

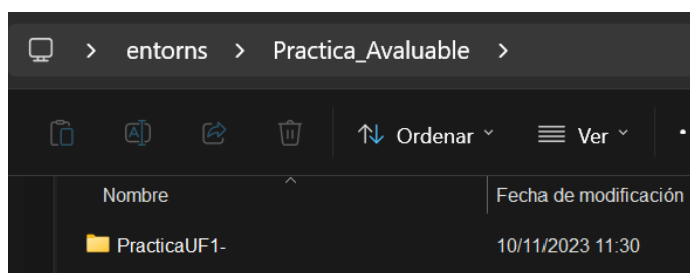
**ACTIVITAT AVALUABLE AC2****Mòdul:** MP05-ENTORNS DE DESENVOLUPAMENT**Alumne:****UF:** UF1 – Desenvolupament de programari**Professor:** Marc Callejón**Data d'entrega:** 17/11/2022**Mètode d'entrega:** clickEdu / GitHub**Resultats de l'aprenentatge:**

1. Reconeix els elements i les eines que intervenen en el desenvolupament d'un programa informàtic, analitzant les seves característiques i les fases en què actuen fins arribar a la seva posada en funcionament.
2. Avalua entorns de desenvolupament integrat analitzant les seves característiques per editar codi font i generar executable.

**Tareas a realizar:****PARTE A (5p)****Tarea 1 (1p)**

Genera un repositorio de GitHub con el nombre PracticaUF1 (1) y sincronízalo con tu carpeta local

```
C:\Users\bea\Desktop\entorns\Practica_Avaluable>git clone https://github.com/beatrizabadj/PracticaUF1-.git
Cloning into 'PracticaUF1-'...
warning: You appear to have cloned an empty repository.
```





## Tarea 2 (2p)

Codifica un programa sencillo en C# y realiza los commits y actualizaciones necesarias en el repositorio remoto con comentarios descriptivos. Explica con tus propias palabras cual es el proceso para llevar a cabo estas acciones.

En la terminal escribimos “dotnet new console” para crear un proyecto nuevo en el directorio “Practica UF1-“, esto creará una estructura de consola en C#, incluyendo archivos Program.cs que contiene el código fuente. Una vez creado el archivo .cs, podremos editarlo en Visual Studio y modificar el código del programa para que ejecute una suma de dos números. A continuación, usaremos los comandos GIT para subir los archivos al repositorio de GitHub:

```
C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1->dotnet new console
La plantilla "Aplicación de consola" se creó correctamente.

Procesando acciones posteriores a la creación...
Ejecutando "dotnet restore" en C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1-\PracticaUF1-.csproj...
Determinando los proyectos que se van a restaurar...
Se ha restaurado C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1-\PracticaUF1-.csproj (en 92 ms).
Restauración realizada correctamente.
```

```
C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1->git status
On branch main
Your branch is based on 'origin/main', but the upstream is gone.
(use "git branch --unset-upstream" to fixup)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Program.cs

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        PracticaUF1-.sln
        bin/
        obj/Debug/

no changes added to commit (use "git add" and/or "git commit -a")
```










```
C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1->git add .



C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1->git commit -m "programa_sumes"
[main 08bda8f] programa_sumes
22 files changed, 122 insertions(+), 3 deletions(-)
create mode 100644 PracticaUF1-.sln
create mode 100644 bin/Debug/net6.0/PracticaUF1-.deps.json
create mode 100644 bin/Debug/net6.0/PracticaUF1-.dll
create mode 100644 bin/Debug/net6.0/PracticaUF1-.exe
create mode 100644 bin/Debug/net6.0/PracticaUF1-.pdb
create mode 100644 bin/Debug/net6.0/PracticaUF1-.runtimeconfig.json
create mode 100644 obj/Debug/net6.0/.NETCoreApp,Version=v6.0.AssemblyAttributes.cs
create mode 100644 obj/Debug/net6.0/PracticaUF1-.AssemblyInfo.cs
create mode 100644 obj/Debug/net6.0/PracticaUF1-.AssemblyInfoInputs.cache
create mode 100644 obj/Debug/net6.0/PracticaUF1-.GeneratedMSBuildEditorConfig.editorconfig
create mode 100644 obj/Debug/net6.0/PracticaUF1-.GlobalUsings.g.cs
create mode 100644 obj/Debug/net6.0/PracticaUF1-.assets.cache
create mode 100644 obj/Debug/net6.0/PracticaUF1-.csproj.AssemblyReference.cache
create mode 100644 obj/Debug/net6.0/PracticaUF1-.csproj.CoreCompileInputs.cache
create mode 100644 obj/Debug/net6.0/PracticaUF1-.csproj.FileListAbsolute.txt
create mode 100644 obj/Debug/net6.0/PracticaUF1-.dll
create mode 100644 obj/Debug/net6.0/PracticaUF1-.genruntimeconfig.cache
create mode 100644 obj/Debug/net6.0/PracticaUF1-.pdb
create mode 100644 obj/Debug/net6.0/apphost.exe
create mode 100644 obj/Debug/net6.0/ref/PracticaUF1-.dll
create mode 100644 obj/Debug/net6.0/refint/PracticaUF1-.dll
```

```
C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1->git push -u origin main
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 8 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (37/37), 85.97 KiB | 4.78 MiB/s, done.
Total 37 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), done.
To https://github.com/beatrizabadj/PracticaUF1-.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1->git pull
Already up to date.
```






