

Actividad 2 - Aplicación 2

Desarrollo de aplicaciones móviles III

Ingeniería en Desarrollo de Software

Tutor: Sandra Luz Lara Dévora

Alumno: Fernando Pedraza Garate

Fecha: 01 de agosto 2024

Índice

Etapa 1 – Instalación de Xcode/ Aplicación número par o impar

- Introducción. Pág. 3
- Descripción Pág. 4 - 5
- Justificación Pág. 6 - 7
- Desarrollo Pág. 8 - 16
 - Recursos
 - Codificación
 - Prueba de la aplicación

Etapa 2 – Aplicación 2 / Inventario

- Desarrollo Pág. 17 - 28
 - Codificación
 - Prueba de la aplicación
- Conclusión Pág. 29
- Referencias Pág. 30 - 31

Introducción

Empezaremos por analizar y entender que es un compilador, un compilador es un programa que traduce el código fuente escrito en un lenguaje de programación de alto nivel (como C, C++, Java) a un lenguaje de bajo nivel (como el lenguaje maquina) que una computadora puede entender y ejecutar.

Xcode es la herramienta principal utilizada por los desarrolladores de Apple para crear aplicaciones nativas para las plataformas de Apple, es un entorno de desarrollo integrado (IDE) creado por Apple el cual está diseñado para el desarrollo de aplicaciones para los sistemas operativos de Apple, incluyendo macOS, iOS, iPadOS, watchOS y tvOS, que proporciona a los desarrolladores una variedad de herramientas necesarias para crear software, como incluir un editor de código, una interfaz gráfica de usuario (GUI), un compilador, un depurador, simuladores, herramientas de prueba, y control de versiones como Git para la gestión del código fuente, y para poder desarrollar aplicaciones para las plataformas de Apple desde Windows se deberán utilizar los compiladores existentes en línea como: Swift, el cual es un lenguaje de programación desarrollado por Apple diseñado para ser un lenguaje poderoso y fácil de usar en el desarrollo de aplicaciones para macOS, iOS, iPadOS, watchOS y tvOS, o también, se puede utilizar Replit, que es una herramienta versátil y accesible que admite una gran variedad de lenguajes de programación y proporciona un entorno de desarrollo completo sin necesidad de instalar software adicional, al ser una plataforma en línea que permite a los desarrolladores escribir, ejecutar, y colaborar en código desde un navegador web. (*ChatGPT*, n.d.)

Descripción.

Para esta actividad, se utilizará la plataforma Replit debido a su versatilidad y accesibilidad, lo que simplifica el desarrollo de aplicaciones en un entorno en línea, donde Replit permite escribir, ejecutar y depurar código directamente desde un navegador web, eliminando la necesidad de instalar software adicional en el sistema operativo Windows, siendo fundamental contar con un buen conocimiento del funcionamiento de Replit para aprovechar al máximo sus características, incluyendo familiarizarse con su editor de código, entender cómo configurar y utilizar el entorno de ejecución, y cómo colaborar en tiempo real con otros desarrolladores si es necesario, además, saber cómo utilizar las plantillas y los proyectos de inicio en Replit permite acelerar el proceso de desarrollo, proporcionando un punto de partida para la creación de la aplicación, implicando varios pasos, como diseñar la interfaz de usuario para ingresar el número, escribir la lógica para determinar si el número es par o impar, y asegurar que la aplicación funcione correctamente en diferentes escenarios.

Aprovechar las herramientas de depuración y prueba de Replit será crucial para identificar y corregir errores durante el desarrollo.

Así como también se deberá desarrollar una aplicación en lenguaje Swift para la tienda de la esquina, basados en la necesidad que presentan para cubrir su requerimiento de controlar diversas funciones de inventario, dicha aplicación deberá contar con un menú con las especificaciones para poder registrar artículos nuevos en la misma, que les permita poder visualizar listados sus productos existentes a inventariar, así como consultar su existencia física, y que les permita salir de la aplicación, si así se requiere, para lograr obtener un mejor control en la gestión de sus inventarios y existencias físicas.

Justificación.

Replit es una plataforma en línea que ofrece numerosas ventajas para desarrollar aplicaciones, especialmente para aquellos que buscan un entorno de desarrollo accesible, colaborativo y eficiente, algunas de las principales razones para usar Replit para el desarrollo de aplicaciones es que es accesible desde cualquier navegador web, lo que significa que se puede escribir y ejecutar código desde cualquier lugar y en cualquier dispositivo con conexión a internet, no se requiere instalación de software adicional, todo lo necesario para escribir, compilar y ejecutar código está integrado en la plataforma, soportando una amplia variedad de lenguajes de programación, lo que lo hace ideal para aprender y desarrollar en múltiples lenguajes sin cambiar de entorno, ofrece un editor de código completo con características como resaltado de sintaxis, autocompletado, y depuración, permite ejecutar y probar código en tiempo real dentro del navegador, facilitando el desarrollo y la depuración rápida, similar a Google Docs, Replit permite la edición colaborativa en tiempo real, lo que facilita trabajar en proyectos con otros desarrolladores, compartiendo proyectos fácilmente con un enlace que permitir a otros ver o editar el código, ofreciendo plantillas predefinidas y proyectos de inicio que pueden ayudar a comenzar rápidamente con nuevas aplicaciones y tecnologías, integración con GitHub para gestionar el control de versiones del código, facilitando la colaboración y la gestión de proyectos a largo plazo, ampliamente utilizado en entornos educativos debido a su facilidad de uso y su capacidad para soportar una amplia gama de lenguajes de programación, proporcionando herramientas específicas para profesores y estudiantes, como la creación de asignaciones y el seguimiento del progreso, ejecutando cada proyecto en un entorno aislado, lo que proporciona un

nivel adicional de seguridad y evita conflictos entre diferentes proyectos, además de contar con una comunidad activa y gran cantidad de recursos y documentación que pueden ayudar a resolver problemas y aprender nuevas habilidades, ofreciendo una versión gratuita con funcionalidades completas suficientes para la mayoría de los proyectos, con opciones premium para características adicionales y mayor capacidad de recursos.

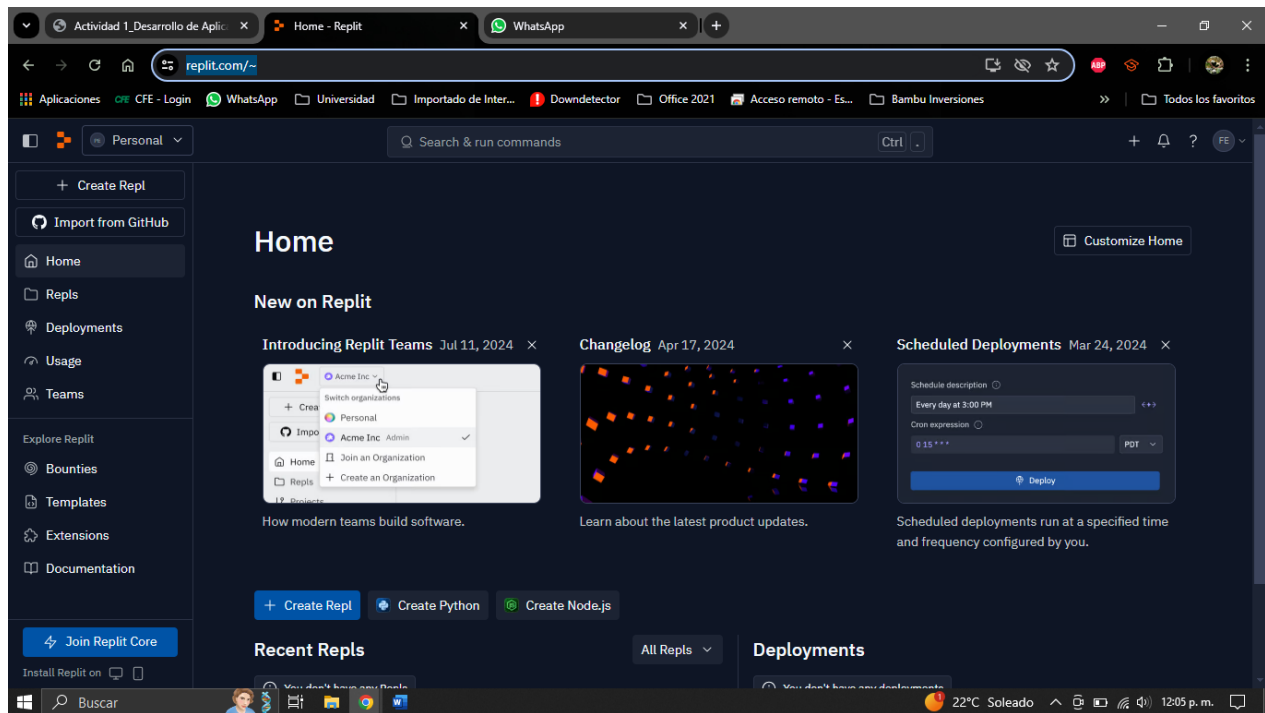
Estas características hacen de Replit una opción atractiva tanto para principiantes como para desarrolladores experimentados que buscan una plataforma flexible y accesible para el desarrollo de aplicaciones.

