

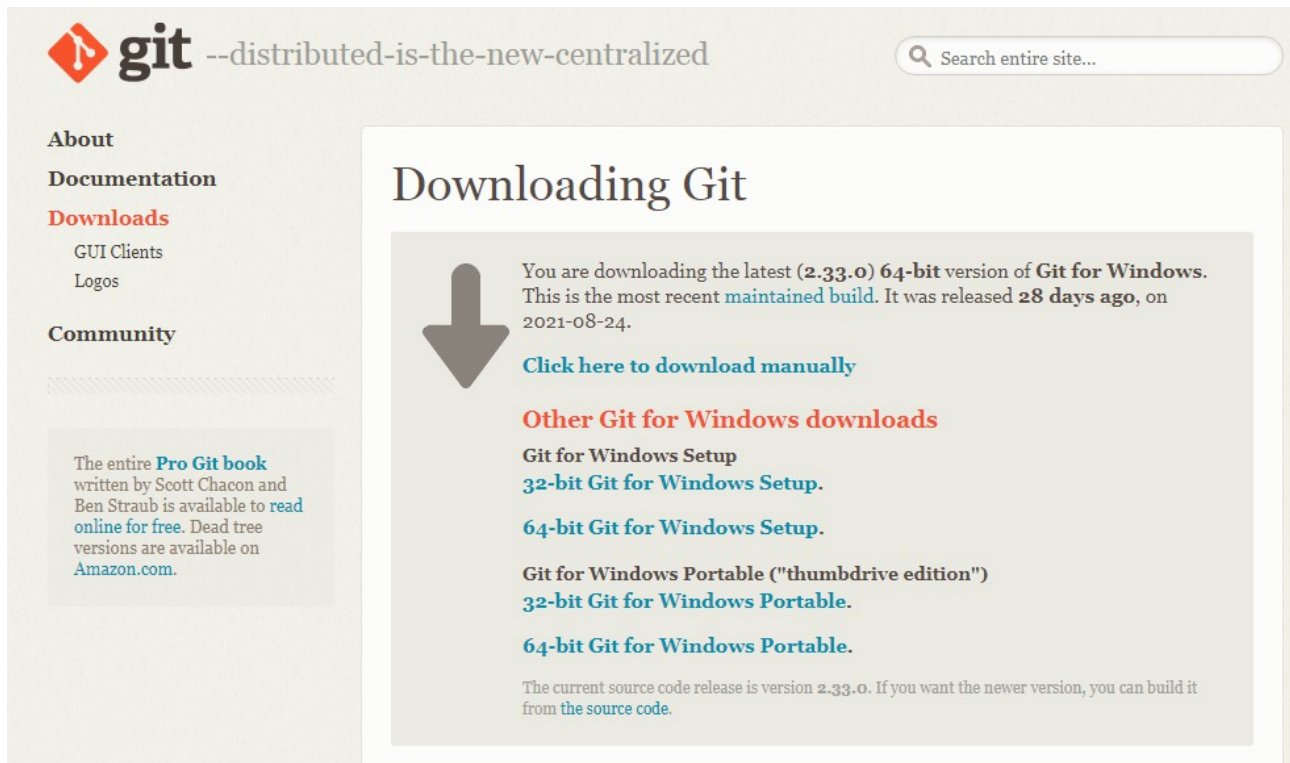
# PRÁCTICA 1 Instalación y uso de Git

- Ciclo formativo: 2º ASIR
- Módulo: GBDLI
- Fecha de entrega de la práctica: 21/09/2021
- Nombre y apellidos: Daniel Muñoz Núñez

## Índice de contenido

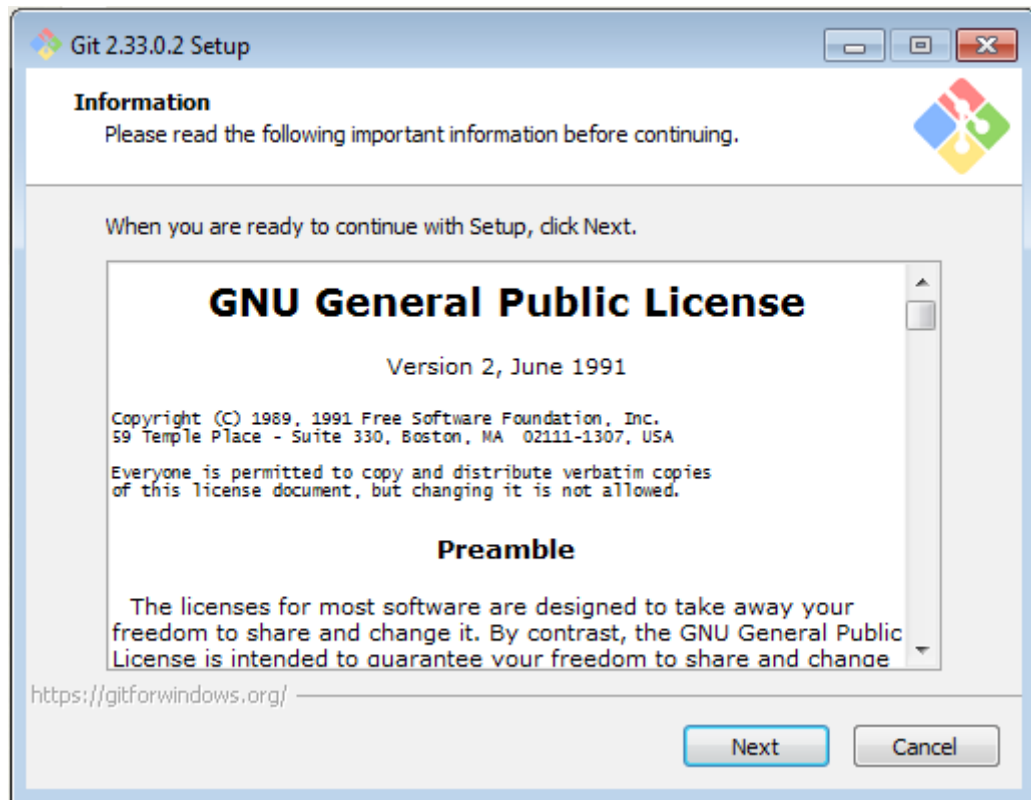
1) Descargarse e instalar Git.....	3
2) Descargar un IDE para desarrollar nuestra aplicación.....	6
3) Crear un repositorio en github y subir archivos desde Visual Studio.....	9
4) Comandos git más importantes.....	17