	<b>beatrizabadj</b> programa_sumes ...	6 minutes ago  2
	bin/Debug/net6.0	6 minutes ago
	obj	6 minutes ago
	PracticaUF1-.csproj	16 minutes ago
	PracticaUF1-.sln	6 minutes ago
	Program.cs	6 minutes ago



 **beatrizabadij** programa\_sumes 6 minutes ago 

15 lines (15 loc) · 448 Bytes

**Code** Blame

Raw     

```
1 // See https://aka.ms/new-console-template for more information
2 internal class Program
3 {
4     private static void Main(string[] args)
5     {
6         int number1, number2, total;
7         Console.WriteLine("Dona'm número: ");
8         number1 = Convert.ToInt32(Console.ReadLine());
9         Console.WriteLine("Dona'm un altre número: ");
10        number2 = Convert.ToInt32(Console.ReadLine());
11        total = number1 + number2;
12        Console.WriteLine("la suma és " + total);
13        Console.ReadLine();
14    }
15 }
```

### Tarea 3 (2p)

compila el programa y sube el .exe al repositorio

Para compilar el programa tendremos que usar el comando “dotnet build”. Esto es, traduce el código fuente (en este caso de C#) en un lenguaje que pueda entender .NET (código ejecutable). Con el proceso anterior se nos ha subido el archivo .exe.



```
C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1
->dotnet build
MSBuild version 17.3.2+561848881 for .NET
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
PracticaUF1- -> C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1-\bin\Debug\net6.0\PracticaUF1-.dll
```

Compilación correcta.

0 Advertencia(s)

0 Errores

Tiempo transcurrido 00:00:01.29

```
C:\Users\bea\Desktop\entorns\Practica_Avaluable\PracticaUF1
->dotnet run
Dona'm número:
5
Dona'm un altre número:
2
la suma és 7
```

**PracticaUF1-** / bin / Debug / **net6.0** /



**beatrizabadj** programa\_sumes

12 minutes ago



Name	Last commit message	Last commit date
..		
PracticaUF1-.deps.json	programa_sumes	12 minutes ago
PracticaUF1-.dll	programa_sumes	12 minutes ago
PracticaUF1-.exe	programa_sumes	12 minutes ago
PracticaUF1-.pdb	programa_sumes	12 minutes ago
PracticaUF1-.runtimeconfi...	programa_sumes	12 minutes ago

PARTE B (5p)

Tasca 1 (1p)



**Explica que es una metodología ágil (1p) (\*corta-pega no se evaluará)**

Una metodología ágil se refiere a la gestión de proyectos flexible, autónoma y eficaz que se amolda a las condiciones de su entorno de forma inmediata. Esta metodología se ha utilizado sobre todo en el desarrollo de *software*. Algunas de los beneficios que nos ofrece son: la rapidez, la satisfacción inmediata de la clientela, valoración de los y las empleadas, y el *feedback* constante con el cliente.