Desarrollo.

Recursos

Realizar instalación de XCode o utilizar los compiladores online (según la preferencia) y crear la aplicación solicitada.

Software: <https://replit.com/~>



Objetivo:

- Crear una aplicación donde sea posible ingresar un número y diga si es número par o impar.

Codificación

//Se importa el módulo Foundation para usar funciones básicas de entrada y salida

import Foundation

//Se especifica la función para verificar si un número es par o impar

func esPar(numero: Int) -> Bool {

return numero % 2 == 0

}

// Usando **print** se muestra en pantalla la información solicitada al usuario, en este caso que ingrese un número y **readline ()** para que el programa lea la entrada del usuario desde la consola.

print("Ingresa un número:")

if let input = readLine(), let numero = Int(input) {

// El método para solicitar al usuario que ingrese un número, se define con la función **esPar**, la cual toma un número entero como parámetro y devuelve **true** si el número es par y **false** si es impar, se convierte la entrada del usuario a un número entero usando **Int()**.

if esPar(numero: numero) {

print("\(numero) es un número par.")

```
} else {
```

```
    print("\n(numero) es un número impar.")
```

```
}
```

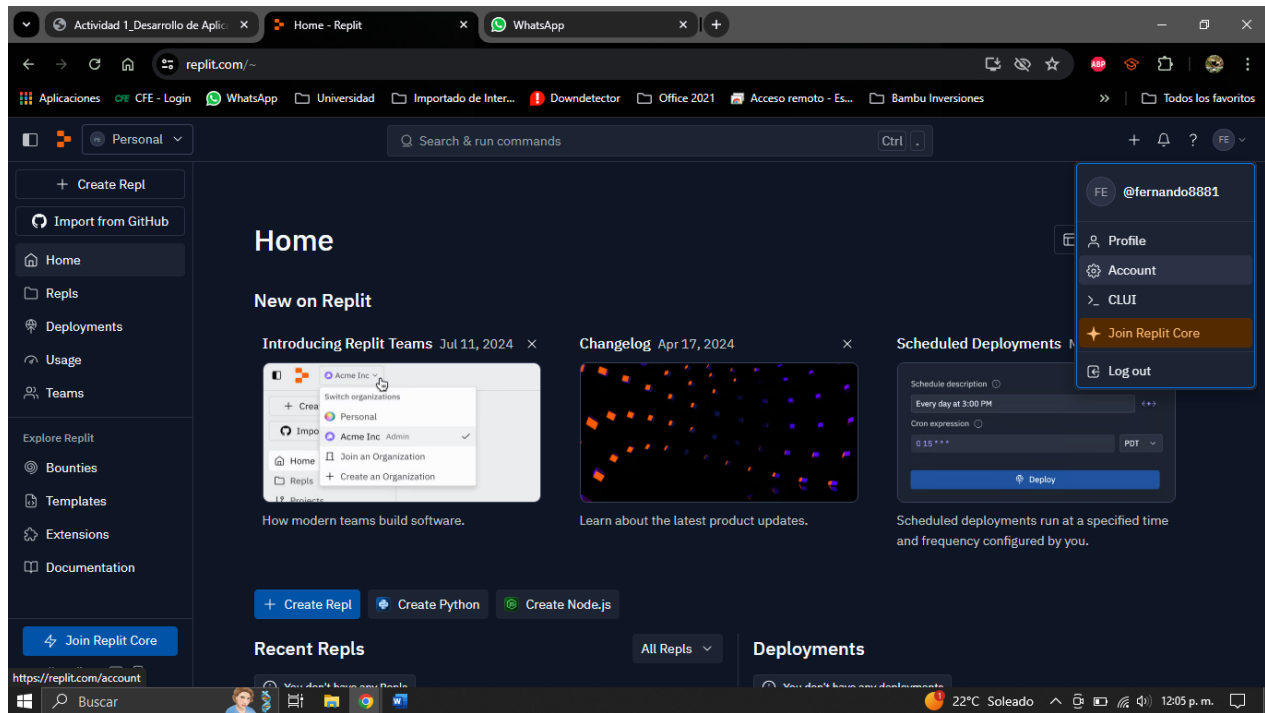
```
// Si la conversión es exitosa, se verifica si el número es par o impar usando la función esPar y se muestra el resultado,
```

```
} else {
```

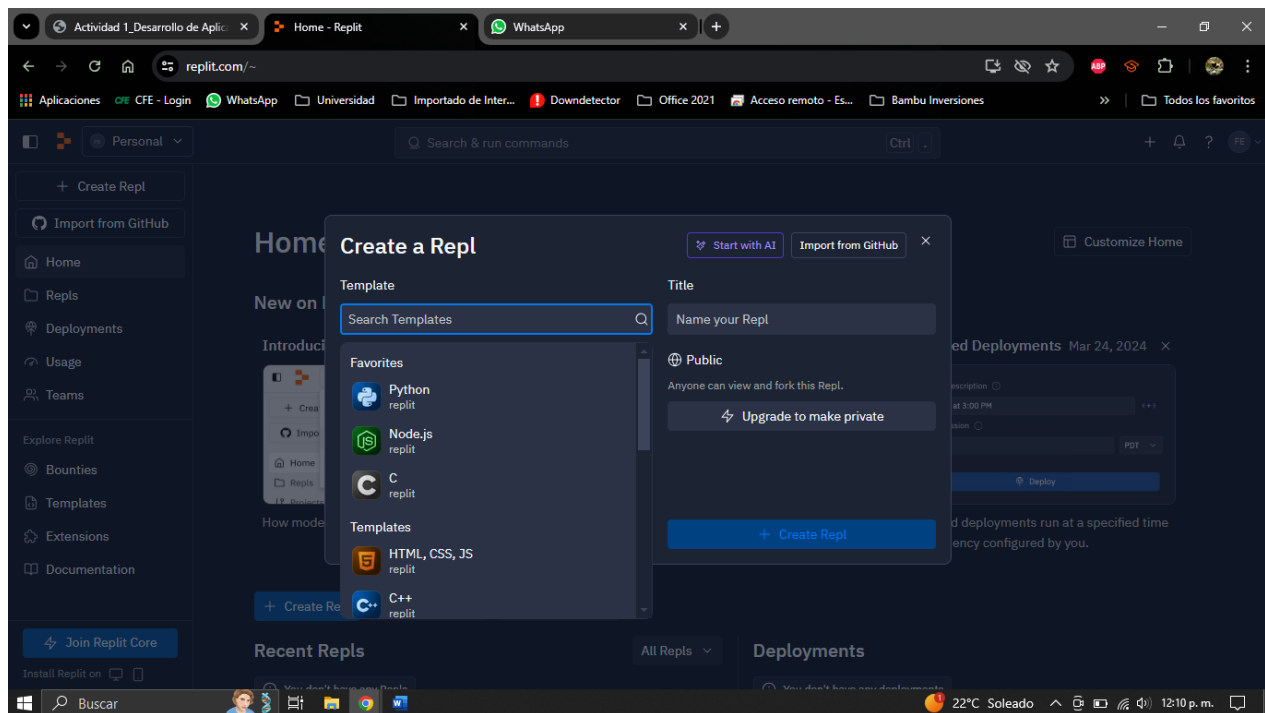
```
    print("Por favor, ingresa un número válido.")
```