# 1) Descargarse e instalar Git

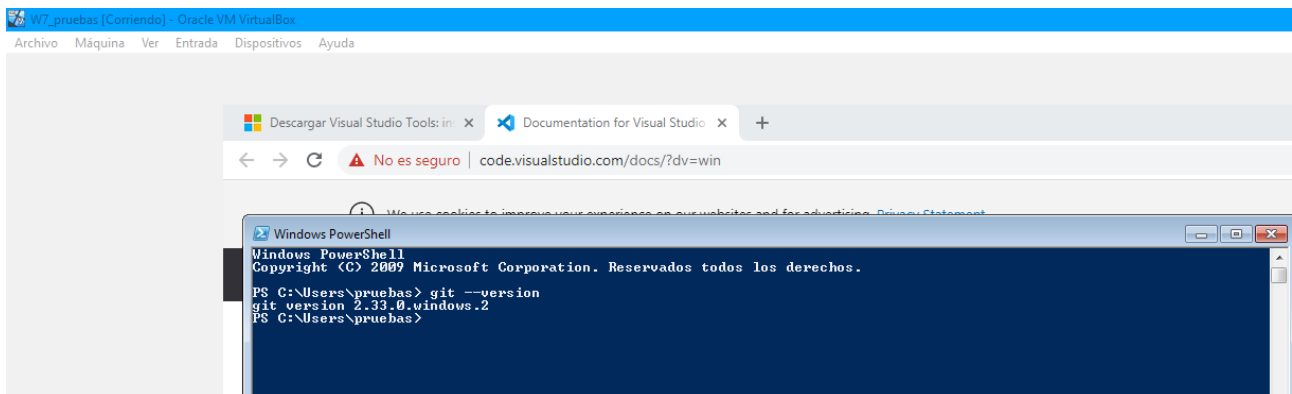


The screenshot shows the Git website's 'Downloading Git' page. The header features the Git logo and the tagline '--distributed-is-the-new-centralized', along with a search bar. The left sidebar contains navigation links: 'About', 'Documentation', 'Downloads' (highlighted), 'GUI Clients', 'Logos', and 'Community'. A box in the sidebar promotes the 'Pro Git book'. The main content area is titled 'Downloading Git' and includes a large downward arrow icon. The text states: 'You are downloading the latest (2.33.0) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 28 days ago, on 2021-08-24.' Below this, there is a link to 'Click here to download manually'. A section titled 'Other Git for Windows downloads' lists: 'Git for Windows Setup', '32-bit Git for Windows Setup.', '64-bit Git for Windows Setup.', 'Git for Windows Portable ("thumbdrive edition")', '32-bit Git for Windows Portable.', and '64-bit Git for Windows Portable.'. At the bottom, it notes: 'The current source code release is version 2.33.0. If you want the newer version, you can build it from the source code.'

En primer lugar nos vamos a la pagina web oficial de git y nos lo descargamos en nuestro ordenador.

