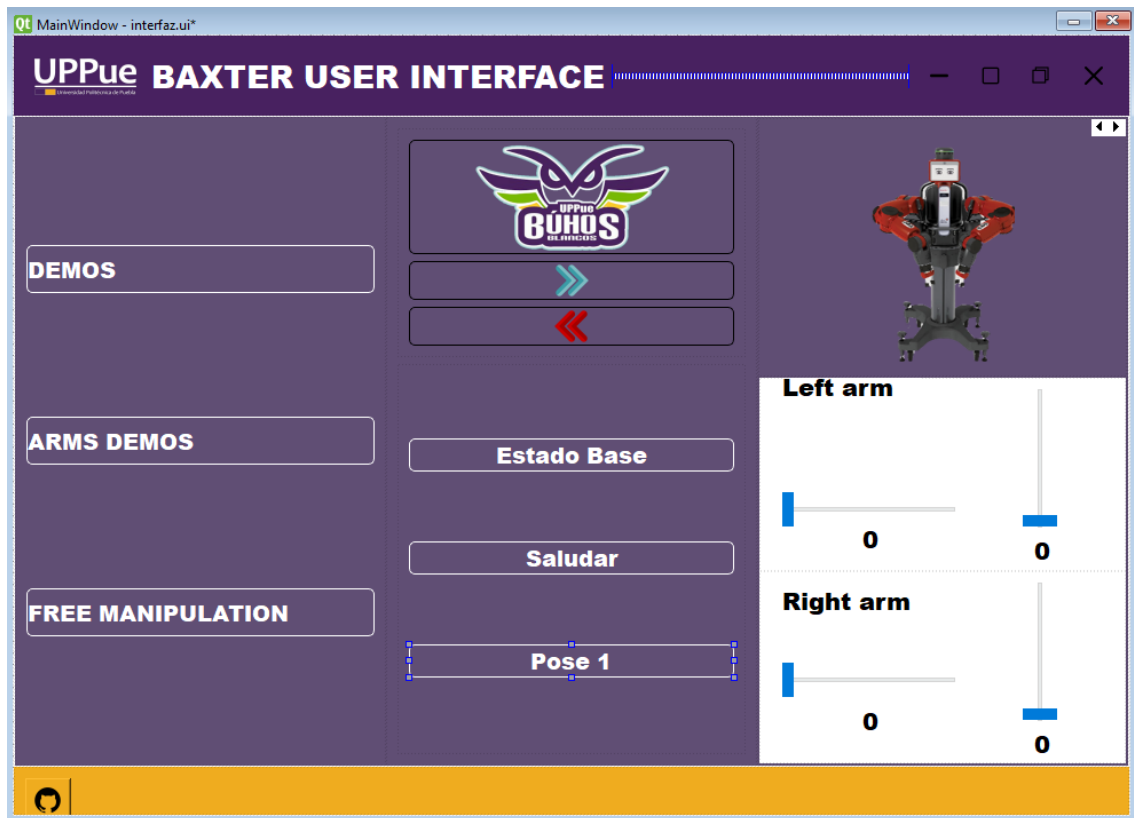


Figura 19 button y página de Arms Demos de Baxter

En esta tercera vista, se muestra del lado derecho 2 vertical sliders y dos horizontal sliders y unas etiquetas, con las sliders puede controlar el movimiento vertical y horizontal de cada brazo con los que cuenta Baxter, posteriormente en una etiqueta cercana a las sliders que mostrarán en que posición se encuentran, cabe mencionar que el límite de movimiento será de 100 e irán avanzando por una unidad.

En la siguiente imagen se puede notar los siguientes parámetros

Left Arm horizontal “72”

Left arm vertical “55”

Y

Right Arm horizontal “26”

Right arm vertical “64”

Esas será las coordenadas en las que se planea que se mueva Baxter.

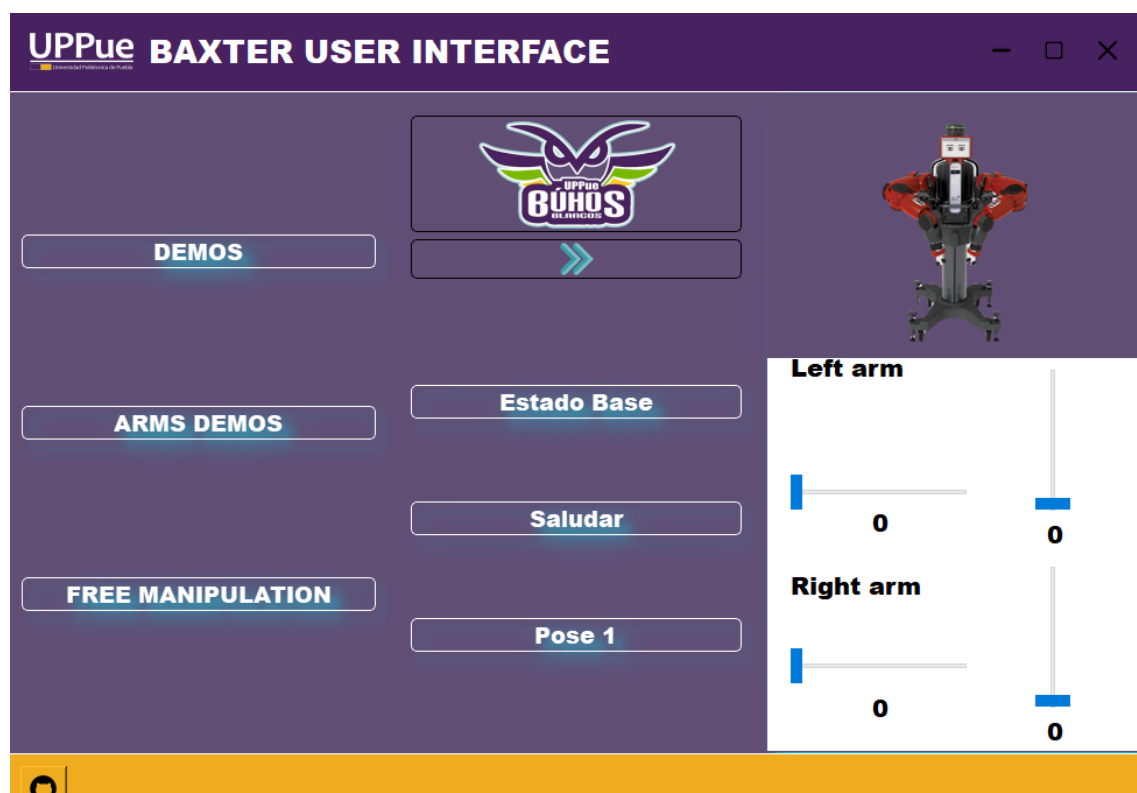


Figura 20 button y página de free manipulación de Baxter

Posteriormente en la parte central, se puede destacar el logo de “Búhos blancos”, logo de la mascota oficial de la Universidad Politécnica de Puebla, que al dar clic sobre ella se direccionará a la página oficial de la universidad en cuestión.