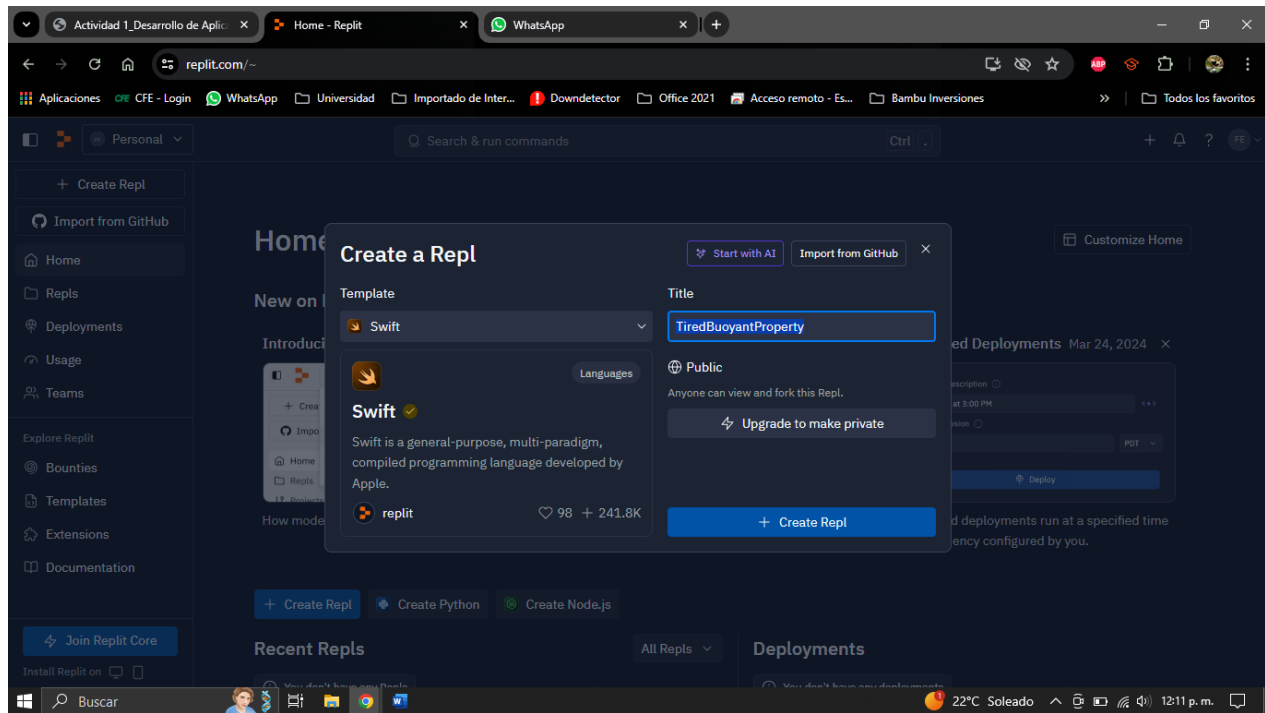
```
}
```

```
//Si la conversión falla o se ingresa un carácter diferente se muestra un mensaje de error.
```

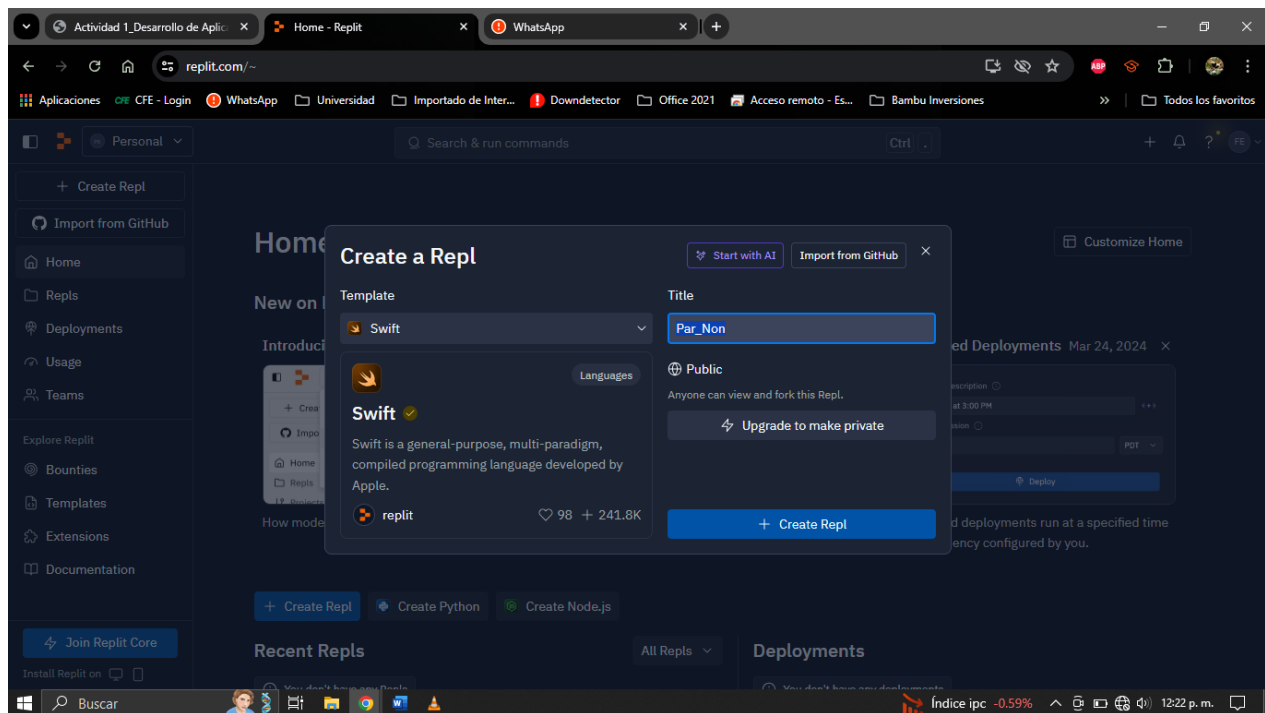


Una vez dentro de la plataforma se crea un nuevo repl.