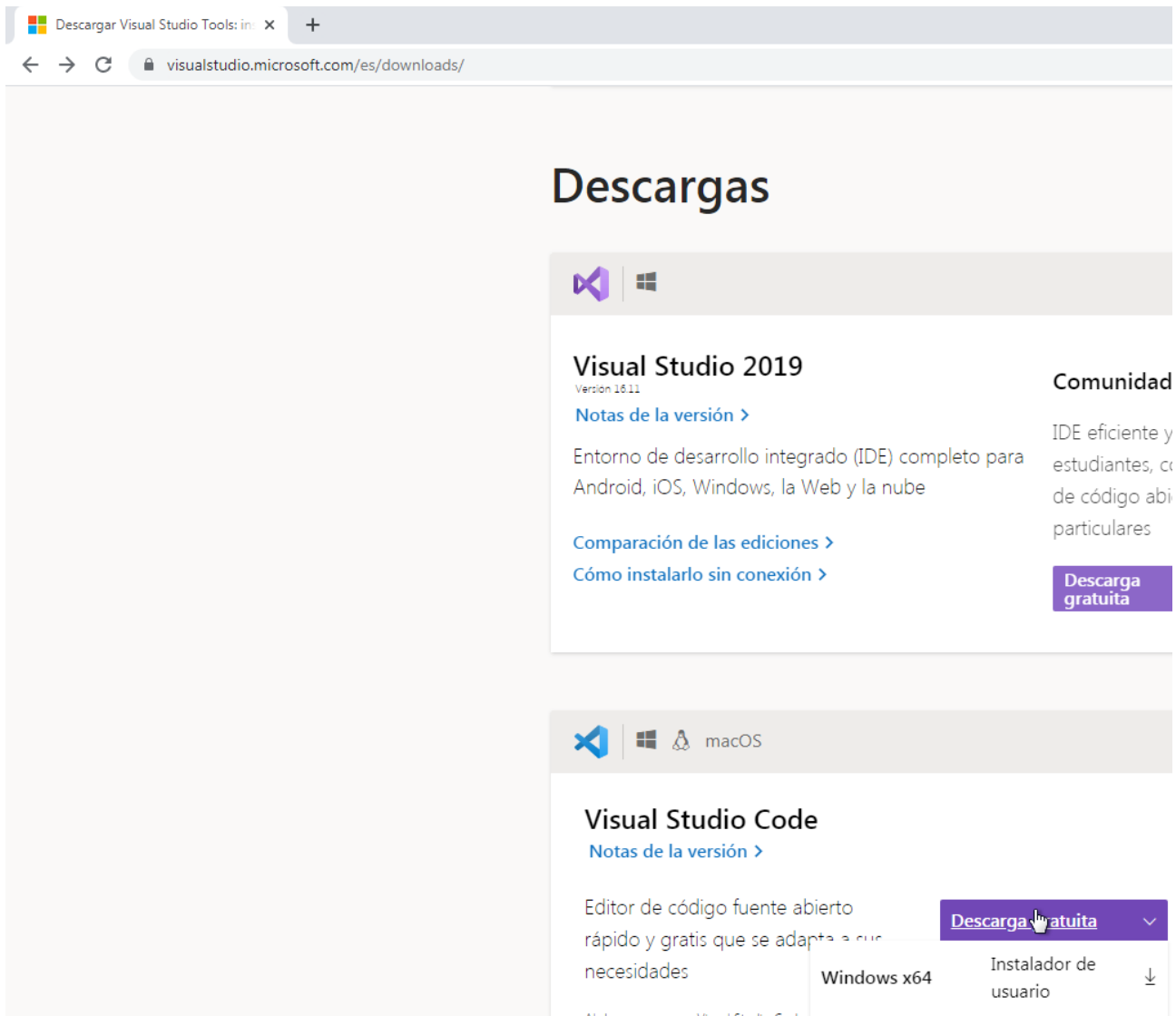


Una vez descargado el .exe lo ejecutamos y nos debería aparecer el asistente de instalación, desde aquí lo dejaremos todo por defecto.

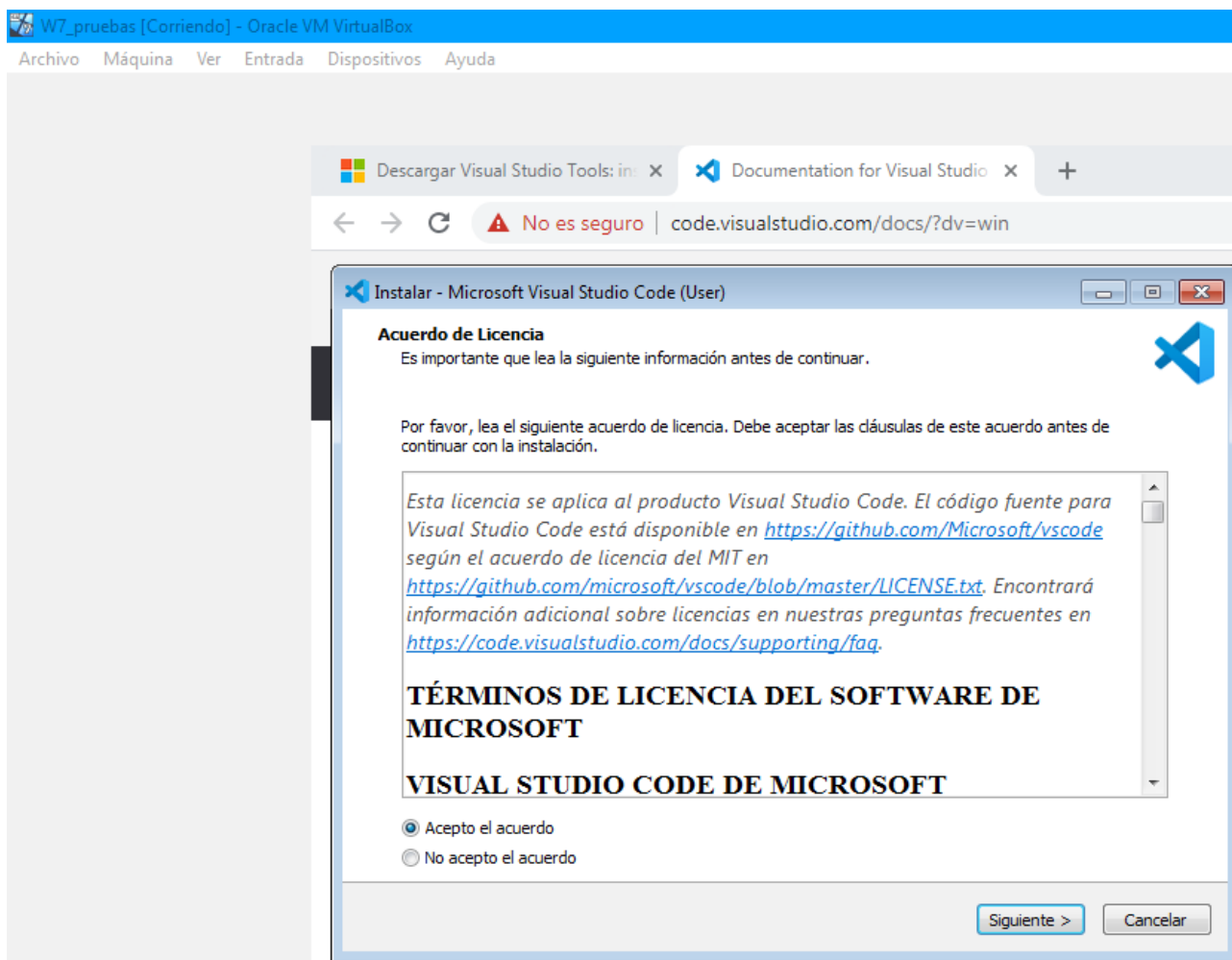


Para comprobar que se ha instalado correctamente ejecutamos en powershell `git --version` y nos mostrará la versión del programa.