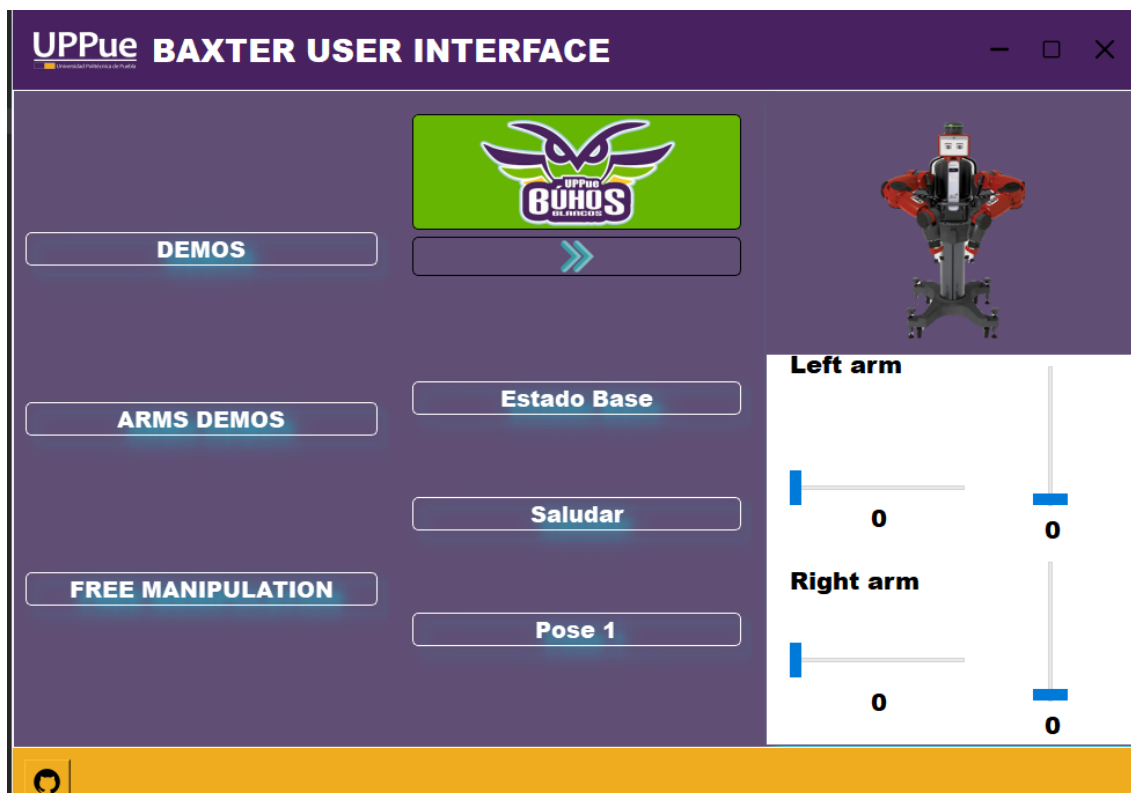


Figura 21 Button Búhos Blancos



Figura 22 Página oficial de la Universidad Politécnica de Puebla

En la parte superior se puede destacar el nombre asignado para la interfaz y del lado izquierdo el Logotipo oficial de la Universidad Politécnica de Puebla, que al dar clic sobre ella redireccionará a su página oficial de Facebook

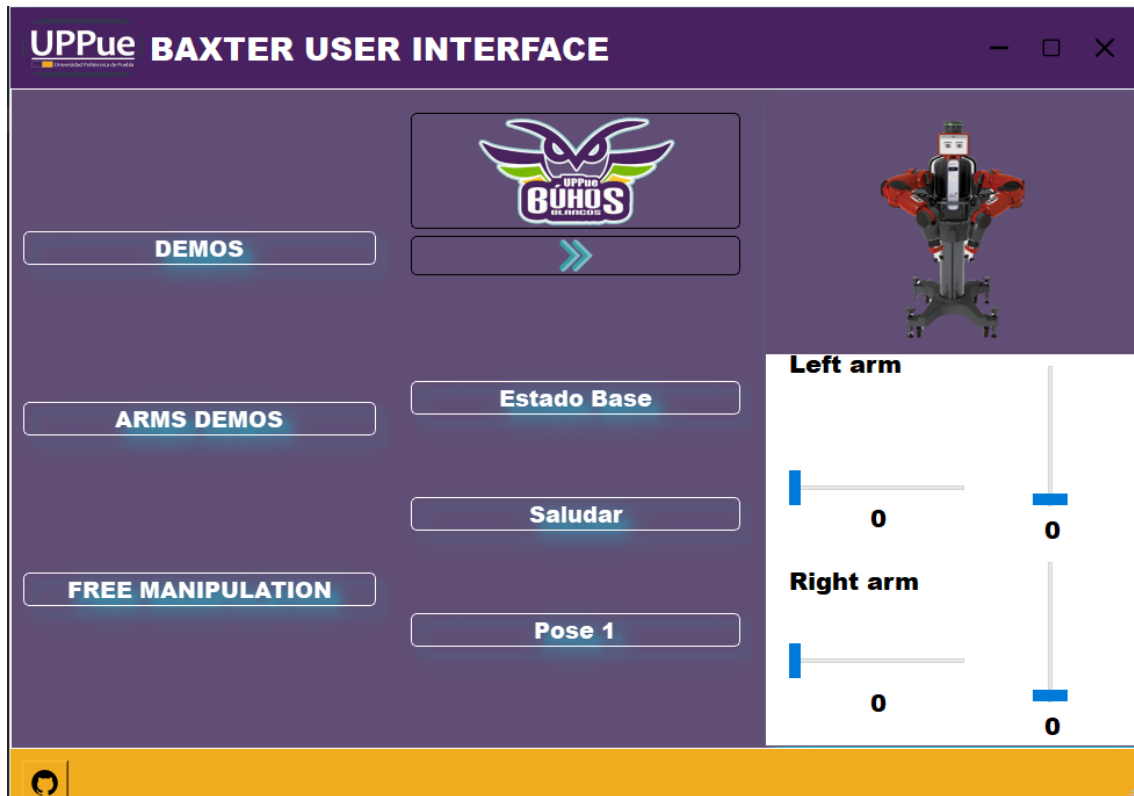


Figura 23 Button UPPUEBLA



Figura 24 Página oficial de Facebook de la Universidad Politécnica de Puebla

En la parte Inferior del lado izquierdo, se destaca el logo de Git Hub, en esta parte al dar clic sobre el logotipo redireccionará a un repositorio de git donde se podrá encontrar el código de este proyecto, además de otros proyectos realizados por el asesorado del proyecto.