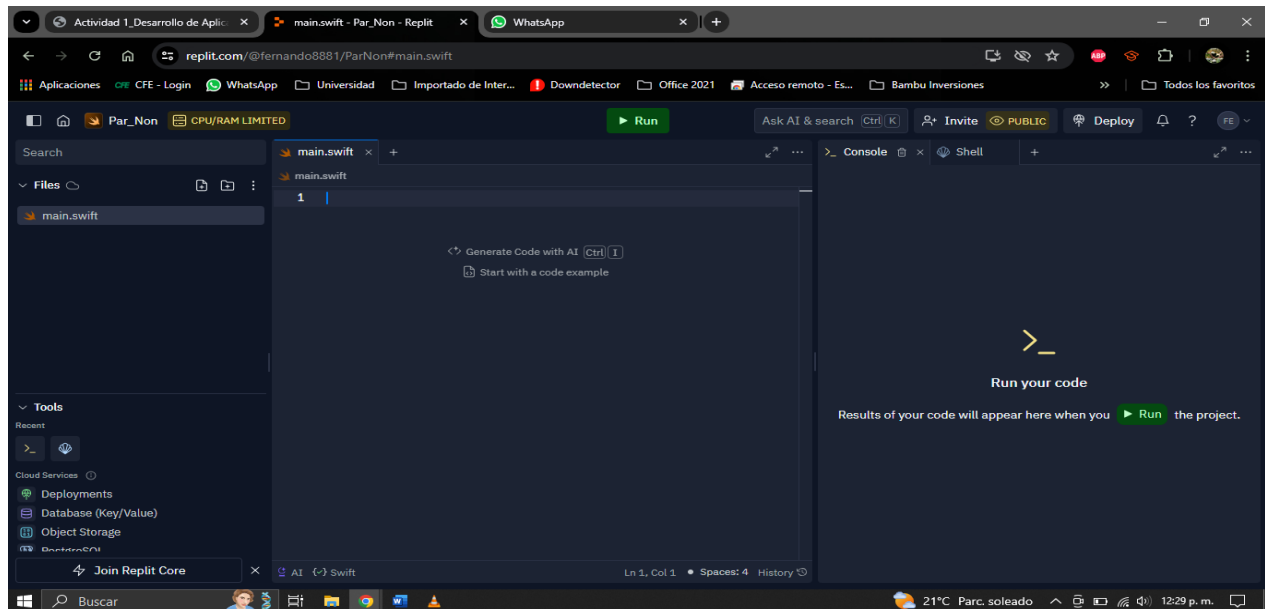




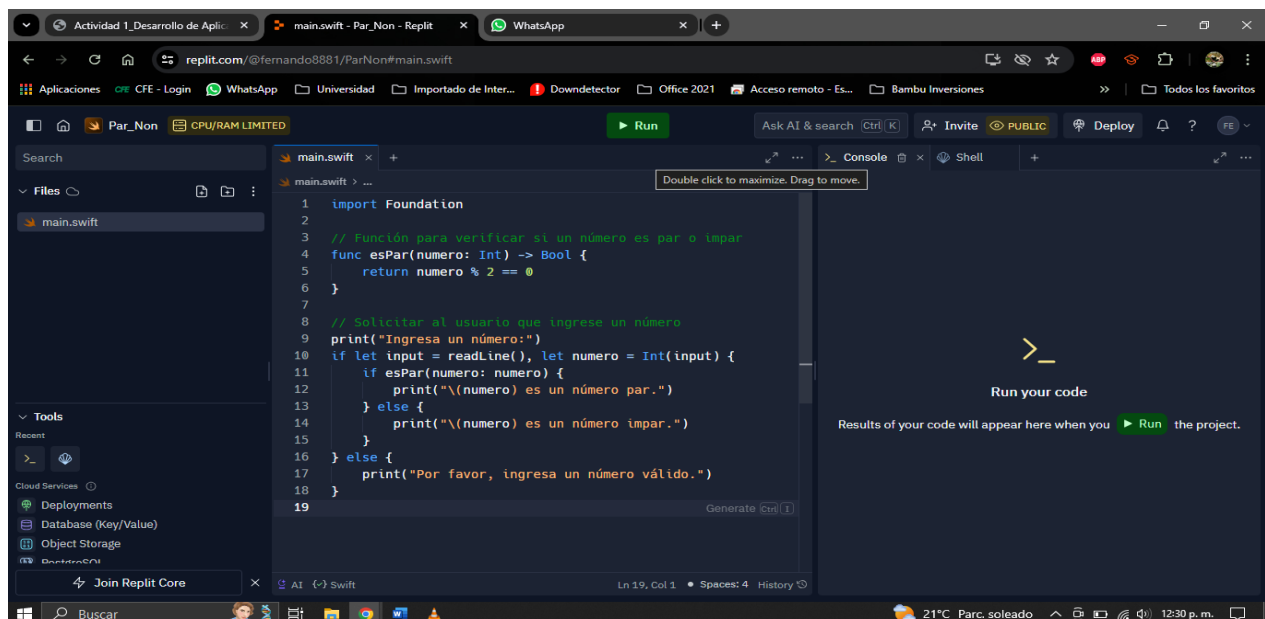
Se selecciona el lenguaje Swift y se asigna el título Par_Non.



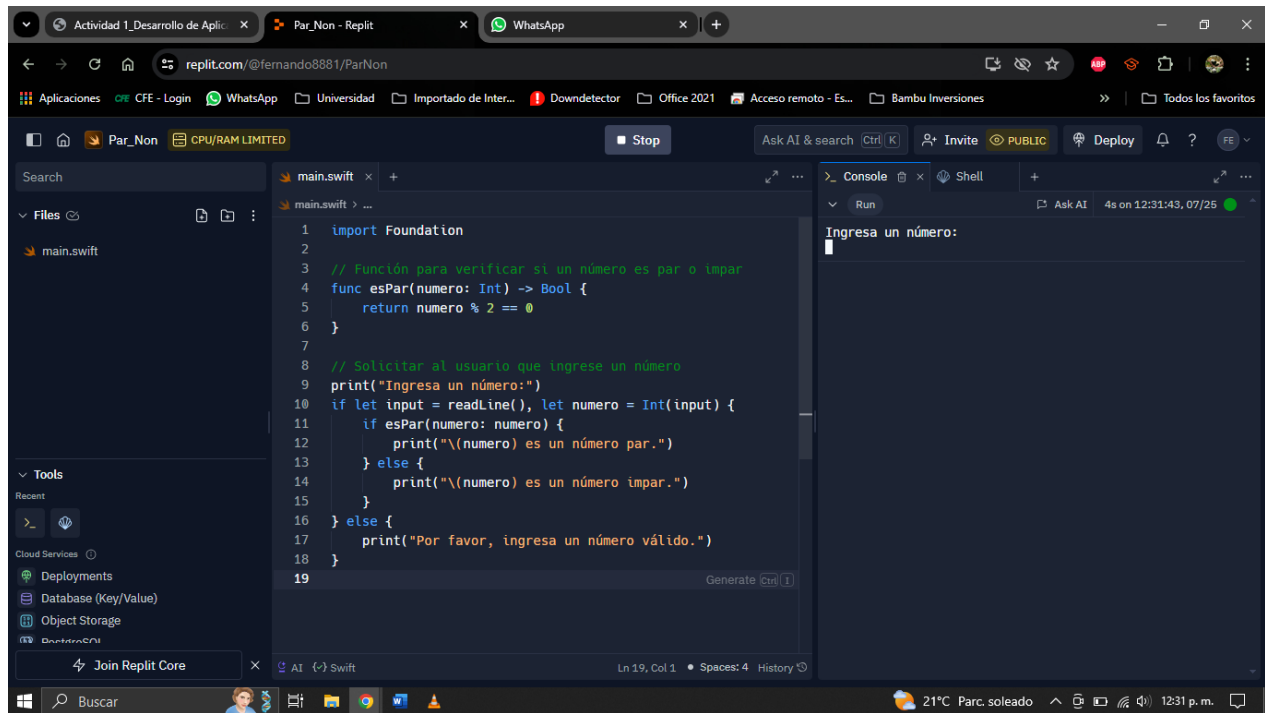
Prueba de la aplicación



Una vez en el entorno main se escribe el código para la realización del programa



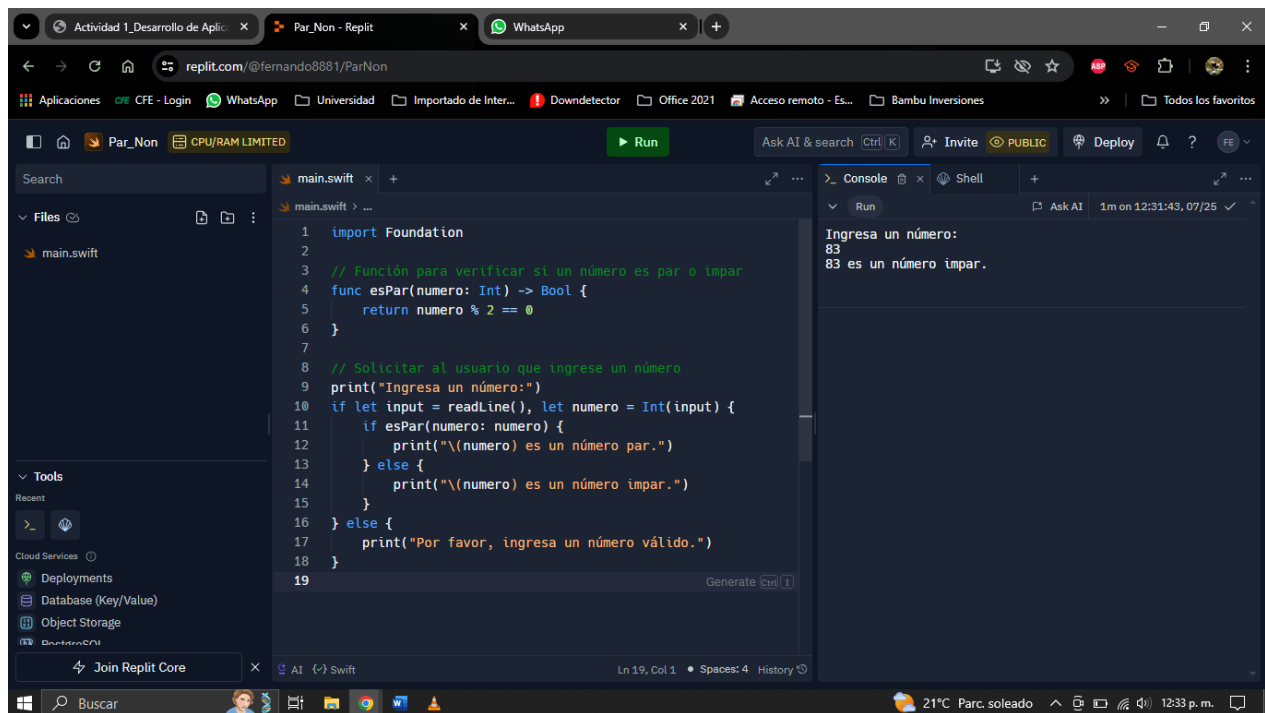
Se da click en Run para ejecutarlo



The screenshot shows a Replit environment with a Swift script in the main.swift file. The script defines a function `esPar` that checks if a number is even or odd. The console is empty, and the prompt "Ingresa un número:" is visible.

```
1 import Foundation
2
3 // Función para verificar si un número es par o impar
4 func esPar(numero: Int) -> Bool {
5     return numero % 2 == 0
6 }
7
8 // Solicitar al usuario que ingrese un número
9 print("Ingresa un número:")
10 if let input = readLine(), let numero = Int(input) {
11     if esPar(numero: numero) {
12         print("\(numero) es un número par.")
13     } else {
14         print("\(numero) es un número impar.")
15     }
16 } else {
17     print("Por favor, ingresa un número válido.")
18 }
19
```

Se solicita en consola se ingrese un número



The screenshot shows the same Replit environment after running the script. The console displays the output for the input 83, indicating it is an odd number.

```
1 import Foundation
2
3 // Función para verificar si un número es par o impar
4 func esPar(numero: Int) -> Bool {
5     return numero % 2 == 0
6 }
7
8 // Solicitar al usuario que ingrese un número
9 print("Ingresa un número:")
10 if let input = readLine(), let numero = Int(input) {
11     if esPar(numero: numero) {
12         print("\(numero) es un número par.")
13     } else {
14         print("\(numero) es un número impar.")
15     }
16 } else {
17     print("Por favor, ingresa un número válido.")
18 }
19
```

Ingresa un número:
83
83 es un número impar.