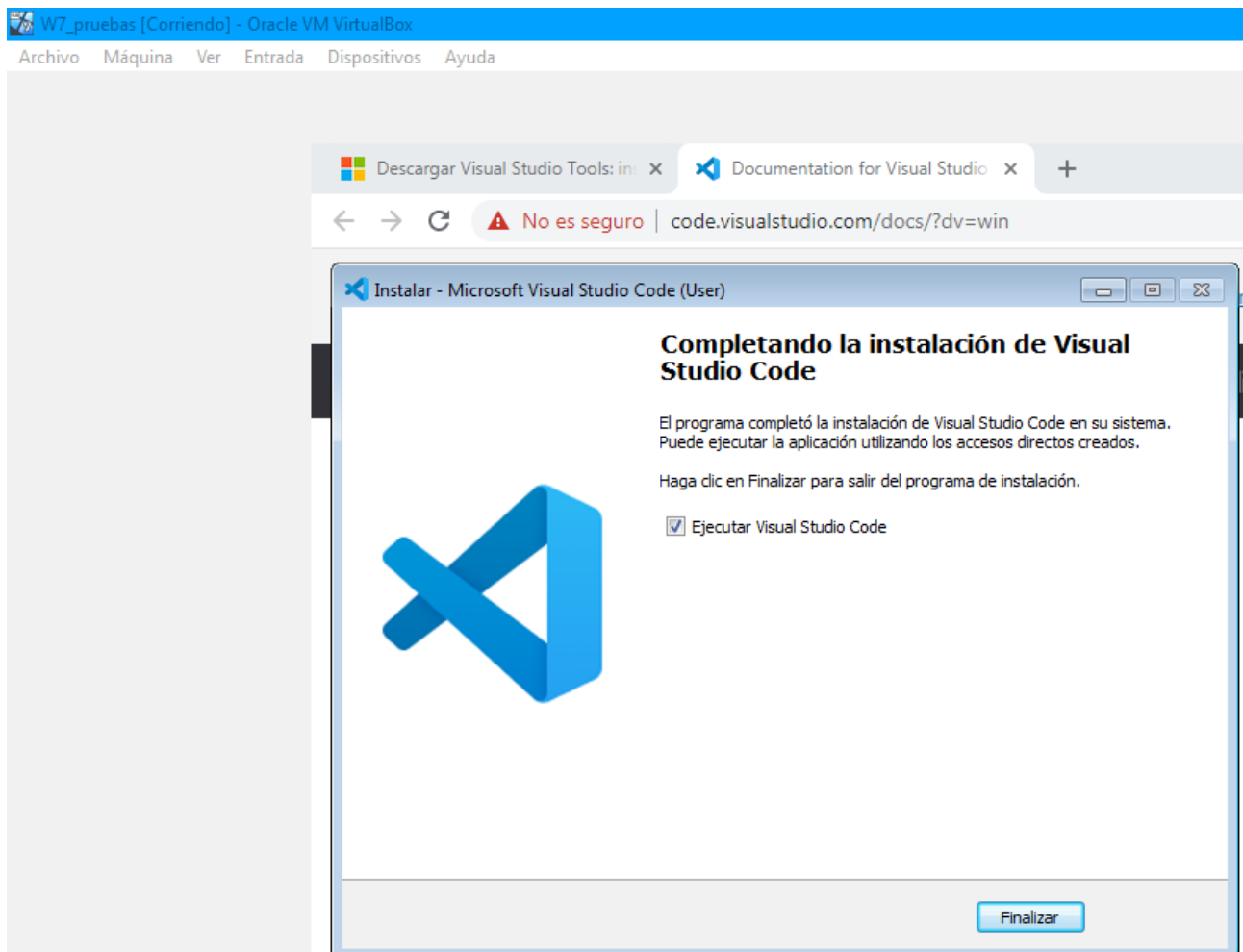
## 2) Descargar un IDE para desarrollar nuestra aplicación



Para utilizar git debemos descargarnos un IDE para empezar a desarrollar nuestro código, esto no es mas que un editor de texto con utilidades adicionales para ayudarnos a redactar código, en esta caso vamos a intalar visual studio code, para ello nos vamos a su página web y nos lo descargamos.



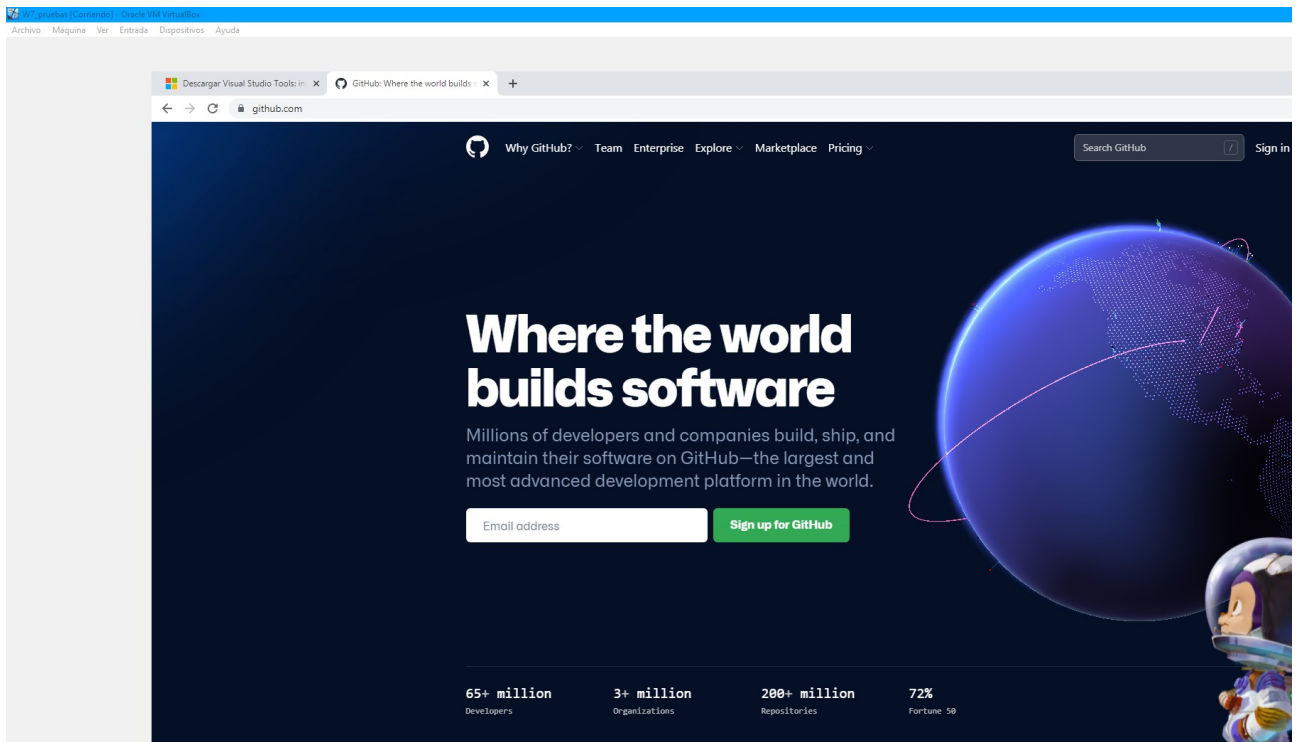
Una vez descargado dejamos todo por defecto y continuamos el asistente.