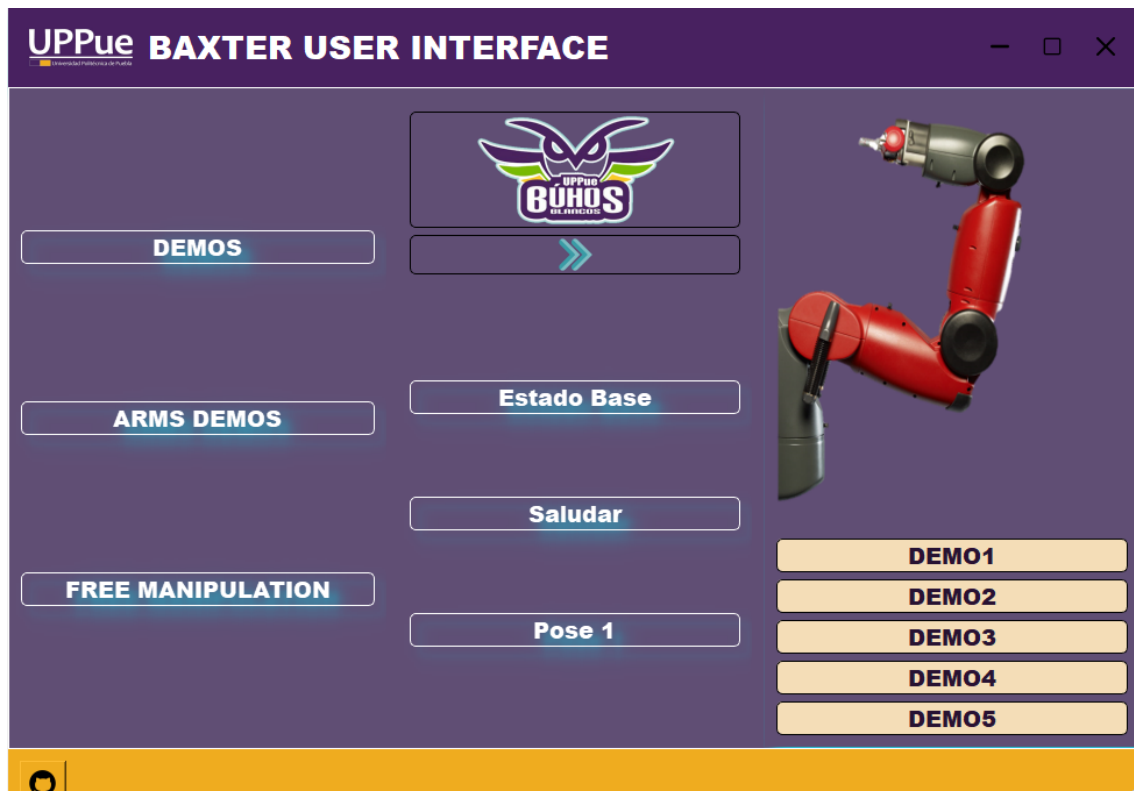


Figura 25 Button Git Hub

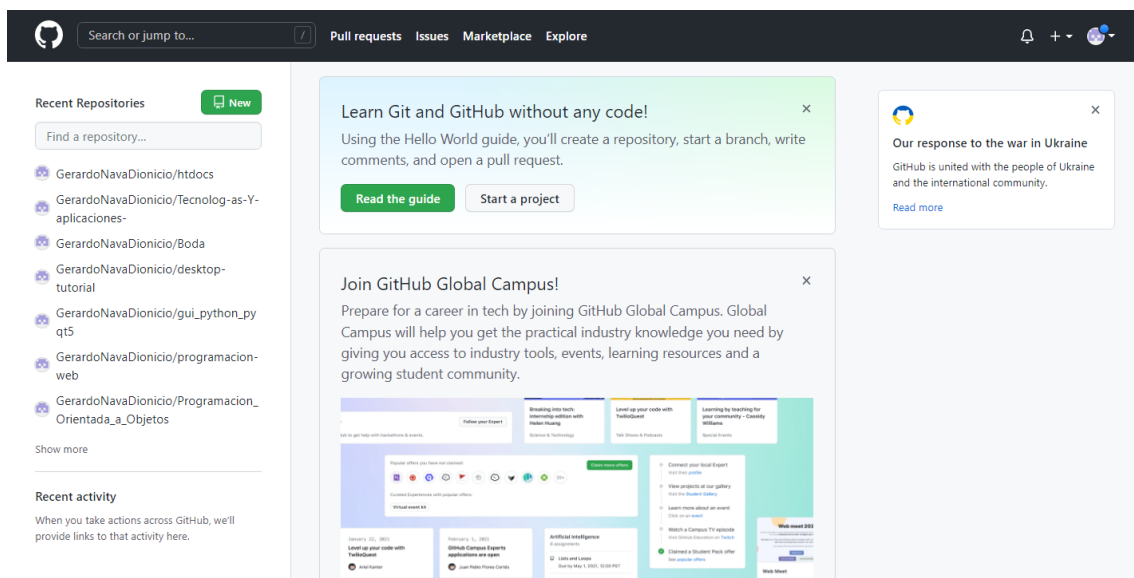


Figura 26 Página oficial del proyecto Baxter User Interface en un repositorio de Git Hub

Por último y no menos importante, en el centro de la interfaz, se encuentran un button con la imagen de una flecha roja, que, al dar clic sobre la misma, oculta el menú de funcionalidades del lado izquierdo

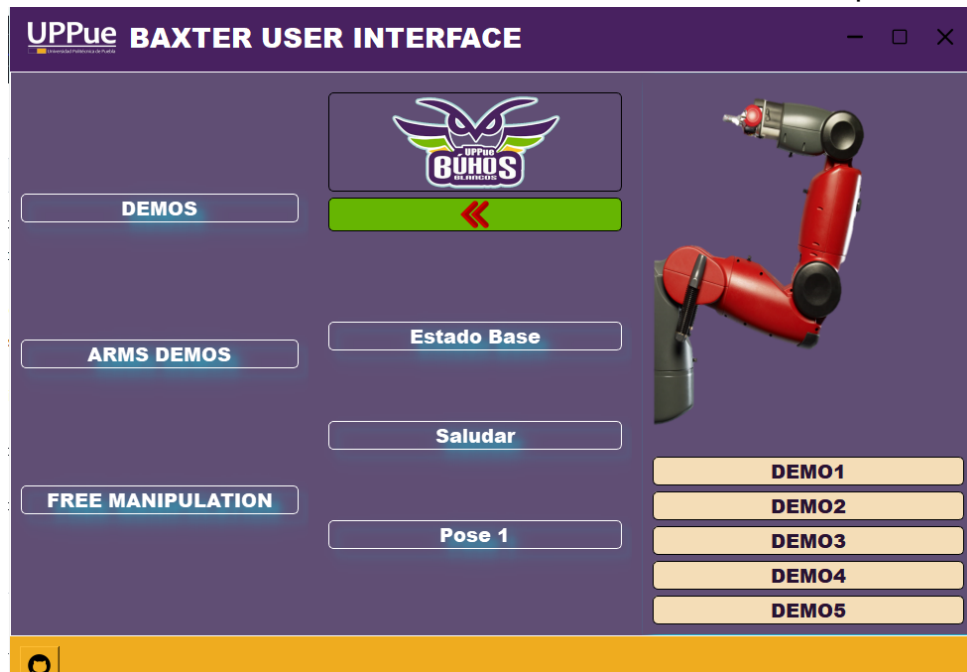


Figura 27 Button de Ocultar

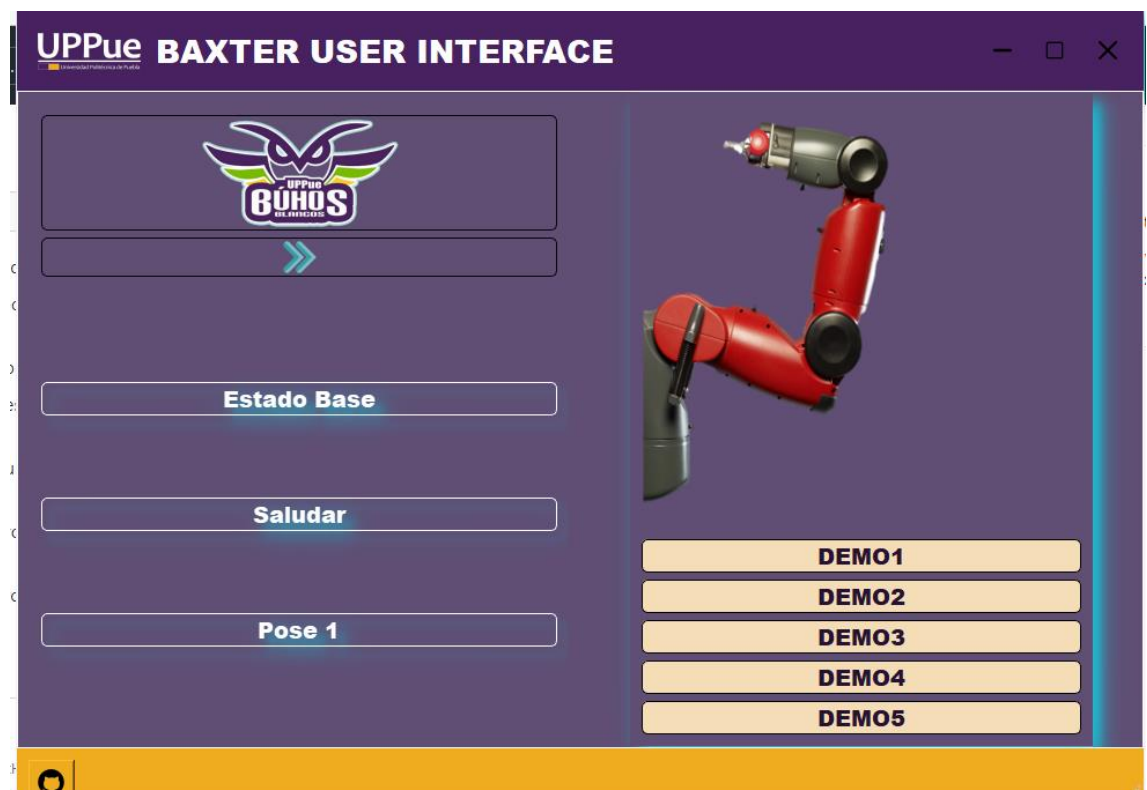


Figura 28 Button Mostrar

Una vez oculta mostrará una flecha azul que al darle clic hará que se muestre de nueva cuenta el menú de la izquierda.

