Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Vinculando una Aplicación móvil con Xamarin Forms y ChatGPT

Linking a Mobile Application with Xamarin Forms and ChatGPT

Fecha de recepción: 2023-06-02 • Fecha de aceptación: 2023-07-25 • Fecha de publicación: 2023-08-05

Daniel Molina H. [1]
[1] Universidad Tecnológica Israel, Ecuador
E9017963126@uisrael.edu.ec

Resumen

En este ensayo, se presenta el desarrollo de una aplicación móvil de chat utilizando Xamarin Forms y ChatGPT. La aplicación permite a los usuarios realizar preguntas y obtener respuestas generadas por el modelo de lenguaje de ChatGPT. Se describe la implementación de las vistas, controladores, interfaces y clases necesarias, así como el uso de la API de ChatGPT para generar respuestas. Además, se muestra cómo almacenar las preguntas y respuestas en una base de datos local SQLITE. El trabajo también aborda las consideraciones de diseño y las mejores prácticas para desarrollar aplicaciones de chat en Xamarin Forms.

Palabras clave. Aplicación móvil, Xamarin Forms, ChatGpt, SQLITE

Abstract

This article presents the development of a chat application using Xamarin Forms and ChatGPT. The application allows users to ask questions and receive responses generated by the ChatGPT language model. The implementation of the necessary views, controllers, interfaces, and classes is described, along with the usage of the ChatGPT API to generate responses. Furthermore, the article demonstrates how to store the questions and responses in a local database. Design considerations and best practices for developing chat applications in Xamarin Forms are also discussed.

Keywords. Mobile Application, Xamarin Forms, ChatGpt, SQLITE

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Introducción

La comunicación en tiempo real a través de aplicaciones de chat se ha vuelto cada vez más popular en diversos ámbitos, como el servicio al cliente, la asistencia técnica, la educación y la interacción con chatbots. Desde finales del 2022 esta herramienta a tomado mucha fuerza en el ámbito de desatollo de software, en el trabajo [1], se proporciona una descripción general de la interfaz de programación de aplicaciones (API) de ChatGPT, aborda cómo esta API se puede utilizar de manera efectiva en el campo del desarrollo de software, destacando sus aplicaciones y ventajas clave.

En los últimos años, el desarrollo de aplicaciones móviles ha crecido notablemente, convirtiéndose en una herramienta primordial para la vida de las personas, se han desarrollado aplicaciones móviles relacionadas en la salud [2], en aplicaciones gubernamentales para la gestión del tránsito y transporte [3], concientización de usuarios en aspectos cotidianos como el consumo energético [4], entre otros campos importantes.

En este contexto, Xamarin Forms [5], un framework de desarrollo multiplataforma, ofrece una solución eficiente para crear aplicaciones de chat compatibles con iOS, Android y otros dispositivos [6].

A lo largo de este documento, abordaremos los desafíos y oportunidades que surgen al vincular una aplicación móvil con ChatGPT, destacando las ventajas de utilizar Xamarin Forms como plataforma de desarrollo. Además, proporcionaremos ejemplos prácticos y casos de uso para ilustrar cómo esta integración puede potenciar la funcionalidad y la interactividad de las aplicaciones móviles, abriendo la puerta a una nueva dimensión de experiencias de usuario personalizadas y conversacionales.

Este ensayo sirve como guía completa para desarrolladores y entusiastas que buscan aprovechar las capacidades de ChatGPT en sus aplicaciones móviles, ofreciendo un enfoque paso a paso, desde la configuración inicial hasta la implementación práctica. Al final, esperamos inspirar y capacitar a la comunidad de desarrolladores a explorar nuevas fronteras en la interacción humano-máquina a través de la combinación de Xamarin Forms y ChatGPT.

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Cuerpo

2.1. Xamarin Forms

Es un subconjunto de Xamarin que permite la creación de interfaces de usuario compartidas a través de XAML, se pueden diseñar interfaces de usuario única que se renderiza de manera nativa en cada plataforma [7].