Muestra el resultado correcto al ser un número impar

The screenshot shows a Replit IDE window with the URL `replit.com/@fernando8881/ParNon`. The main.swift file contains the following Swift code:

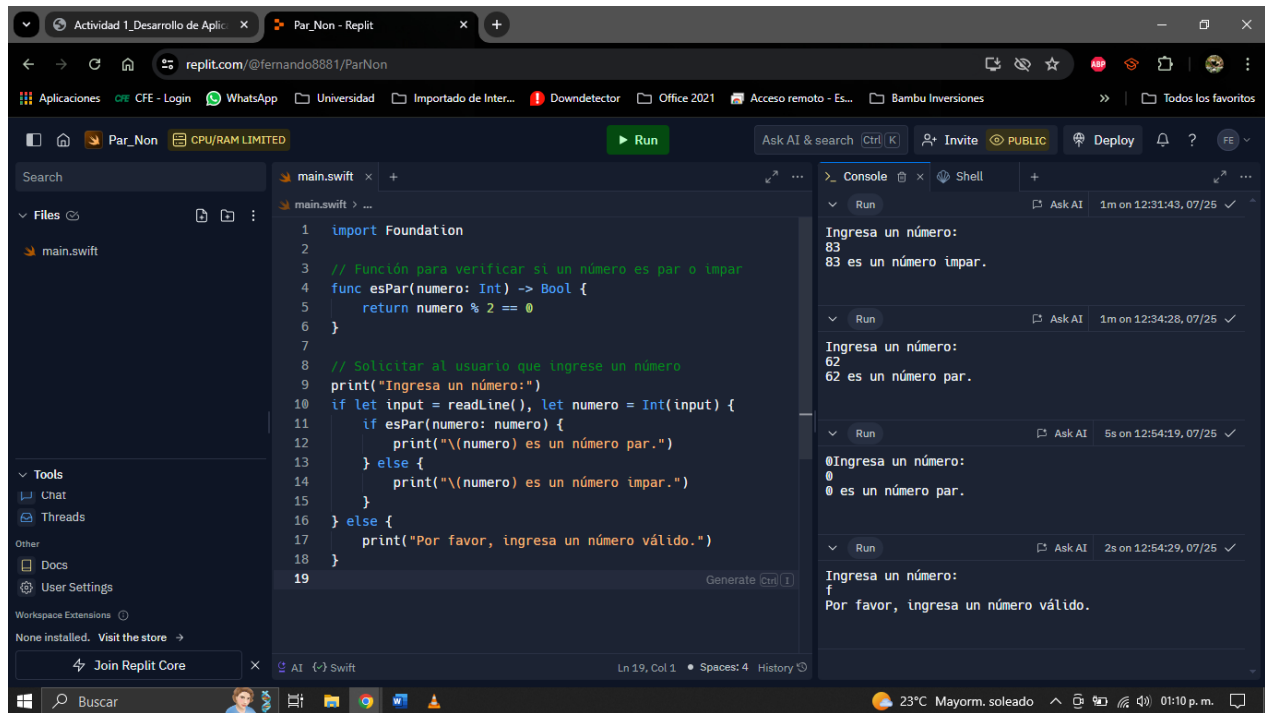
```
1 import Foundation
2
3 // Función para verificar si un número es par o impar
4 func esPar(numero: Int) -> Bool {
5     return numero % 2 == 0
6 }
7
8 // Solicitar al usuario que ingrese un número
9 print("Ingresa un número:")
10 if let input = readLine(), let numero = Int(input) {
11     if esPar(numero: numero) {
12         print("\(numero) es un número par.")
13     } else {
14         print("\(numero) es un número impar.")
15     }
16 } else {
17     print("Por favor, ingresa un número válido.")
18 }
19
```

The console on the right shows the program's execution. It prompts "Ingresa un número:" and receives the input "83". The output is "83 es un número impar.".

Se da click en Run para ejecutar nuevamente el programa y solicite el ingreso del número.

This screenshot shows the same Replit IDE window after clicking the "Run" button. The code in main.swift is identical to the previous screenshot. The console now shows a new execution where the input is "62". The output is "62 es un número par.".

Mostrando el resultado de forma correcta, indicando que es un número par.



```
1 import Foundation
2
3 // Función para verificar si un número es par o impar
4 func esPar(numero: Int) -> Bool {
5     return numero % 2 == 0
6 }
7
8 // Solicitar al usuario que ingrese un número
9 print("Ingresa un número:")
10 if let input = readLine(), let numero = Int(input) {
11     if esPar(numero: numero) {
12         print("\(numero) es un número par.")
13     } else {
14         print("\(numero) es un número impar.")
15     }
16 } else {
17     print("Por favor, ingresa un número válido.")
18 }
19
```

Console output:

```
Ingresa un número:
83
83 es un número impar.

Ingresa un número:
62
62 es un número par.

Ingresa un número:
0
0 es un número par.

Ingresa un número:
f
Por favor, ingresa un número válido.
```

Se da click nuevamente en Run y en caso de escribir algún carácter invalido se solicita el ingreso de un número valido.

Link Replit: <https://replit.com/@fernando8881/ParNon?v=1>

Link Google Drive:

<https://drive.google.com/file/d/13L5AOSb8xmmJfLzQIA7EBjeZCDvKAfpv/view?usp=sharing>

Desarrollo.

Codificación

```
struct Artículo { //crea la estructura artículo

    var nombre: String //crea la variable nombre

    var cantidad: Int //crea la variable cantidad

} //cierra la estructura artículo


func registroArtículo() -> Artículo { //crea la función registroArtículo

    print("Registrar artículos") //imprime el mensaje

    print("Ingrese el nombre del artículo:") //imprime el mensaje

    let nombre = readLine() ?? "" //lee la variable nombre

    print("Ingrese existencia de inventario:") //imprime el mensaje

    let cantidad = Int(readLine()!) ?? 0 //lee la variable cantidad

    return Artículo(nombre: nombre, cantidad: cantidad) //retorna la estructura artículo

} //cierra la función registroArtículo
```

```

func mostrarListado(artículos: [Artículo]) { //crea la función mostrarListado

    print("Listado de artículos") //imprime el mensaje

    for (indice, artículo) in artículos.enumerated() { //recorre el arreglo artículos

        print("Artículo \$(indice + 1): \$(artículo.nombre)") //imprime el mensaje

        print("Cantidad: \$(artículo.cantidad)") //imprime el mensaje

    } //cierra el ciclo for

} //cierra la función mostrarListado

```

```

func consultadeArtículo(artículos: [Artículo]) { //crea la función consultadeArtículo

    print("Existencias") //imprime el mensaje

    print("Ingrese la descripción del artículo a consultar:") //imprime el mensaje

    let nombre = readLine() ?? "" //lee la variable nombre

    var encontrado = false //crea la variable encontrado

    for (indice, artículo) in artículos.enumerated() { //recorre el arreglo artículos

        if artículo.nombre == nombre { //compara la variable nombre

```

```
print("Artículo \(<indice + 1>): \(<artículo.nombre>") //imprime el mensaje
```

```
print("Cantidad: \(<artículo.cantidad>") //imprime el mensaje
```

```
encontrado = true //asigna el valor true a la variable encontrado
```

```
break //rompe el ciclo for
```

```
} //cierra el if
```

```
} //cierra el ciclo for
```

```
if !encontrado { //compara la variable encontrado
```

```
print("Artículo no encontrado.") //imprime el mensaje
```

```
} //cierra el if
```

```
} //cierra la función consultadeArtículo
```

```
var artículos: [Artículo] = [] //crea el arreglo artículos
```

```
var opción = 0 //crea la variable opción
```

```
repeat { //repite el ciclo hasta que se cumpla la condición
```

```
print("1. Registro de artículos") //imprime el mensaje
```

```
print("2. Listado de artículos") //imprime el mensaje
```

```
print("3. Consulta de existencias") //imprime el mensaje
```

```
print("4. Salir") //imprime el mensaje
```

```
print("Selecciona una opción:") //imprime el mensaje
```

```
if let entrada = readLine(), let seleccion = Int(entrada) { //compara la variable entrada y  
seleccion
```

```
    opción = seleccion //asigna el valor seleccion a la variable opción
```

```
} else { //si no se cumple la condición
```

```
    print("Opción inválida. Por favor ingrese un número.") //imprime el mensaje
```

```
    continue //continua el ciclo
```

```
} //cierra el if
```

```
switch opción { //compara la variable opción
```

```
case 1: //si la variable opción es igual a 1
```

```
    let nuevoArtículo = registroArtículo() //asigna el valor de la función registroArtículo a la  
variable nuevoArtículo
```

```
    artículos.append(nuevoArtículo) //agrega el valor de la variable nuevoArtículo al arreglo  
artículos
```

```
    print("***** Registro exitoso *****") //imprime el mensaje
```

```
case 2: //si la variable opción es igual a 2
```

```
    mostrarListado(artículos: artículos) //llama a la función mostrarListado
```

```
case 3: //si la variable opción es igual a 3
```

```
    consultadeArtículo(artículos: artículos) //llama a la función consultadeArtículo
```

```
case 4: //si la variable opción es igual a 4
```