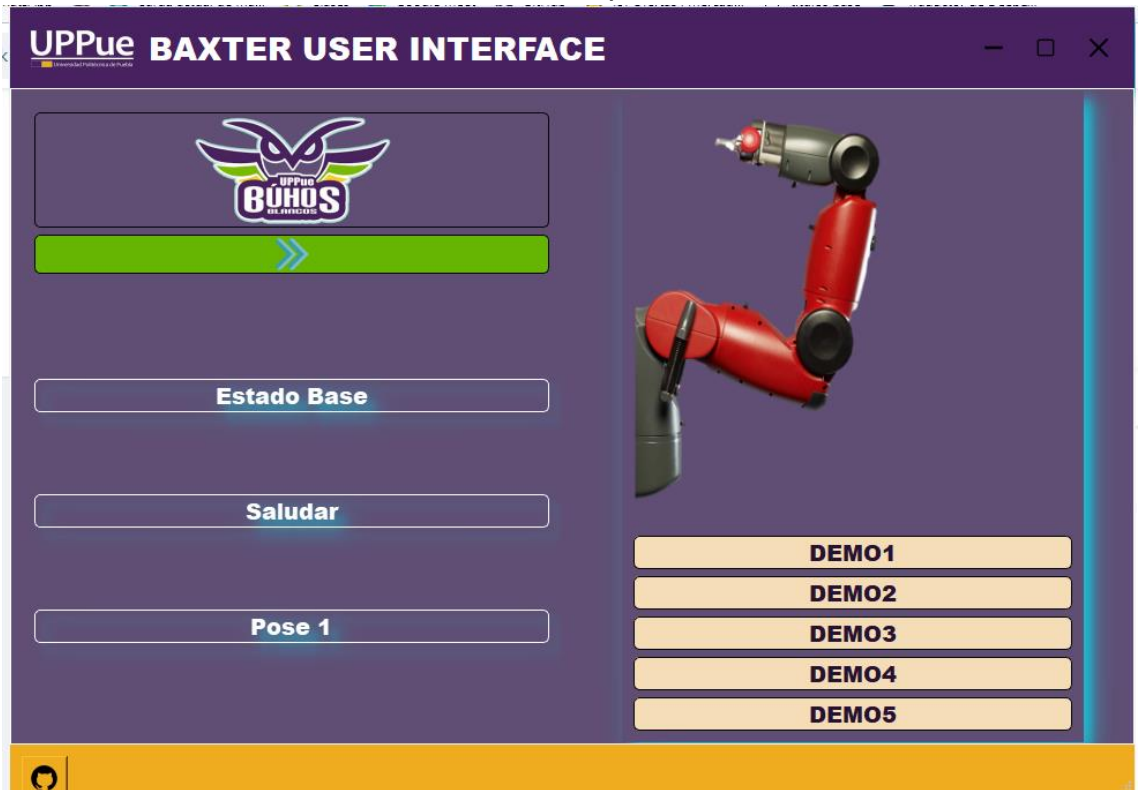


Figura 29 Button Mostrar 1

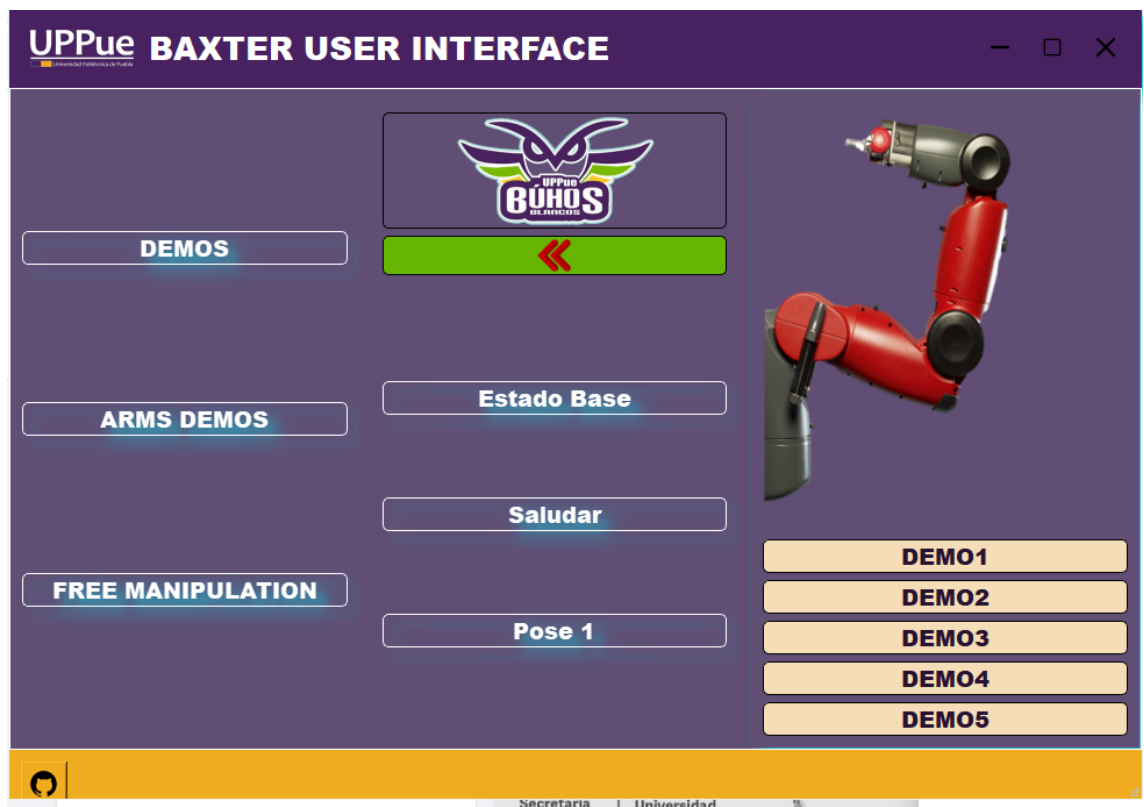


Figura 30 Button Mostrar 2

Para finalizar, en la parte inferior central, se muestran 3 buttons que ejecutarán funciones de movilidad de Baxter, entre ellas Saludar y estado base.