La estructura de proyectos en Xamarin puede variar según las preferencias del equipo de desarrollo y la complejidad del proyecto, para este ejemplo se utiliza la siguiente estructura.

Proyecto Principal (Solution):

- Nombre de la solución: Nombre de la solución
- Proyecto compartido (Shared Project): Contiene el código compartido entre las plataformas (Xamarin.Forms)
- Proyecto de Android: Contiene el código específico de Android.
- Proyecto de iOS: Contiene el código específico de iOS.

2.2. Vincular ChatGPT

La ventana de ChatGPT es donde los usuarios pueden realizar consultas a la API de ChatGPT y obtener respuestas. A continuación, se muestra el código del archivo ChatGPT.xaml:

Para lograr la comunicación con ChatGPT es necesario crear una llave API y esto lo podemos lograr de la siguiente forma:

2.2.1. Crear una llave API

Para lograr la comunicación con ChatGPT es necesario crear una llave API y esto lo podemos lograr de la siguiente forma:

1. Crear una cuenta en ChatGPT: Se debe ingresar al sitio web oficial de OpenAI dónde con un correo electrónico válido se puede registrar la cuenta para poder acceder a los servicios.
2. Una vez creada la cuenta en OpenAI se debe ingresar a la dirección: https://platform.openai.com/overview dónde en la esquina superior derecha en el área personal se va a ingresar la opción “View API Key”, una vez dentro de esta opción se debe dar clic a el botón “Create new secret key” donde se va a mostrar la llave API que se deberá guardar en un lugar conocido de la PC ya que esta clave solo se muestra una vez.

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

2.2.2. Crear las clases con los métodos para realizar las consultas a ChatGPT

Para crear el cuerpo de la solicitud vamos a revisar los modelos propuestos en el sitio web oficial de OpenAI en la sección de chat: https://platform.openai.com/docs/api-reference/chat/create.

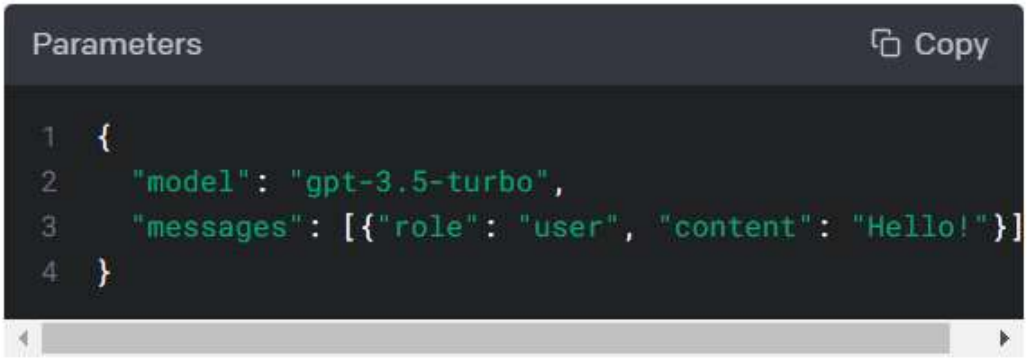
2.2.3. Crear la clase Request

Para crear el cuerpo de la pregunta que vamos a realizar a ChatGPT creamos una clase llamada Request en nuestro proyecto principal, dónde se escribirá el código de las preguntas. Se utiliza el modelo que se muestra en API-References copiando directamente el código y luego se lo pegándolo como una clase Json:

1. Copiar el código

Figura 1

Modelo de API-Reference



2. Pegar como Json class en nuestra clase Request

2.2.4. Crear la clase Response

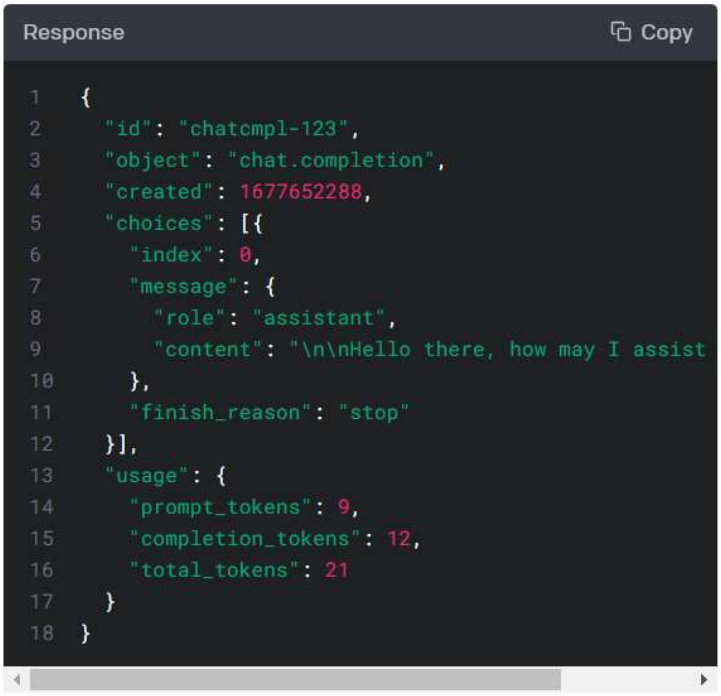
Creamos una clase y la nombramos como Response. Luego de igual forma que con la clase Request, copiamos el código del modelo mostrado en API-Reference y lo pegamos como una clase Json:

1. Copiar el código

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Figura 2

Modelo de API-Reference - Response



2. Pegar como Json class en nuestra clase Response

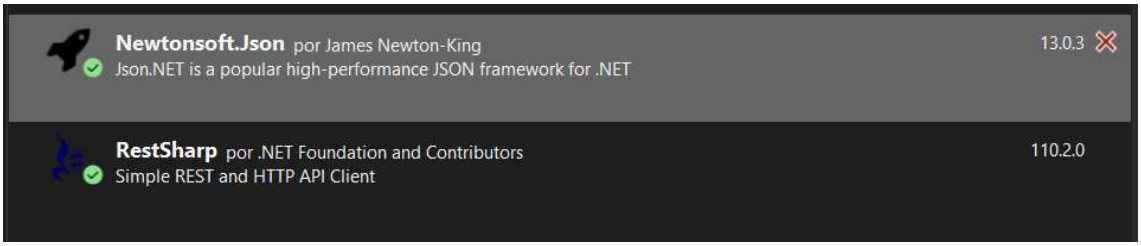
2.2.5. Crear el cuerpo de la consulta y respuesta a ChatGPT en el controlador de la ventana principal.

Como es conocido anteriormente se ha creado la ventana principal donde se van a realizar las consultas a la API de ChatGPT. Dentro de esta vista tenemos la implementación del botón "Enviar" donde vamos a realizar las consultas a través de POST. Para realizarlo debemos instalar la librería RestSharp dando clic derecho en la solución del proyecto y en el administrador de paquetes NuGet agregamos las librerías RestSharp y Newtonsoft.Json como se muestra a continuación:

Figura 3

Paquete Nuget - JSON

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50



Una vez instalada las librerías necesarias vamos a escribir los siguientes códigos para realizar la conexión con ChatGPT:

1. Declarar 3 variables como se muestra a continuación:

Figura 4

Variables

```
private const string EndPoint = "https://api.openai.com/";
private const string URI = "v1/chat/completions";
private const string APIKey = "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";
```

Como se puede observar las variables Enpoint y URI contienen la URL de la API de ChatGPT la cual utilizaremos más adelante. La variable APIKey contiene el API key que conseguimos al registrarnos en OpenAI y que previamente se guardó.

2. **var cliente = new RestClient(EndPoint);**

Aquí se crea una instancia del cliente de RestClient, que es una biblioteca RestSharp utilizada para realizar solicitudes HTTP a una API. La variable EndPoint es la URL que indica el punto final de la API de ChatGPT a la que se enviará la solicitud.

3. **var solicitud = new RestRequest(URI, Method.Post);**

Se crea una instancia de la clase RestRequest que representa una solicitud HTTP. El argumento URI es el recurso específico al que se enviará la solicitud (como una ruta en la API). Method.Post indica que se realizará una solicitud HTTP POST.