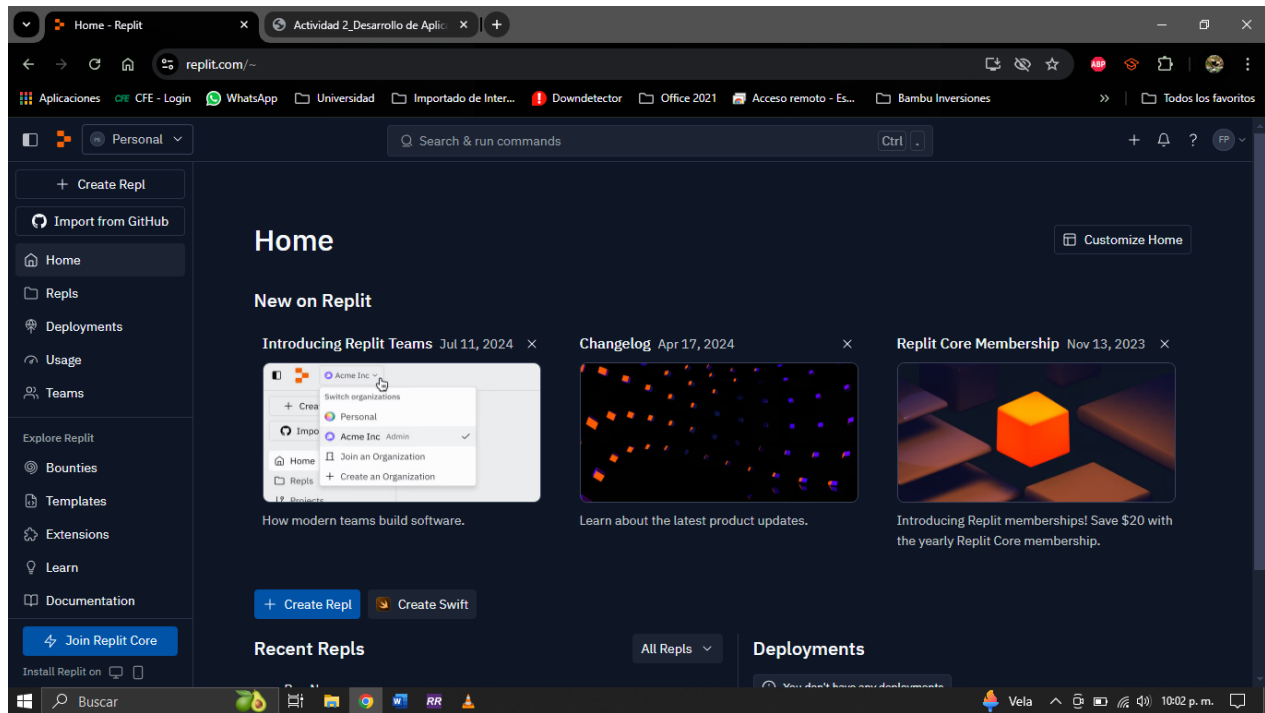
```
    print("Salir") //imprime el mensaje
```

```
default: //si no se cumple ninguna de las condiciones
```

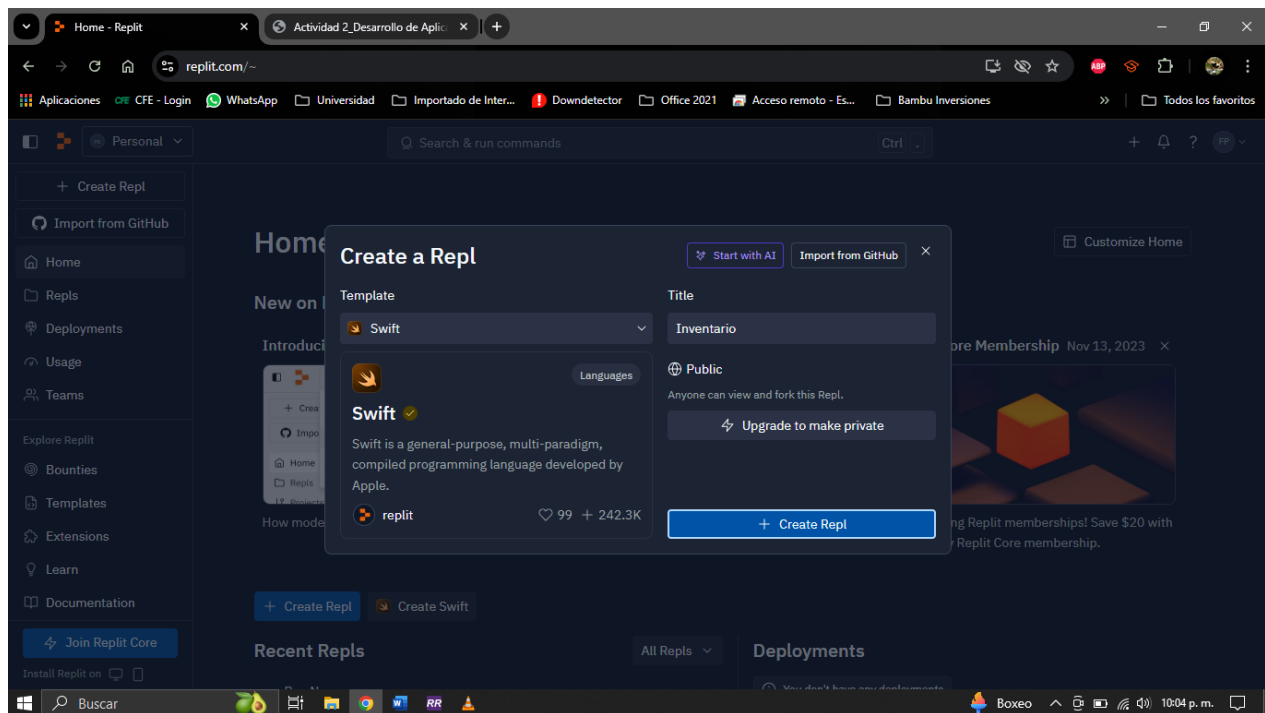
```
    print("Opción inválida. Por favor seleccione una opción entre 1 y 4.") //imprime el mensaje
```

```
    } //cierra el switch
```

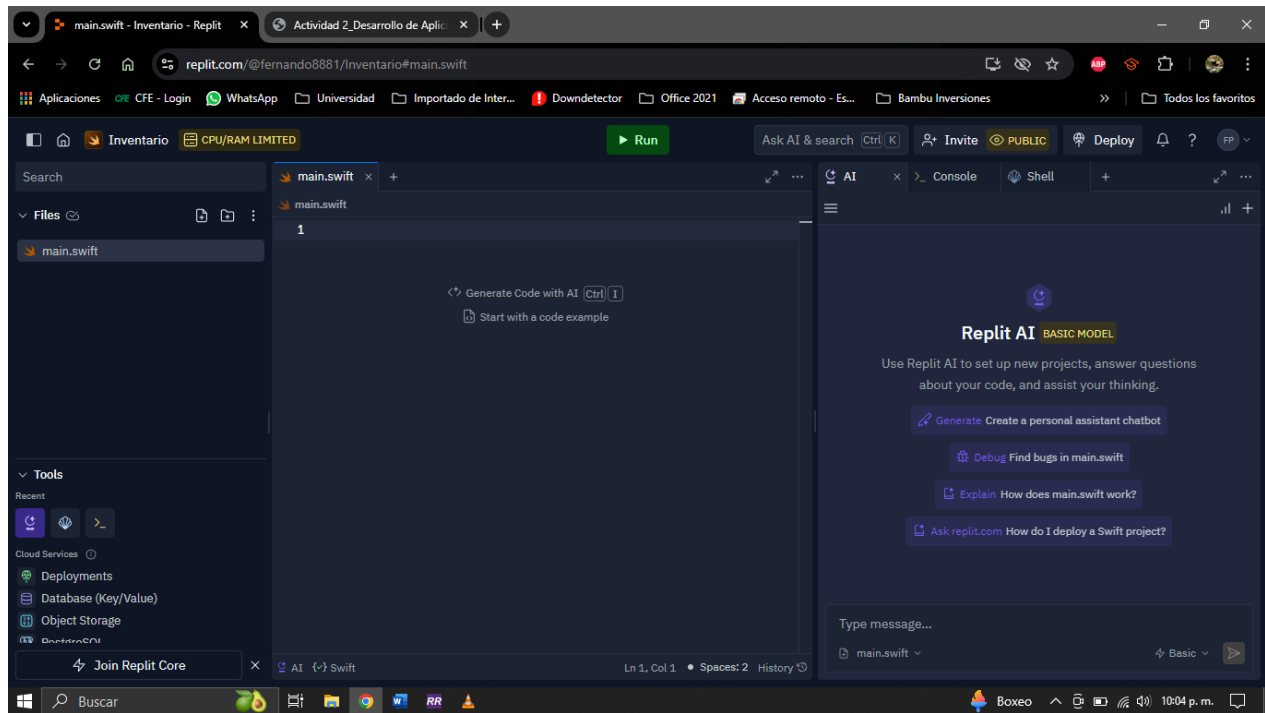
```
    } while opción != 4 //repite el ciclo hasta que la variable opción sea igual a 4
```