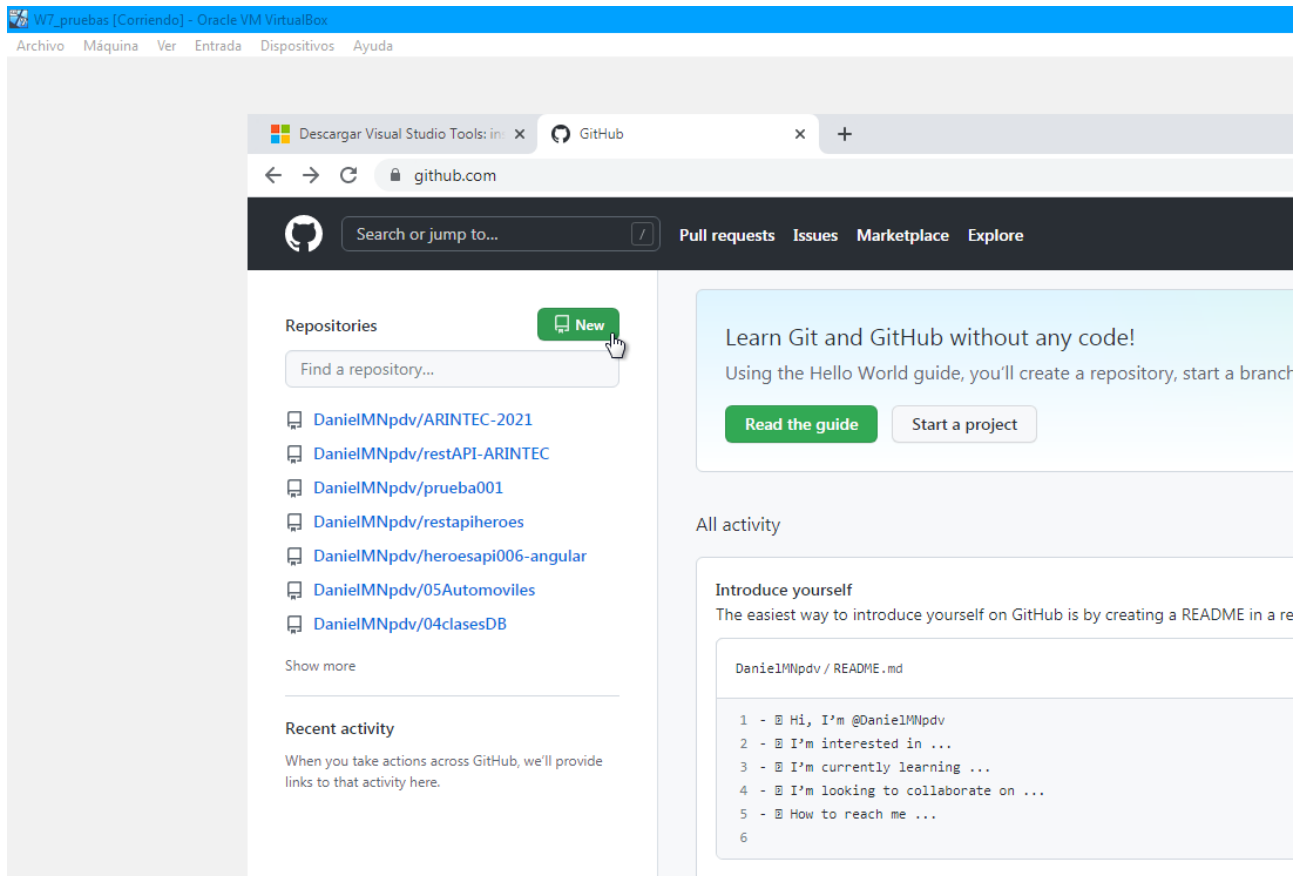
Una vez que hemos terminado de instalarlo lo ejecutamos.