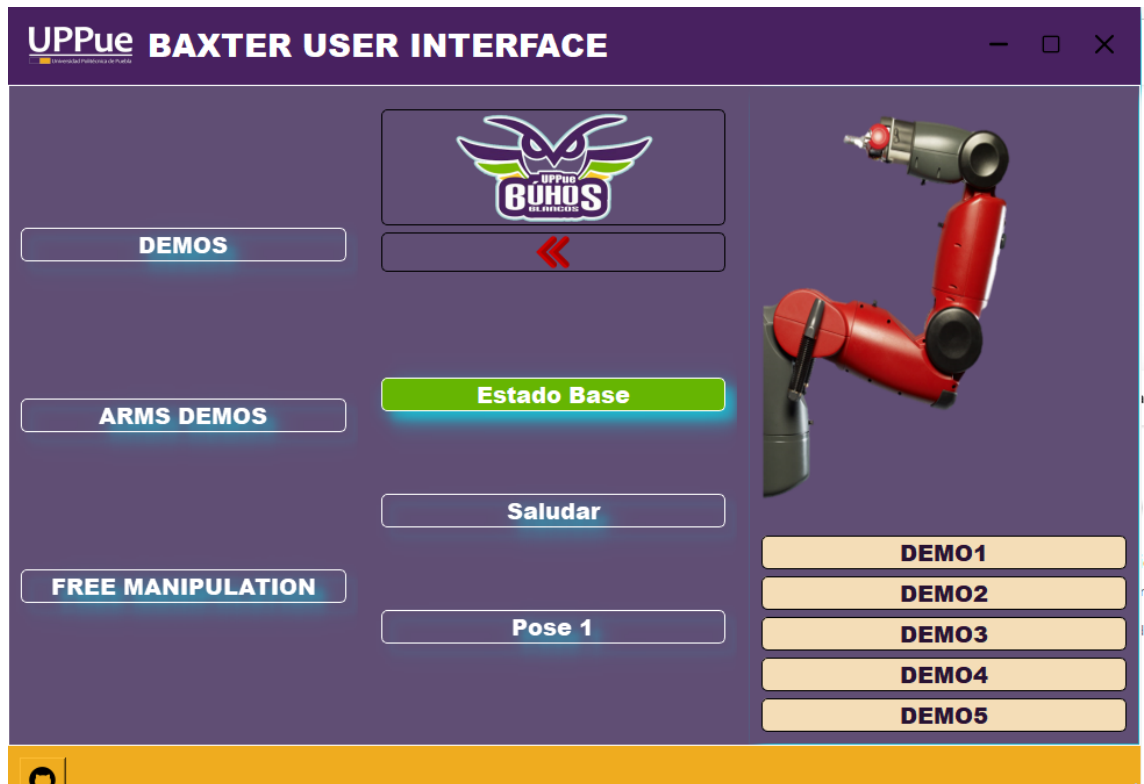


Figura 31 Button estado Base

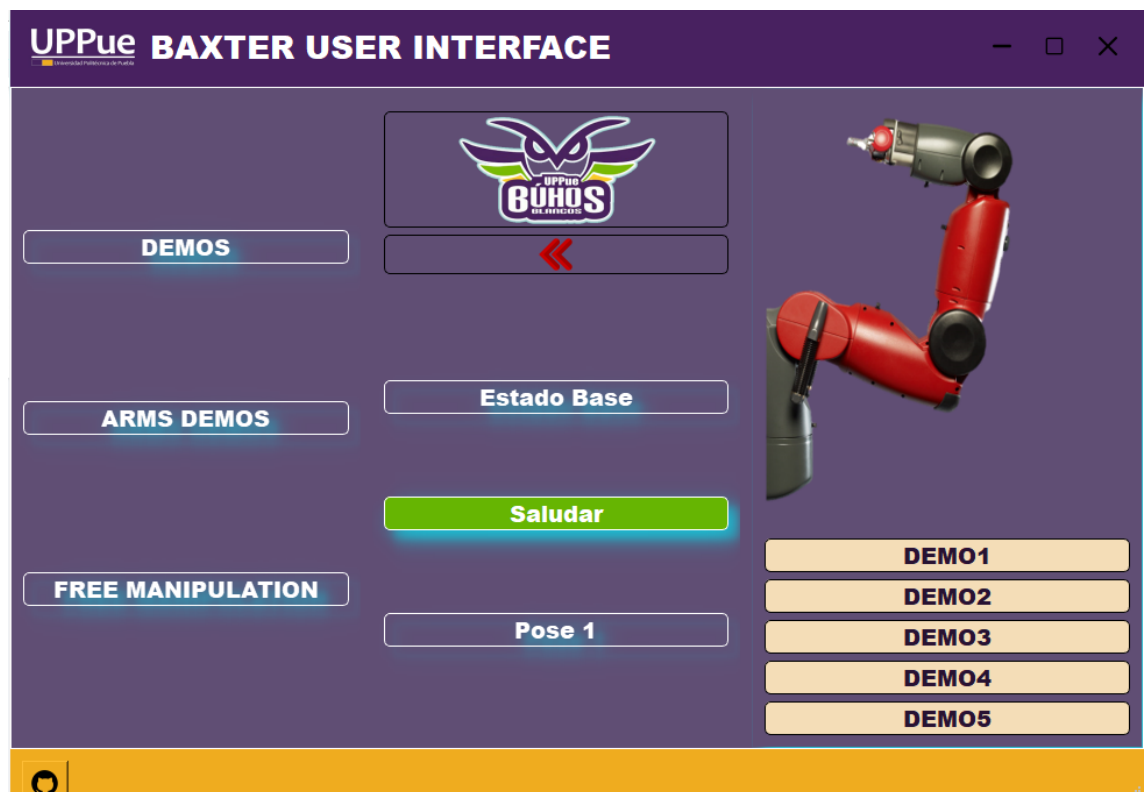


Figura 32 Button saludar

3.2 Encuestas de Validación de interfaz

Para esta parte, se diseñó un conjunto de preguntas las cuales se describen en el apéndice b, fueron aplicadas a alumnos del 5to cuatrimestre de la ingeniería en tecnologías de la información y las respuestas obtenidas son las siguientes.

1. ¿Qué tan difícil es leer los textos en pantalla?

5 respuestas

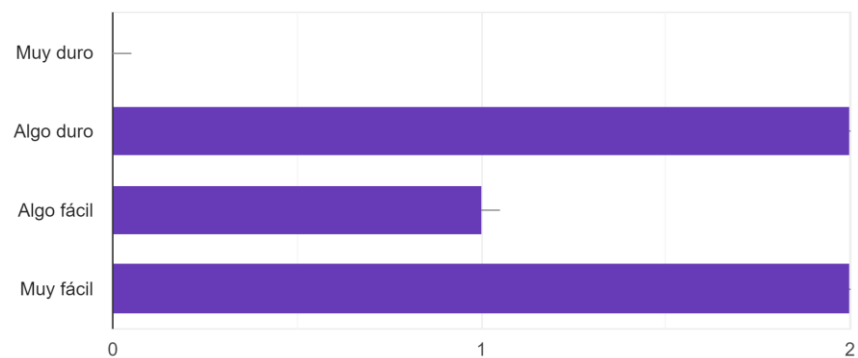


Figura 33 Pregunta 1 encuesta

2. ¿Cuál es tu Opinión sobre la organización de la información en pantalla?

5 respuestas

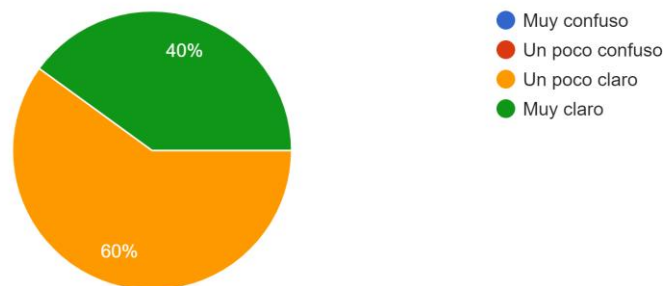


Figura 34 Pregunta 2 encuesta

3. ¿Los botones de las funcionalidades son claras?

5 respuestas

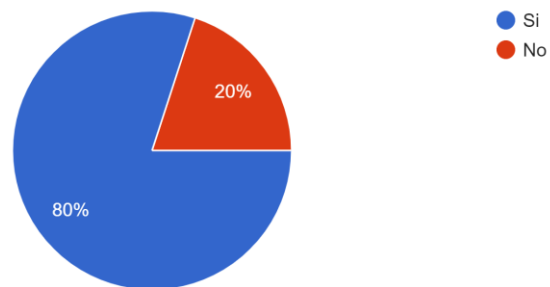


Figura 35 Pregunta 3 encuesta

5. ¿Los colores son Amigables?