Se ingresa a la plataforma de Swift

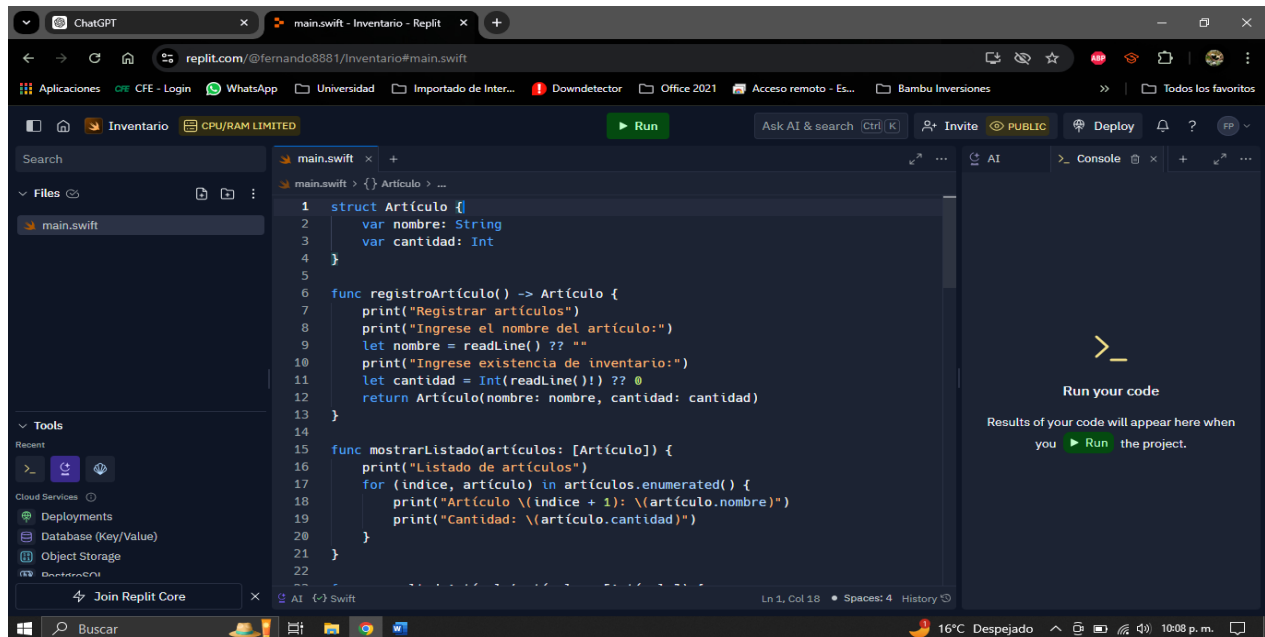


Se crea el Repl con el nombre Inventario en el lenguaje Swift

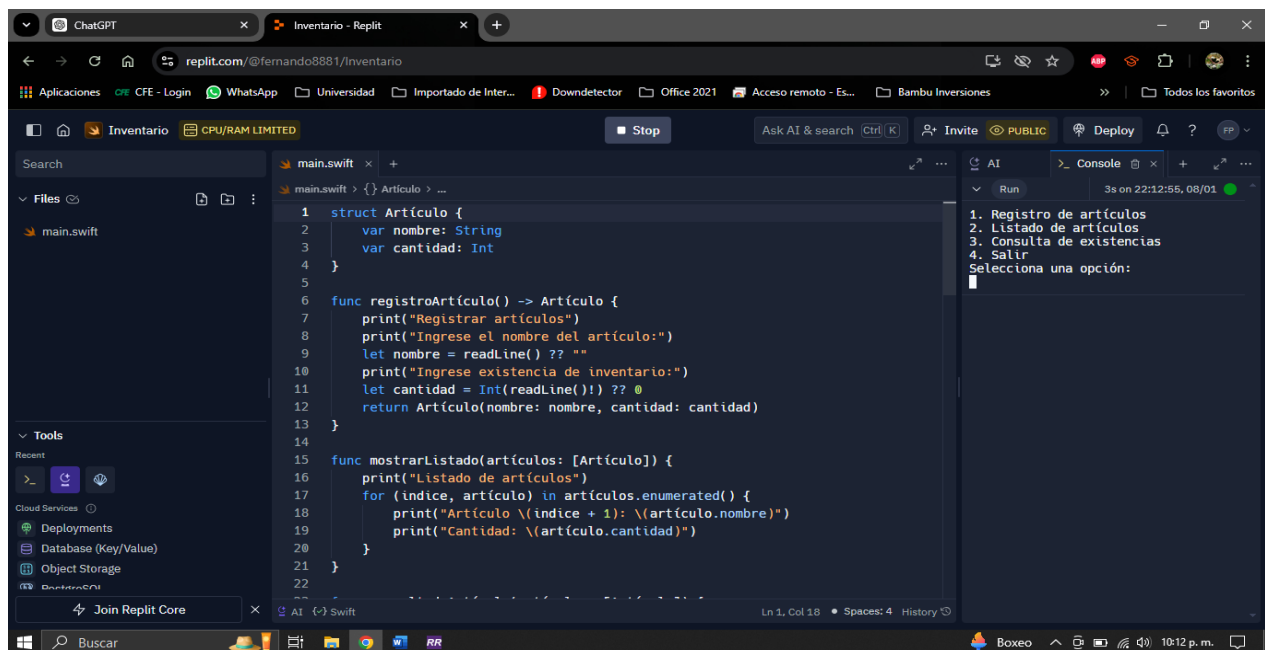


Una vez creada la interfaz se crea la aplicación en base al requerimiento solicitado

Prueba de la aplicación



Se hace click en el botón Run



Se muestra el programa en la consola y se ingresan los datos solicitados


```
1 struct Articulo {
2     var nombre: String
3     var cantidad: Int
4 }
5
6 func registroArticulo() -> Articulo {
7     print("Registrar articulos")
8     print("Ingrese el nombre del articulo:")
9     let nombre = readLine() ?? ""
10    print("Ingrese existencia de inventario:")
11    let cantidad = Int(readLine()!) ?? 0
12    return Articulo(nombre: nombre, cantidad: cantidad)
13 }
14
15 func mostrarListado(articulos: [Articulo]) {
16     print("Listado de articulos")
17     for (indice, articulo) in articulos.enumerated() {
18         print("Articulo \(indice + 1): \(articulo.nombre)")
19         print("Cantidad: \(articulo.cantidad)")
20     }
21 }
22
```

Console output:

```
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
1
Registrar articulos
Ingrese el nombre del artículo:
Motocicleta
Ingrese existencia de inventario:
1
**** Registro exitoso ****
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
```

Creando el articulo Motocicleta con una existencia de 1 y una vez registrado el programa solicita nuevamente seleccionar una opción

```
1 struct Articulo {
2     var nombre: String
3     var cantidad: Int
4 }
5
6 func registroArticulo() -> Articulo {
7     print("Registrar articulos")
8     print("Ingrese el nombre del articulo:")
9     let nombre = readLine() ?? ""
10    print("Ingrese existencia de inventario:")
11    let cantidad = Int(readLine()!) ?? 0
12    return Articulo(nombre: nombre, cantidad: cantidad)
13 }
14
15 func mostrarListado(articulos: [Articulo]) {
16     print("Listado de articulos")
17     for (indice, articulo) in articulos.enumerated() {
18         print("Articulo \(indice + 1): \(articulo.nombre)")
19         print("Cantidad: \(articulo.cantidad)")
20     }
21 }
22
```