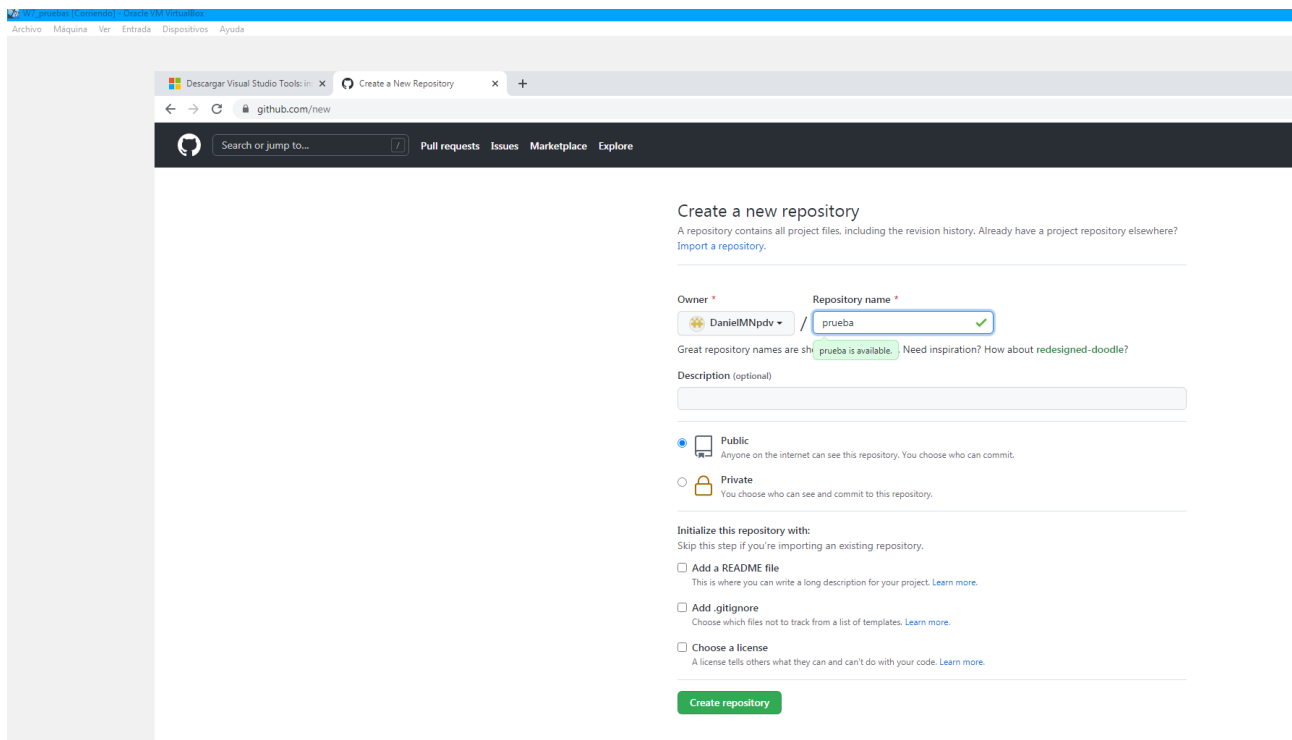
### 3) Crear un repositorio en github y subir archivos desde Visual Studio.



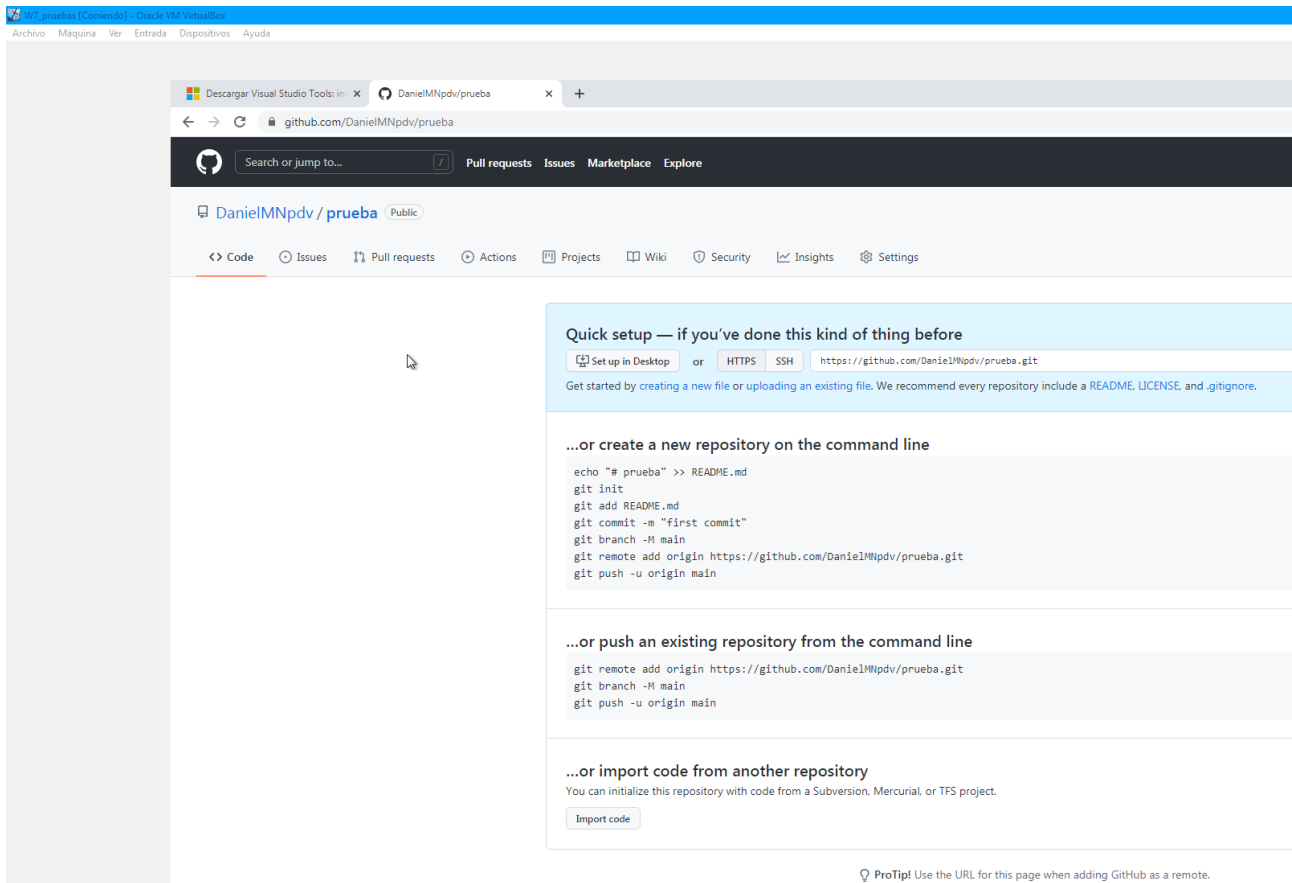
En primer lugar nos vamos a la página oficial de github e iniciamos sesión en nuestra cuenta.