5 respuestas

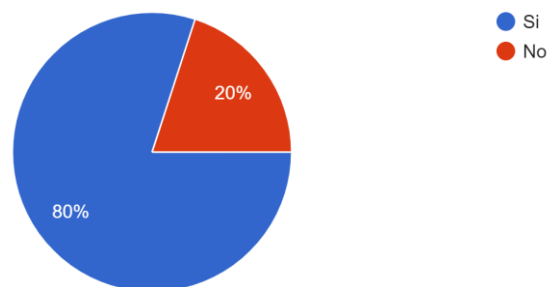


Figura 36 Pregunta 4 encuesta

6. ¿La tipografía te gusta?

5 respuestas

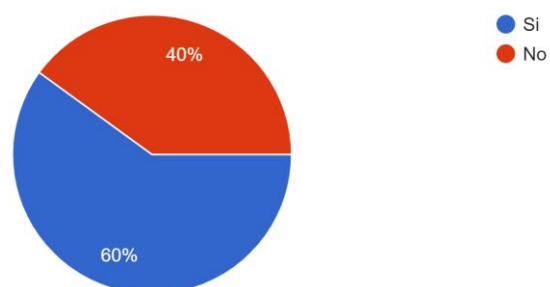


Figura 37 Pregunta 5 encuesta

7. ¿La interfaz te pareció intuitiva?

5 respuestas

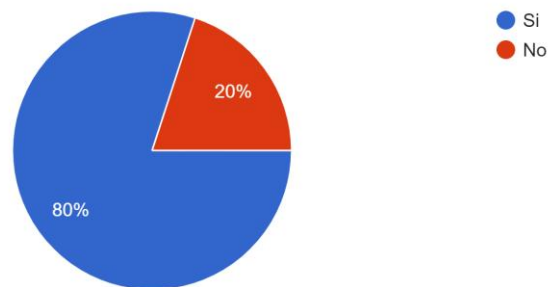


Figura 38 Pregunta 6 encuesta

8. ¿Los colores de la interfaz te parecen?

5 respuestas

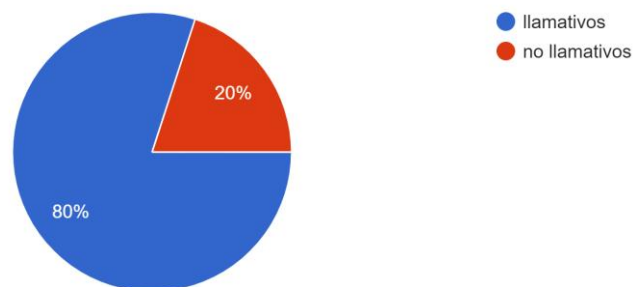


Figura 39 Pregunta 7 encuesta

9. ¿El tamaño de los botones son?

5 respuestas

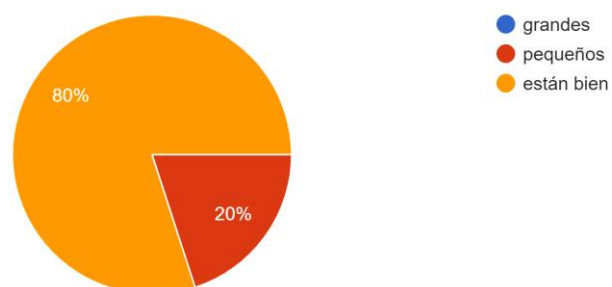


Figura 40 Pregunta 8 encuesta

10. ¿Te es funcional la distribución de los botones?

5 respuestas

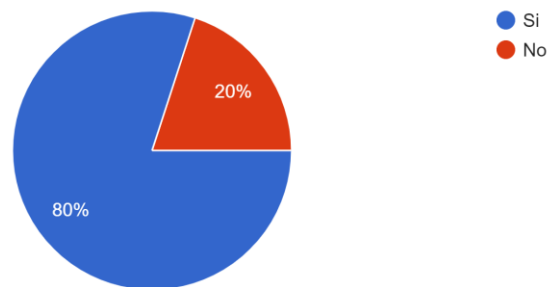


Figura 41 Pregunta 9 encuesta

¿Alguna Sugerencia respecto a la interfaz previamente mostrada?

5 respuestas

Considero que los textos de las funciones las deberías poner un poco más grandes, ya que, para personas con alguna discapacidad visual, no es muy apreciable.

Ya le hice mis comentarios

A mi parecer la interfaz esta muy bien.

Mejorar el diseño

Solo tiene que agregar más funcionalidad y que el diseño sea más atractivo.

Figura 42 Pregunta 10 encuesta

Como se puede observar en los resultados, más del 50% de las respuestas son positivas, lo que me da indicios de que mi interfaz es funcional para usuarios no expertos en el tema.

4. Conclusiones y recomendaciones

El resultado obtenido es el esperado, el desarrollo de una interfaz visual para rutinas de movilidad de Baxter.

Se implementó una interfaz visual de rutinas de movilidad para el robot Baxter que se encuentra en la universidad Politécnica de Puebla, bajo las tecnologías necesarias, como lenguaje de programación Python, para ser compatible con el framework ROS (Robot Operating System).

El desarrollo fue interesante, desde saber las necesidades del usuario, hasta la implementación del código de la interfaz en lenguaje de programación Python.

Durante el proceso del diseño de la interfaz se tuvo que buscar información y ejemplos de ciertos objetos, debido a que los conocimientos aprendidos no fueron suficientes y de esta manera me vi obligado a ser autodidacta, lo cual fue de mucha ayuda.

Como experiencia personal, fue de mucho agrado el hecho de haber podido diseñar algo así y me entusiasma que, en algún futuro no muy lejano, esta interfaz se usará para las rutinas de movilidad de Baxter, y cuando algún compañero de la institución se sienta sorprendido de las cosas las cuales somos capaces de lograr.

De esta manera se aporta esta interfaz visual a la Universidad Politécnica de Puebla en el área de laboratorio de robótica.

5. Referencias bibliográficas

- [1] W. C. School, «La guía del principiante: ¿Que es UX/UI?,» Wild Code School, 11 06 2021. [En línea]. Available: <https://www.wildcodeschool.com/es-ES/blog/que-es-ux-ui-diseno-interfaz-usuario-experiencia>. [Último acceso: 23 03 2022].
- [2] Python, «Python,» 2022 03 23. [En línea]. Available: <https://www.python.org/>.
- [3] R. KeepCoding, «Ventajas y Desventajas de Python,» Tech School, 22 01 2022. [En línea]. Available: <https://keepcoding.io/blog/ventajas-y-desventajas-de-python/>. [Último acceso: 23 03 2022].
- [4] Python, «Python Tutorial,» qt designer python, 2021. [En línea]. Available: <https://pythonbasics.org/qt-designer-python/>. [Último acceso: 23 03 2022].
- [5] A. Lima, «GUI DE PYTHON - PYQT VS TKINTER,» Acervo Lima, 2022. [En línea]. Available: <https://es.acervolima.com/gui-de-python-pyqt-vs-tkinter/>. [Último acceso: 23 03 2022].
- [6] U. P. d. Puebla, «Universidad Politécnica de Puebla,» Facebook, [En línea]. Available: <https://www.facebook.com/universidadpolitecnicadepuebla>. [Último acceso: 23 03 2022].
- [7] IMAGECOLORPICKER.com, «imagecolorpicker.com,» imagecolorpicker.com, [En línea]. Available: <https://imagecolorpicker.com/es>. [Último acceso: 23 03 2022].
- [8] Flaticon, «Flaticon,» Flaticon, [En línea]. Available: <https://www.flaticon.es/>. [Último acceso: 23 03 2022].
- [9] Freepng.es, «Robot móvil Baxter Móvil de manipulador de Clearpath Robótica - La robótica Imágen de Png,» Freepng.es, [En línea]. Available: <https://www.freepng.es/png-3prhu9/>. [Último acceso: 23 03 2022].
- [10] IEEE, «Baxter,» [En línea]. Available: <https://robots.ieee.org/robots/baxter/?gallery=photo4>. [Último acceso: 23 03 2022].
- [11] «programador clic,» programmerclick.com, [En línea]. Available: <https://programmerclick.com/article/24861793699/>. [Último acceso: 29 03 2022].