4. **solicitud.AddHeader("Content-Type", "application/json");**

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Se agrega un encabezado a la solicitud para indicar que el contenido que se enviará será de tipo JSON.

5. **solicitud.AddHeader("Authorization", "Bearer " + APIKey);**

Se agrega un encabezado de autorización a la solicitud. APIKey la variable que contiene la clave de autenticación necesaria para acceder a la API de ChatGPT.

6. **var cuerpo = new { model = "gpt-3.5-turbo", messages = new List<object> {new {role = "user", content = pregunta} } };**

Se crea un objeto anónimo llamado cuerpo que representa el cuerpo de la solicitud. Este objeto contiene el modelo de ChatGPT a utilizar que es tomado de la página de API-Reference, y una lista de mensajes, donde cada mensaje tiene un rol ("user" en este caso) y el contenido de la pregunta.

7. **var jsonString = JsonConvert.SerializeObject(cuerpo);**

El objeto cuerpo se serializa a una cadena de caracteres JSON utilizando la biblioteca Newtonsoft.Json (también conocida como Json.NET). Esto convierte el objeto en un formato que puede ser enviado a la API.

8. **solicitud.AddJsonBody(jsonString);**

Se agrega el cuerpo de la solicitud JSON al objeto solicitud utilizando el método AddJsonBody(). Esto envía el contenido de la pregunta al servidor de ChatGPT.

9. **var respuesta = cliente.Post<Response>(solicitud);**

Se realiza una solicitud POST al servidor de ChatGPT utilizando el cliente “cliente” y el objeto “solicitud”. La respuesta del servidor se guarda en la variable respuesta, que se espera que sea de tipo Response (clase creada previamente).

10. **string contenidoRespuesta = respuesta.choices[0].message.content;**

Se obtiene el contenido de la respuesta enviada por el servidor. Parece que la respuesta se encuentra en la propiedad choices del objeto respuesta, y se accede al primer elemento de la lista. Luego, se extrae el contenido del mensaje de la respuesta y se guarda en la variable contenidoRespuesta.

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

11. **LabelRespuesta.Text = contenidoRespuesta;**

El contenido de la respuesta obtenida se asigna a una etiqueta de texto para mostrarlo al usuario.

De esta forma podemos escribir nuestro código para que la consumir la API de ChatGPT y obtener una respuesta que luego guardaremos en una base de datos local SQLite.

Después de implementar la aplicación móvil utilizando Xamarin.Forms y la API de ChatGPT, se obtuvieron los siguientes resultados:

- Ventana de Login: Esta ventana permite al usuario ingresar su nombre de usuario y contraseña. Si las credenciales son correctas, el usuario es redirigido a la ventana de ChatGPT. De lo contrario, se muestra una alerta de usuario incorrecto.
- Ventana de ChatGPT: En esta ventana, el usuario puede ingresar preguntas y obtener respuestas generadas por la API de ChatGPT. La pregunta se envía a la API utilizando el formato de código #C, y la respuesta generada se muestra en la interfaz de usuario. Además, cada pregunta y respuesta se almacenan en una base de datos local.
- Ventana de ConsultaPregyResp: Esta ventana permite al usuario consultar los registros almacenados en la base de datos local. Los registros se muestran en una lista, lo que permite al usuario revisar las preguntas y respuestas generadas anteriormente.

La integración de la API de ChatGPT permite generar respuestas coherentes y relevantes a las preguntas de los usuarios. La capacidad de almacenar los registros en una base de datos local brinda la posibilidad de realizar análisis posteriores y obtener información valiosa sobre las interacciones de los usuarios.

En general, la aplicación móvil desarrollada muestra una implementación exitosa de la interacción con la API de ChatGPT y el uso de una base de datos local para almacenar los registros. Los resultados obtenidos demuestran la viabilidad de utilizar Xamarin.Forms como marco de desarrollo para crear aplicaciones móviles interactivas y escalables.

