TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CUAUTLA

INGENIERÍA EN SISTEMAS

TOPICOS AVANZADOS DE PROGRAMACIÓN

SEMESTRE **ENERO JUNIO-2026**

DOCENTE: CABALLERO ALFARO ARÍSTIDES

UNIDAD 1

MARTÍNEZ GALLARDO SANTIAGO

4TO SEMESTRE

GRUPO: 1

25 DE FEBRERO DEL 2026

El presente proyecto consiste en el desarrollo de una aplicación de chat interactiva utilizando el lenguaje de programación Python junto con la librería Flet, la cual permite crear interfaces gráficas modernas mediante una estructura basada en controles, eventos y comunicación en tiempo real. Desde el inicio del código se observa una organización clara y estructurada, comenzando con la importación del decorador dataclass desde el módulo dataclasses, herramienta que facilita la creación de clases destinadas únicamente a almacenar datos sin necesidad de definir manualmente métodos como el constructor. Posteriormente se importa Flet con el alias ft, que será la base para todos los componentes visuales y funcionales del programa.

Se define la clase Message utilizando @dataclass, lo cual permite representar cada mensaje del chat como un objeto estructurado con tres atributos fundamentales: user_name, que almacena el nombre del usuario que envía el mensaje; text, que contiene el contenido escrito; y message_type, que identifica si el mensaje corresponde a un mensaje normal de conversación (“chat_message”) o a un mensaje informativo de ingreso al sistema (“login_message”). Esta estructura facilita la organización de la información y permite diferenciar visualmente los tipos de mensajes dentro de la aplicación.

Posteriormente se crea la clase ChatMessage, que hereda de ft.Row y está decorada con @ft.control, lo cual indica que se trata de un componente personalizado de Flet. Esta clase se encarga de definir cómo se mostrará cada mensaje en pantalla. En su constructor recibe un objeto de tipo Message y configura los elementos visuales que lo componen. Se establece una alineación vertical al inicio para que los elementos queden correctamente organizados. Dentro de los controles se incluye un CircleAvatar, que muestra la inicial del usuario dentro de un círculo de color, lo cual mejora la identificación visual en el chat. La inicial se obtiene mediante el método get_initials, el cual toma la primera letra del nombre del usuario y la convierte en mayúscula; si el nombre no existe, devuelve un valor por defecto. El color del avatar se determina mediante el método get_avatar_color, que selecciona un color de una lista predefinida utilizando el resultado de la función hash aplicada al nombre del usuario, lo que garantiza que cada usuario mantenga siempre el mismo color durante la sesión.

La función principal main recibe como parámetro el objeto page, que representa la ventana principal de la aplicación. Dentro de esta función se configuran propiedades generales como la alineación horizontal y el título de la ventana. Posteriormente se definen varias funciones internas que controlan la lógica del programa. La función join_chat_click se ejecuta cuando el usuario intenta ingresar al chat; esta valida que

el campo de texto donde se escribe el nombre no esté vacío. Si lo está, muestra un mensaje de error. En caso contrario, almacena el nombre en la sesión mediante page.session.store.set, cierra el cuadro de diálogo de bienvenida y envía un mensaje global utilizando el sistema pubsub de Flet, indicando que el usuario se ha unido al chat. El sistema pubsub es fundamental, ya que permite enviar mensajes a todos los usuarios conectados en tiempo real.

Se define también la función asíncrona send_message_click, que se activa cuando el usuario presiona Enter o el botón de enviar. Esta función verifica que el campo de mensaje no esté vacío; si contiene texto, crea un nuevo objeto Message con el nombre almacenado en la sesión y el contenido escrito, enviándolo a todos los usuarios mediante page.pubsub.send_all. Posteriormente limpia el campo de texto y vuelve a colocar el cursor en él para permitir continuar escribiendo sin interrupciones. El uso de programación asíncrona permite que la interfaz se mantenga fluida y receptiva.

La función on_message es la encargada de recibir y mostrar los mensajes que llegan a través del sistema pubsub. Si el mensaje recibido es de tipo “chat_message”, se crea una instancia de ChatMessage para mostrarlo con avatar, nombre en negrita y texto seleccionable. Si el mensaje es de tipo “login_message”, se muestra como un texto en cursiva, con tamaño reducido y color más tenue, diferenciándolo visualmente de los mensajes normales. Cada mensaje se agrega a un componente ListView, el cual está configurado con desplazamiento automático (auto_scroll=True) para que siempre se visualice el mensaje más reciente.

En la construcción de la interfaz se incluye un cuadro de diálogo modal (AlertDialog) que solicita al usuario ingresar su nombre antes de poder interactuar en el chat. Este diálogo contiene un campo de texto con enfoque automático y un botón para confirmar el ingreso. Además, se crea un campo de entrada para nuevos mensajes (TextField) que permite múltiples líneas y admite combinaciones como Shift+Enter para saltos de línea. Finalmente, todos los componentes se agregan a la página dentro de un contenedor con borde y esquinas redondeadas, seguido de una fila que contiene el campo de mensaje y un botón con ícono de enviar.

El programa concluye con la instrucción ft.run(main), que ejecuta la aplicación llamando a la función principal. En términos generales, este proyecto aplica conceptos de programación orientada a eventos, manejo de sesiones, diseño de componentes personalizados, comunicación en tiempo real mediante pubsub y uso de programación asíncrona. La aplicación permite que múltiples usuarios interactúen dinámicamente, mostrando mensajes organizados visualmente con identificadores

personalizados, logrando así un sistema de chat funcional, estructurado y visualmente atractivo.