2022].

- [12] rakshitarora, «¿Cómo crear el widget Etiqueta en PyQt5?,» GeeksforGeeks, 26 03 2020. [En línea]. Available: <https://www.geeksforgeeks.org/how-to-create-label-widget-in-pyqt5/>. [Último acceso: 29 03 2022].
- [13] «QPushButton Clase,» Qt, [En línea]. Available: <https://doc.qt.io/qt-5/qpushbutton.html>. [Último acceso: 29 03 2022].
- [14] «Python Tutorial,» <https://pythonbasics.org> 2021, [En línea]. Available: <https://pythonbasics.org/pyqt-toolbox/>. [Último acceso: 29 03 2022].

6. Apéndice a - código de programación de la interfaz visual

```
1 import sys # Usar variables de interacción
2 from PyQt5.QtWidgets import QApplication, QMainWindow, QGraphicsDropShadowEffect, QMessageBox #asignar sombras
3 from PyQt5.QtCore import QPropertyAnimation, QEasingCurve # animaciones de menú y efectos de ocultar y mostrar
4 from PyQt5.QtGui import QColor #asignar colorees
5 from PyQt5 import QtCore, QtWidgets # Manipular widgets
6 from PyQt5.uic import loadUi #importar la interfaz
7 import webbrowser #Redireccionar a ciertas páginas
```

Figura 43 Librerías

```
1 class project(QMainWindow):
2     def __init__(self): #Constructor
3         super(project, self).__init__()
4         loadUi('interfaz/interfaz.ui',self) #se declara la interfaz
```

Figura 44 Creación de clase e importación de interfaz

```
1 self.PB_mostrar.clicked.connect(self.mover_menu)
2 self.PB_ocultar.clicked.connect(self.mover_menu)
3 self.PB_uppue.clicked.connect(self.paginauni)
4 self.git.clicked.connect(self.gith)
5 self.PB_PagUni.clicked.connect(self.universidad)
6 self.PB_demo1.clicked.connect(self.notificacion)
7 self.PB_demo2.clicked.connect(self.notificacion)
8 self.PB_demo3.clicked.connect(self.notificacion)
9 self.PB_demo4.clicked.connect(self.notificacion)
10 self.PB_demo5.clicked.connect(self.notificacion)
11 self.PB_AD1.clicked.connect(self.notificacion)
12 self.PB_AD2.clicked.connect(self.notificacion)
13 self.PB_AD3.clicked.connect(self.notificacion)
14 self.PB_AD4.clicked.connect(self.notificacion)
15 self.PB_AD5.clicked.connect(self.notificacion)
16 self.PB_f4.clicked.connect(self.notificacion)
17 self.PB_f5.clicked.connect(self.notificacion)
18 self.PB_f6.clicked.connect(self.notificacion)
```

Figura 45 Declarar Buttons y asignarlos a una función

```

1  #ocultar menú
2      self.maximizar.hide()
3      self.PB_ocultar.hide()
4      #sombra de los widgets
5      self.sombra_frame(self.stackedWidget)
6      self.sombra_frame(self.frame_superior)
7      self.sombra_frame(self.PB_f1)
8      self.sombra_frame(self.PB_f2)
9      self.sombra_frame(self.PB_f3)
10     self.sombra_frame(self.PB_f4)
11     self.sombra_frame(self.PB_f5)
12     self.sombra_frame(self.PB_f6)
13
14     #SizeGrip
15     self.gripSize=10
16     self.grip = QtWidgets.QSizeGrip(self)
17     self.grip.resize(self.gripSize, self.gripSize)
18
19     # mover ventana
20     self.frame_superior.mouseMoveEvent = self.mover_ventana

```

Figura 46 asignar funciones de sombra, mover menú y tamaño

```

1  #Barra de titulos
2      self.minimizar.clicked.connect(self.control_minimizar)
3      self.restaurar.clicked.connect(self.control_normal)
4      self.maximizar.clicked.connect(self.control_maximizar)
5      self.cerrar.clicked.connect(lambda: self.close())
6
7      #Eliminar barra y titulo - opacidad
8      self.setWindowFlag(QtCore.Qt.FramelessWindowHint)
9      self.setWindowOpacity(1)
10
11     #acceder a las paginas
12     self.PB_f1.clicked.connect(self.paginaUno)
13     self.PB_f2.clicked.connect(self.paginaDos)
14     self.PB_f3.clicked.connect(self.paginaTres)
15

```

Figura 47 asignar funciones de control de la para de menú

```

1  #Horizontal Slider
2      self.lef_arm_horizontal.setMinimum(0)
3      self.lef_arm_horizontal.setMaximum(100)
4      self.lef_arm_horizontal.setSingleStep(1)
5      self.lef_arm_horizontal.setValue(0)
6      self.lef_arm_horizontal.valueChanged.connect(self.getValueHorizontal)
7      self.lef_arm_vertical.setMinimum(0)
8      self.lef_arm_vertical.setMaximum(100)
9      self.lef_arm_vertical.setSingleStep(1)
10     self.lef_arm_vertical.setValue(0)
11     self.lef_arm_vertical.valueChanged.connect(self.getValueVertical)
12     #Hvertical Slide
13     self.right_arm_horizontal.setMinimum(0)
14     self.right_arm_horizontal.setMaximum(100)
15     self.right_arm_horizontal.setSingleStep(1)
16     self.right_arm_horizontal.setValue(0)
17     self.right_arm_horizontal.valueChanged.connect(self.getValueHorizontal)
18     self.right_arm_vertical.setMinimum(0)
19     self.right_arm_vertical.setMaximum(100)
20     self.right_arm_vertical.setSingleStep(1)
21     self.right_arm_vertical.setValue(0)
22     self.right_arm_vertical.valueChanged.connect(self.getValueVertical)
23

```

Figura 48 Sliders

```

1  #Esta función Obtiene los datos de los sliders y los muestra en una etiqueta
2      def getValueHorizontal(self):
3          value=self.lef_arm_horizontal.value()
4          self.lef_arm_horizontal_value.setText(str(value))
5
6          value=self.right_arm_horizontal.value()
7          self.right_arm_horizontal_value.setText(str(value))
8      #Esta función Obtiene los datos de los sliders y los muestra en una etiqueta
9      def getValueVertical(self):
10         value=self.lef_arm_vertical.value()
11         self.lef_arm_vertical_value.setText(str(value))
12
13         value=self.right_arm_vertical.value()
14         self.right_arm_vertical_value.setText(str(value))