Console output:

```
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
1
Registrar articulos
Ingrese el nombre del artículo:
Motocicleta
Ingrese existencia de inventario:
1
**** Registro exitoso ****
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
3
Registrar articulos
Ingrese el nombre del artículo:
Televisión
Ingrese existencia de inventario:
3
**** Registro exitoso ****
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
```

Se registra el siguiente articulo como Televisión con una existencia de 3

```
1 struct Articulo {
2     var nombre: String
3     var cantidad: Int
4 }
5
6 func registroArticulo() -> Articulo {
7     print("Registrar articulos")
8     print("Ingrese el nombre del articulo:")
9     let nombre = readLine() ?? ""
10    print("Ingrese existencia de inventario:")
11    let cantidad = Int(readLine()!) ?? 0
12    return Articulo(nombre: nombre, cantidad: cantidad)
13 }
14
15 func mostrarListado(articulos: [Articulo]) {
16     print("Listado de articulos")
17     for (indice, articulo) in articulos.enumerated() {
18         print("Articulo \(indice + 1): \(articulo.nombre)")
19         print("Cantidad: \(articulo.cantidad)")
20     }
21 }
22
```

Console output:

```
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
1
Registrar articulos
Ingrese el nombre del articulo:
Television
Ingrese existencia de inventario:
3
***** Registro exitoso *****
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
1
Registrar articulos
Ingrese el nombre del articulo:
Laptop
Ingrese existencia de inventario:
2
***** Registro exitoso *****
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
```

Se registra el siguiente articulo como Laptop con una existencia de 2

```
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
1
Registrar articulos
Ingrese el nombre del articulo:
Laptop
Ingrese existencia de inventario:
2
***** Registro exitoso *****
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
2
Listado de articulos
Articulo 1: Motocicleta
Cantidad: 1
Articulo 2: Television
Cantidad: 3
Articulo 3: Laptop
Cantidad: 2
1. Registro de articulos
2. Listado de articulos
3. Consulta de existencias
4. Salir
Selecciona una opción:
```

Se selecciona la opción 2 para listar los artículos

```
1 struct Artículo {
2     var nombre: String
3     var cantidad: Int
4 }
5
6 func registroArticulo() -> Artículo {
7     print("Registrar artículos")
8     print("Ingrese el nombre del artículo:")
9     let nombre = readLine() ?? ""
10    print("Ingrese existencia de inventario:")
11    let cantidad = Int(readLine()!) ?? 0
12    return Artículo(nombre: nombre, cantidad: cantidad)
13 }
14
15 func mostrarListado(articulos: [Artículo]) {
16     print("Listado de artículos")
17     for (indice, articulo) in articulos.enumerated() {
18         print("Artículo \(indice + 1): \(articulo.nombre)")
19         print("Cantidad: \(articulo.cantidad)")
20     }
21 }
22
```

2. Listado de artículos
3. Consulta de existencias
4. Salir
Selecciona una opción:
2
Listado de artículos
Artículo 1: Motocicleta
Cantidad: 1
Artículo 2: Televisión
Cantidad: 3
Artículo 3: Laptop
Cantidad: 2
1. Registro de artículos
2. Listado de artículos
3. Consulta de existencias
4. Salir
Selecciona una opción:

Mostrando sus descripciones y existencias

```
1 struct Artículo {
2     var nombre: String
3     var cantidad: Int
4 }
5
6 func registroArticulo() -> Artículo {
7     print("Registrar artículos")
8     print("Ingrese el nombre del artículo:")
9     let nombre = readLine() ?? ""
10    print("Ingrese existencia de inventario:")
11    let cantidad = Int(readLine()!) ?? 0
12    return Artículo(nombre: nombre, cantidad: cantidad)
13 }
14
15 func mostrarListado(articulos: [Artículo]) {
16     print("Listado de artículos")
17     for (indice, articulo) in articulos.enumerated() {
18         print("Artículo \(indice + 1): \(articulo.nombre)")
19         print("Cantidad: \(articulo.cantidad)")
20     }
21 }
22
```

2. Listado de artículos
3. Consulta de existencias
4. Salir
Selecciona una opción:
3
Existencias
Ingrese la descripción del artículo
a consultar:
Laptop
Artículo 3: Laptop
Cantidad: 2
1. Registro de artículos
2. Listado de artículos
3. Consulta de existencias
4. Salir
Selecciona una opción:

Se selecciona la opción 3 para consultar la existencia por articulo

```
1 struct Artículo {
2     var nombre: String
3     var cantidad: Int
4 }
5
6 func registrarArtículo() -> Artículo {
7     print("Registrar artículos")
8     print("Ingrese el nombre del artículo:")
9     let nombre = readLine() ?? ""
10    print("Ingrese existencia de inventario:")
11    let cantidad = Int(readLine()!) ?? 0
12    return Artículo(nombre: nombre, cantidad: cantidad)
13 }
14
15 func mostrarListado(artículos: [Artículo]) {
16     print("Listado de artículos")
17     for (indice, artículo) in artículos.enumerated() {
18         print("Artículo \(indice + 1): \(artículo.nombre)")
19         print("Cantidad: \(artículo.cantidad)")
20     }
21 }
22
```

Console Output:

```
4. Salir
Selecciona una opción:
2
Listado de artículos
Artículo 1: Motocicleta
Cantidad: 1
Artículo 2: Televisión
Cantidad: 3
Artículo 3: Laptop
Cantidad: 2
1. Registro de artículos
2. Listado de artículos
3. Consulta de existencias
4. Salir
Selecciona una opción:
3
Existencias
Ingrese la descripción del artículo
a consultar:
Laptop
Artículo 3: Laptop
Cantidad: 2
1. Registro de artículos
2. Listado de artículos
3. Consulta de existencias
4. Salir
Selecciona una opción:
4
Salir
```

Se selecciona la opción 4 y detiene el proceso del programa

Link Replit: <https://replit.com/@fernando8881/Inventario#main.swift>

Link Google Drive:

https://drive.google.com/file/d/1RukGuw2mH3bWPjI6N_oUBCkogIOrISLS/view?usp=sharing

Conclusión.

En resumen, la creación de esta aplicación en Replit no solo implica el conocimiento del lenguaje de programación utilizado, sino también una comprensión profunda de la plataforma Replit y sus capacidades, para garantizar un desarrollo fluido y efectivo en el sistema operativo Windows. Además de saber cómo utilizar las plantillas y los proyectos de inicio en Replit para poder acelerar el proceso de desarrollo, proporcionando un punto de partida para la creación de la aplicación, permitiendo la colaboración en línea de los integrantes en el desarrollo del mismo.

¿Qué aprendo?

Que hay muchas áreas de oportunidad para aprender y así poder dominar esta herramienta en el diseño y creación de aplicaciones, que aún cuando es un entorno amigable, las bases de programación se deben de tener bien definidas para poder entender lo que se tiene que hacer al momento de desarrollar y poder solucionar cualquier inconveniente en el proceso de desarrollo y utilizando las herramientas de vanguardia se puede aprender más logrando implementar y desarrollar aplicaciones con errores mínimos al momento de su ejecución.

Enlace Github: <https://github.com/Chifer888/Desarrollo-de-apps-moviles-3.git>

Referencias

ChatGPT. (n.d.). <https://chatgpt.com/c/bf20380e-244c-4a47-92ae-80d87f9626d1>

MoureDev by Brais Moure. (2020, January 10). *XCODE: COMO Crear una APP (para*

Principiantes)  [Tutorial] [Video]. YouTube.

<https://www.youtube.com/watch?v=MyzZnIR5gC4>

Connecting Replit to GitHub / Replit Docs. (2024, May 3). <https://docs.replit.com/replit-workspace/using-git-on-replit/connect-github-to-replit>

ChatGPT. (n.d.). <https://chatgpt.com/>

Videoconferencias, conferencias web, seminario web, uso compartido de pantalla. (n.d.). Zoom.

<https://academiaglobal->

[mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht](https://academiaglobal-mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht)

[yhdTaKUv3Bn8UTZkLfkxh07vyUeU.P76W3pmn8DeRZI8k?canPlayFromShare=true&f](https://academiaglobal-mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht)

[rom=share_recording_detail&continueMode=true&componentName=rec-](https://academiaglobal-mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht)

[play&originRequestUrl=https%3A%2F%2Facademiaglobal-](https://academiaglobal-mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht)

[mx.zoom.us%2Frec%2Fshare%2F_kTZzbYwPp8putzhicBsmucuGVC5df3tBhBaGz5EDf](https://academiaglobal-mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht)

[wK03jhRneq92ssVAX0W-6e.CI9ciPHWG5p7M8eM](https://academiaglobal-mx.zoom.us/rec/play/eNcC9MqNqqRJj45xPwTJ3UwR5eO4l6TfDKQ8iJnPr_aJY_ELfFht)