Algunos ejemplos de metodologías más usadas: Scrum, Kanban, Extreme Programming (XP), Lean Software Development (LSD), Feature Driven Development (FDD), Dynamic Systems Development Method (DSDM), Adaptive Software Development (ASD), Crystal.

**Tasca 2 (2p)**

Define 10 requisitos funcionales (2) y no funcionales (2) de una aplicación web para una tienda en línea que vende productos electrónicos.

La aplicación permitirá a los usuarios navegar por el catálogo de productos, agregar artículos a su carrito de compras, realizar pedidos y gestionar sus cuentas.

FUNCIONALES	NO FUNCIONALES
Servicio de atención al cliente en la página web en caso de no haber ninguna respuesta a un conflicto inesperado	Asegurar permiso de acceso a las cuentas y datos personales para su gestión.
Posibilidad de creación de cuenta asociada a la identidad y la trazabilidad del registro de usuario: mecanismo que permite conocer los cambios realizados en su cuenta. Datos del usuario tales como: número de cuenta, fecha de registro, demografía, fecha de nacimiento, email, saldo... etc.	Privacidad y seguridad de datos de los clientes. El programa deberá ceñirse a las políticas de privacidad y establecer mecanismos que protejan los datos almacenados en las bases de datos.
Programa que realice la conversión de moneda en el proceso de transacción en caso necesario	Presupuesto aproximado en la realización del programa web a través de una consultoría o la propia empresa
Acceso a la base de datos con el inventario	Adaptabilidad de la página web en distintos dispositivos para facilitar su uso y la comodidad de la aplicación ( <i>responsive</i> )



Establecer un cajero para iniciar, cancelar y confirmar las operaciones: el/la cliente debe confirmar la transacción antes de darla por finalizada. En el que antes se verán los productos escogidos, que podrá editar dentro de este. Cada vez que se realice una operación, la base de datos con el inventario se ajustará a los cambios correspondientes.	Velocidad de respuesta y capacidad de procesamiento de la información. Por ejemplo, el tiempo de respuesta del programa debe ser menor de 5 segundos.
Vista de una página web dinámica donde se reflejan los datos de la base de datos relevantes para el cliente, con una imagen de referencia, el código de barras, la descripción del producto y un botón de “añadir al carrito”. Con posibilidad de entrar en el enlace de cada producto para ver más especificaciones.	La aplicación deberá ser intuitiva para que todos los usuarios mayores de 16 años puedan usarla. Se contará de un manual simplificado en el que se enseñará el uso básico de este.
Registrar en una base de datos las operaciones realizadas por los clientes para hacer estudios de mercado y mejorar la productividad de la empresa.	Disponibilidad de la página web al público las 24 horas del día.
El/la cliente puede ver una lista de categorización de productos y filtrarlos en función del precio, marca, más populares, recién llegados, color, material...	Desarrollo de la aplicación de tal forma que no se produzcan discriminaciones para personas con funcionalidad diversa.
En caso de confirmar la operación, se imprimirá un recibo.	Se guardará una copia de seguridad cada semana para asegurar el funcionamiento correcto y su análisis.
Opción de traducción de la página en varios idiomas en función de la demografía o idioma preferente.	Análisis exhaustivo de los posibles errores y fallos del programa.

### Tasca 3 (2p)

Genera un tablero de Trello y coloca los requisitos funcionales como tareas, mueve algunas tareas y explica cómo funciona el tablero.



El tablero permite estructurar y organizar los proyectos de manera visual e interactiva de forma conjunta con otros usuarios. Hemos creado el proyecto “Requisitos funcionales” y hemos rellenado una lista con las tarjetas de los requisitos. La aplicación nos permite mover las tarjetas de modo que puedan clasificarse en función de la lista. Por ejemplo, si hemos completado un proceso, lo movemos a la lista con el nombre “Hecho”. Las mismas tarjetas pueden editarse añadiendo descripciones, fechas, comentarios, entre otros.