```

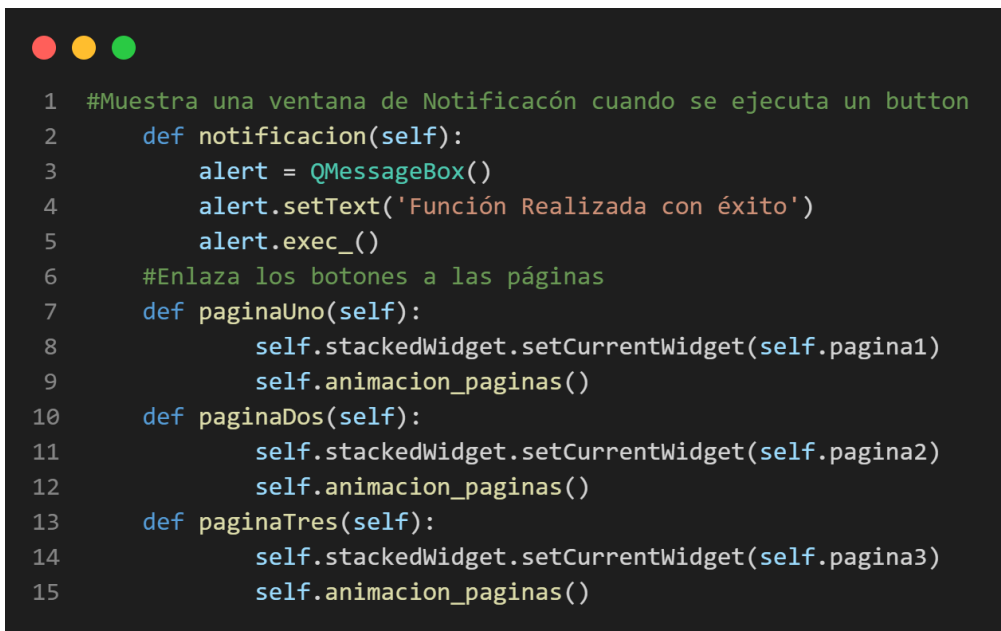
Figura 49 Obtener Datos de los Sliders

```

1  #Redirrecciona los botones a páginas de internet
2      def paginauni(self):
3          webbrowser.open('http://www.uppuebla.edu.mx/joomla1/')
4      def gith(self):
5          webbrowser.open('https://github.com/')
6      def universidad(self):
7          webbrowser.open('https://www.facebook.com/universidadpolitecnicedepuebla')

```

Figura 50 Redirección de páginas

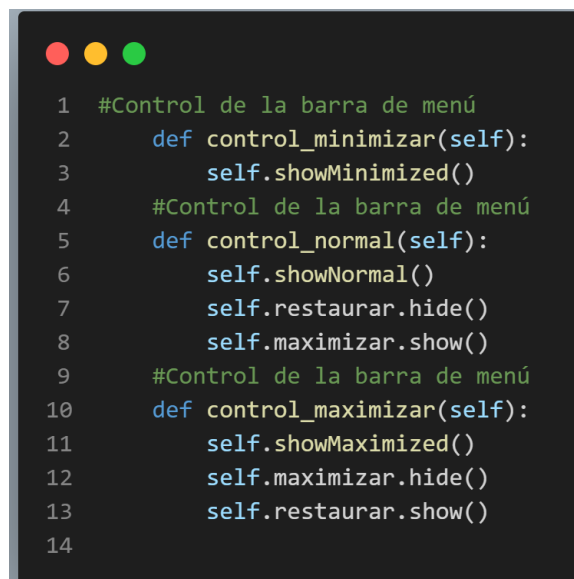


```

1  #Muestra una ventana de Notificación cuando se ejecuta un button
2      def notificacion(self):
3          alert = QMessageBox()
4          alert.setText('Función Realizada con éxito')
5          alert.exec_()
6  #Enlaza los botones a las páginas
7  def paginaUno(self):
8      self.stackedWidget.setCurrentWidget(self.pagina1)
9      self.animacion_paginas()
10 def paginaDos(self):
11     self.stackedWidget.setCurrentWidget(self.pagina2)
12     self.animacion_paginas()
13 def paginaTres(self):
14     self.stackedWidget.setCurrentWidget(self.pagina3)
15     self.animacion_paginas()

```

Figura 51 Función de Notificación y enlace de páginas a buttons



```

1  #Control de la barra de menú
2      def control_minimizar(self):
3          self.showMinimized()
4  #Control de la barra de menú
5  def control_normal(self):
6      self.showNormal()
7      self.restaurar.hide()
8      self.maximizar.show()
9  #Control de la barra de menú
10 def control_maximizar(self):
11     self.showMaximized()
12     self.maximizar.hide()
13     self.restaurar.show()
14

```

Figura 52 Control de la barra de menú

```

1  #función de animación de el menú lateral
2  def mover_menu(self):
3      if True:
4          width=self.menu_lateral.width()
5          normal = 0
6          if width == 0:
7              extender = 300
8              self.PB_mostrar.hide()
9              self.PB_ocultar.show()
10         else:
11             self.PB_mostrar.show()
12             self.PB_ocultar.hide()
13             extender=normal
14         self.animacion=QPropertyAnimation(self.menu_lateral,b"maximumWidth")
15         self.animacion.setStartValue(width)
16         self.animacion.setEndValue(extender)
17         self.animacion.setDuration(100)
18         self.animacion.setEasingCurve(QEasingCurve.OutInBack)
19         self.animacion.start()

```

Figura 53 Función de animación de las páginas y menú lateral izquierdo

```

1  #Función para poder mover libremente la interfaz
2  def mousePressEvent(self,event):
3      self.clickPosition=event.globalPos()
4
5  #función para controlar la barra de menú
6  def mover_ventana(self, event):
7      if self.isMaximized() == False:
8          if event.buttons() == QtCore.Qt.LeftButton:
9              self.move(self.pos()+event.globalPos() - self.clickPosition)
10             self.clickPosition = event.globalPos()
11             event.accept()
12         if event.globalPos().y()<=10:
13             self.showMaximized()
14             self.maximizar.hide ()
15             self.restaurar.show()
16         else:
17             self.showNormal()
18             self.restaurar.hide()
19             self.maximizar.show()

```

Figura 54 función para mover libre la interfaz y control de la barra de menú


```

1  #función para Asignar sombra a los objetos de la interfaz
2  def sombra_frame(self, frame):
3      sombra= QGraphicsDropShadowEffect(self)
4      sombra.setBlurRadius(30)
5      sombra.setXOffset(8)
6      sombra.setYOffset(8)
7      sombra.setColor(QColor(20, 200, 220, 255))#
8      frame.setGraphicsEffect(sombra)
9      ## SizeGrip
10 def resizeEvent (self, event):
11     rect=self.rect()
12     self.grip.move(rect.right() - self.gripSize,rect.bottom() - self.gripSize)
13     ## mover ventana
14 def mousePressEvent (self, event):
15     self.clickPosition = event.globalPos()

```

Figura 55 Sombra, tamaño y movilidad

```

1  #funcion para animar el menú de las páginas
2  def animacion_paginas(self):
3      if True:
4          width = self.stackedWidget.width()
5          x1=self.frame_paginas.rect().right()
6          normal=100
7      if width == 100:
8          extender = x1
9      else:
10         extender=normal
11     self.animacion1=QPropertyAnimation(self.stackedWidget,b"maximumbMidth ")
12     self.animacion1.setStartValue(width)
13     self.animacion1.setEndValue(extender)
14     self.animacion1.setDuration(500)
15     self.animacion1.setEasingCurve(QEasingCurve.InOutQuad) #OutInCubic
16     self.animacion1.start()

```

Figura 56 Animación de las páginas

```

1  if __name__ == '__main__':
2      app = QApplication([])
3      ventana = project()
4      ventana.show()
5      sys.exit(app.exec())

```

Figura 57 ejecución de la clase

6. Apéndice b – Desarrollo de cuestionario de evaluación

Para esta parte, se diseñó un conjunto de pregunta para poder valuar la interfaz visual y obtener retroalimentación de los compañeros de 5to cuatrimestre de la ITI., Las preguntas son:

- ¿Qué tan difícil es leer los textos en pantalla?
- ¿Cuál es tu Opinión sobre la organización de la información en pantalla?
- ¿Los botones de las funcionalidades son claras?
- ¿Los colores son Amigables?
- ¿La tipografía te gusta?
- ¿La interfaz te pareció intuitiva?
- ¿Los colores de la interfaz te parecen?
- ¿El tamaño de los botones son?
- ¿Te es funcional la distribución de los botones?
- ¿Alguna Sugerencia respecto a la interfaz previamente mostrada?



Universidad Politécnica de Puebla
Ingeniería en Informática

Gerardo Nava Dionicio
Dr. Antonio Benítez Ruiz
Dr. Antonio Benítez Ruiz

Este documento se distribuye para los términos de la
Licencia 2.5 Creative Commons (CC-BC-NC-ND 2.5 MX)