Estos resultados muestran el potencial de combinar tecnologías modernas para crear aplicaciones móviles más inteligentes y eficientes.

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Conclusiones

En este artículo, hemos desarrollado una aplicación móvil multiplataforma utilizando Xamarin.Forms para interactuar con la API de ChatGPT. Hemos implementado las ventanas de Login, ChatGPT y ConsultaPregyResp, así como la lógica para consumir la API y almacenar los registros en una base de datos local.

Xamarin.Forms nos permite desarrollar aplicaciones móviles con una sola base de código, lo que simplifica el proceso de desarrollo y reduce los esfuerzos requeridos para mantener múltiples versiones de la aplicación para diferentes plataformas.

La integración de la API de ChatGPT nos permite aprovechar el poder del modelo de lenguaje GPT-3.5 para generar respuestas basadas en preguntas de los usuarios. Esto puede ser utilizado en una amplia gama de aplicaciones que requieren interacciones de lenguaje natural.

La implementación de una base de datos local nos permite almacenar y consultar los registros generados por los usuarios, lo que proporciona una funcionalidad adicional y la capacidad de realizar análisis sobre las interacciones con la API.

En resumen, hemos explorado cómo combinar Xamarin.Forms, la API de ChatGPT y una base de datos local para crear una aplicación móvil interactiva y escalable. Esto es solo el comienzo, y puedes expandir y mejorar esta aplicación según tus necesidades y requisitos específicos.

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50

Referencias

[1] C. M. Gallardo Paredes, C. Machuca, and Y. M. Semblantes Claudio, "ChatGPT API: Brief overview and integration in Software Development," *International Journal of Engineering Insights*, vol. 1, no. 1, pp. 25–29, 2023.

[2] D. Orbes, J. Guevara, P. F. Baldeón Egas, and R. M. Toasa, "Mobile App as an Alternative in the Process of Speech Therapy in Children with Cerebral Palsy," *Lecture Notes in Networks and Systems*, vol. 511 LNNS, pp. 479–489, 2022, doi: 10.1007/978-3-031-11438-0_38/COVER.

[3] D. Corral, R. M. Toasa, Y. Semblantes, and L. F. Aguas, "Propuesta de App Móvil para la gestión de incidentes de tránsito," *Revista Ibérica de Sistemas e Tecnologias de Informação*, no. E55, pp. 67–76, 2023.

[4] R. Toasa, C. Silva, C. Silva, D. Goncalves, L. Neves, and L. Marcelino, "Energy consumption behaviour characterization with mobile gamification," in *Iberian Conference on Information Systems and Technologies, CISTI*, IEEE Computer Society, Jul. 2017. doi: 10.23919/CISTI.2017.7975900.

[5] K. Vishal and A. S. Kushwaha, "Mobile Application Development Research Based on Xamarin Platform," *Proceedings - 4th International Conference on Computing Sciences, ICCS 2018*, pp. 115–118, Jan. 2019, doi: 10.1109/ICCS.2018.00027.

[6] R. M. Toasa, P. F. B. Egas, H. Recalde, and M. G. Saltos, "Mobile Development with Xamarin: Brief Literature, Visualizations and Important Issues," *Lecture Notes in Networks and Systems*, vol. 692 LNNS, pp. 299–307, 2023, doi: 10.1007/978-3-031-33261-6_26/COVER.

[7] J. Bennett, *Xamarin in Action: Creating native cross-platform mobile apps - Jim Bennett - Google Libros*. 2018. Accessed: Jul. 31, 2022. [Online]. Available: https://books.google.com.ec/books?hl=es&lr=&id=8TczEAAAQBAJ&oi=fnd&pg=PT16&dq=XAMARIN+&ots=U9gDj_FKdf&sig=l9HBTQmX-NgGbpwI5ZSm1Zh0TcE&redir_esc=y#v=onepage&q=XAMARIN&f=false

Período julio-diciembre 2023
Vol. 2 No. 2
riif@editorialscientificfuture.com
ISSN-L: 3028-869X **DOI:** https://doi.org/10.62465/riif.v2n2.2023.50