Posteriormente creamos un nuevo repositorio haciendo click en new.



Luego nos aparecerá una ventana donde debemos elegir el nombre de nuestro repositorio.



Una vez creado el repositorio nos deberá aparecer una pequeña explicación con los comandos que debemos introducir en la consola de nuestro ordenador local para subir el código.

```
W7_pruebas [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
File Edit Selection View Go Run Terminal Help
README.md - prueba - Visual Studio Code

EXPLORER
PRUEBA
├── doc
├── src
├── prueba1.txt
├── prueba2.txt
├── .gitignore
└── README.md

README.md
1 //Para subirlo a GitHub:
2 git init
3 git add .
4 git commit -m "primera subida"
5 git remote add origin https://github.com/DanielMNpdv/
6 git push -u origin master
7
8 //Otros Comandos git
9 git clone (para descargar el repositorio desde la nube)
10 git pull (fusiona todos los cambios que se han hecho en el repositorio remoto con el local)
11 git checkout ()
12 git remote -v
13 git remote add origin <host-or-remoteURL>
14 git branch
15 git merge <branch-name>
16
17 //Comandos a introducir para empezar a trabajar (-g es global y se hace 1 sola vez):
18
19 npm init --yes
20 npm install -g typescript
21 npm install tsc-init -g
22 npm install mongoose
23 npm install @types/mongoose
24 npm i @types/node@14.14.6
25 npx tsc -w

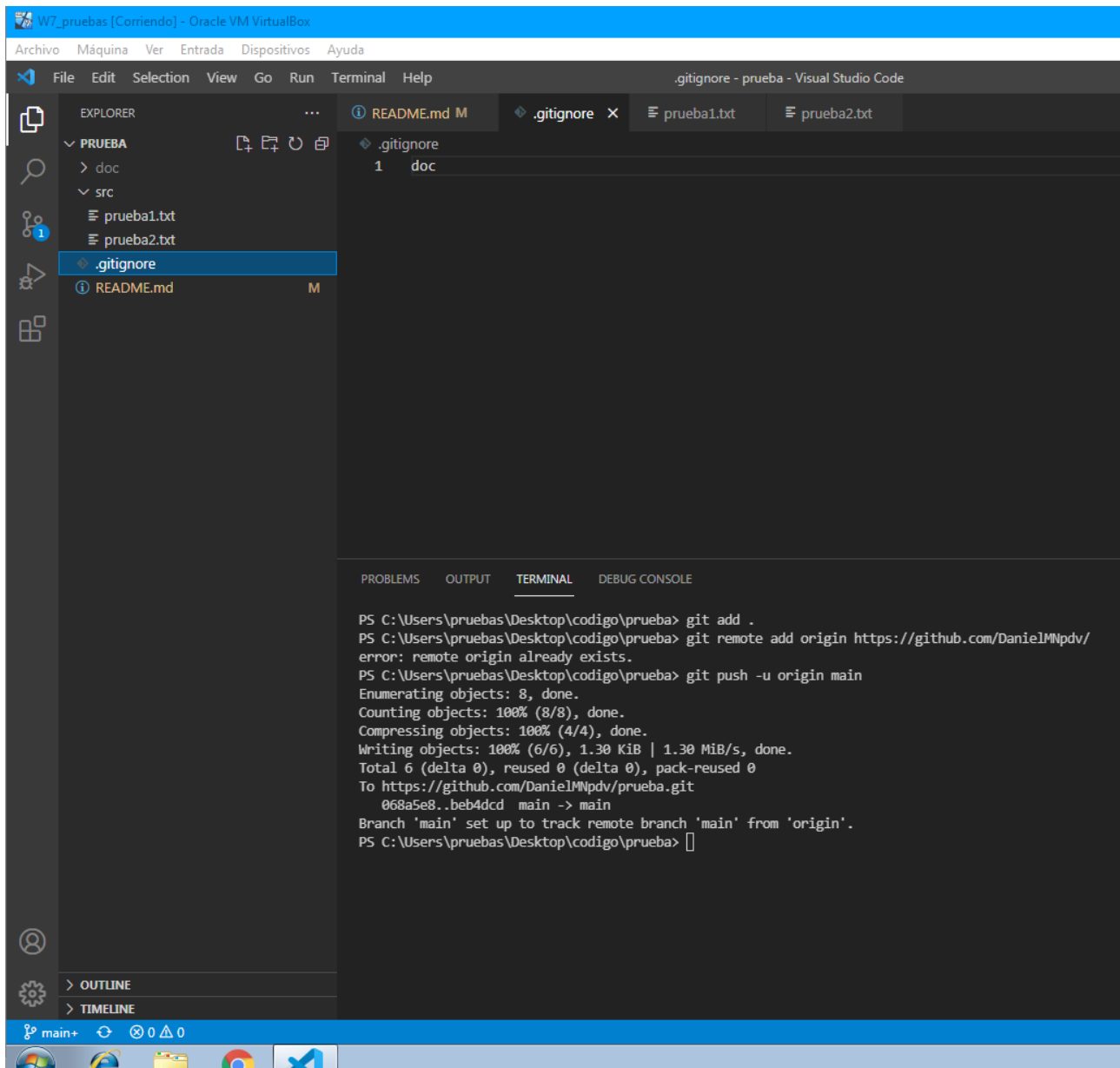
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\pruebas\Desktop\codigo\prueba> git init
Reinitialized existing Git repository in C:/Users/pruebas/Desktop/codigo/prueba/.git/
PS C:\Users\pruebas\Desktop\codigo\prueba> git add .
PS C:\Users\pruebas\Desktop\codigo\prueba> git commit -m "Primera subida"
[main beb4dcd] Primera subida
4 files changed, 64 insertions(+), 1 deletion(-)
create mode 100644 .gitignore
rewrite README.md (100%)
create mode 100644 src/prueba1.txt
create mode 100644 src/prueba2.txt
PS C:\Users\pruebas\Desktop\codigo\prueba>
```

Ahora creamos una carpeta con el nombre prueba y dentro los siguientes archivos y carpetas, también ejecutaremos los comandos “git init” “git add .” y “git commit -m "primera subida"”. Debemos mencionar que la primera vez que ejecutemos commit nos dará error debido a que no hemos establecido ningún correo ni ningún nombre de usuario, para ello solo tendremos que ejecutar “git config –global user.name “tu nombre”” y “git config –global user.email “tu correo””.

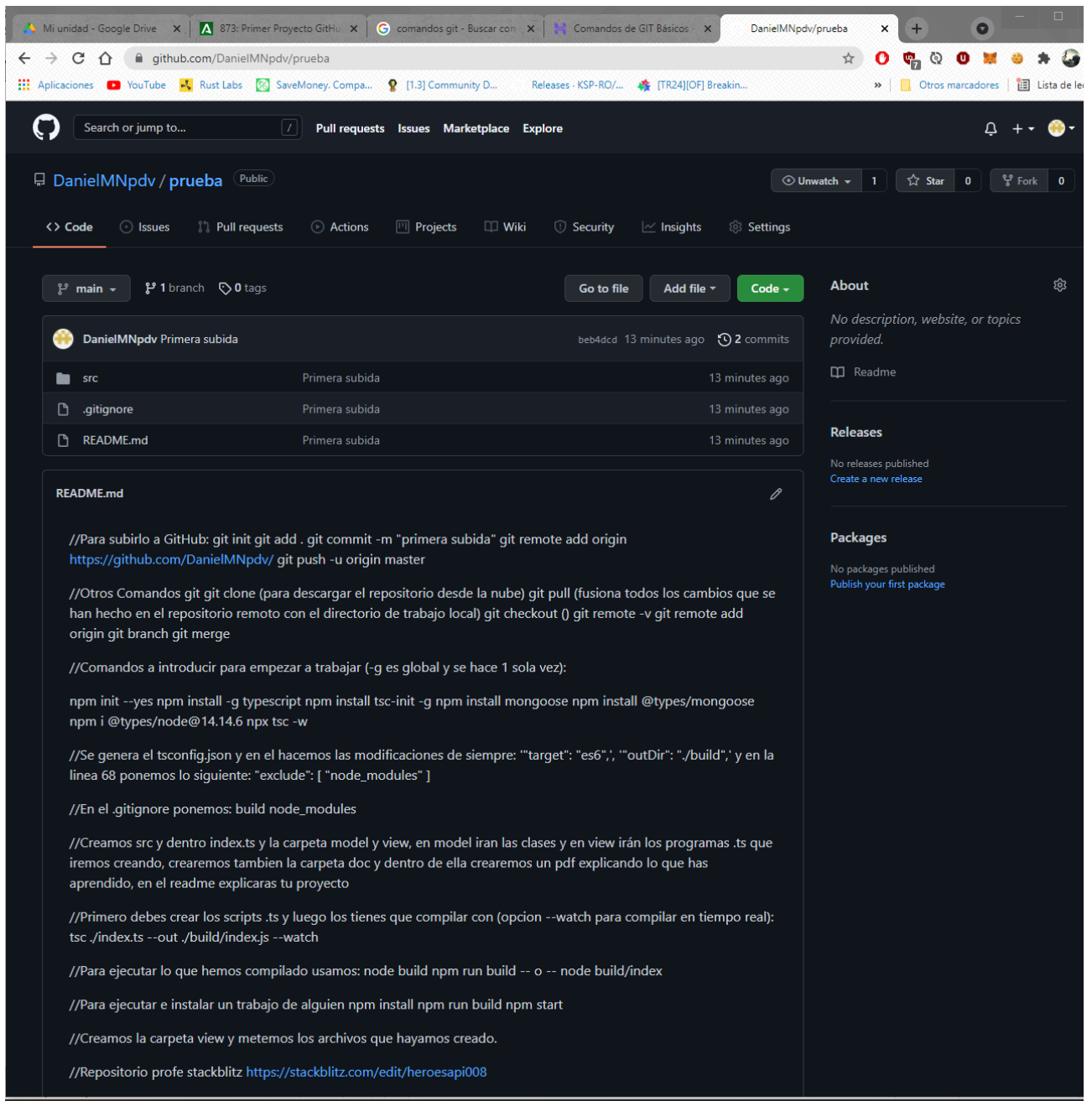
```
1 //Para subirlo a GitHub:
2 git init
3 git add .
4 git commit -m "primera subida"
5 git remote add origin https://github.com/DanielMNpdv/
6 git push -u origin main
7
8 //Otros Comandos git
9 git clone (para descargar el repositorio desde la nube)
10 git pull (fusiona todos los cambios que se han hecho en el repositorio remoto con el direc
11 git checkout ()
12 git remote -v
13 git remote add origin <host-or-remoteURL>
14 git branch
15 git merge <branch-name>
16
17 //Comandos a introducir para empezar a trabajar (-g es global y se hace 1 sola vez):
18
19 npm init --yes
20 npm install -g tuiocssint
```

```
PS C:\Users\pruebas\Desktop\codigo\prueba> git add .
PS C:\Users\pruebas\Desktop\codigo\prueba> git remote add origin https://github.com/DanielMNpdv/
error: remote origin already exists.
PS C:\Users\pruebas\Desktop\codigo\prueba> git push -u origin main
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 1.30 KiB | 1.30 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/DanielMNpdv/prueba.git
068a5e8..beb4dcd main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
PS C:\Users\pruebas\Desktop\codigo\prueba>
```

Posteriormente para subirlos a nuestro repositorio solo tendremos que ejecutar “git remote add origin <tu repositorio>”, para enlazar el repositorio local con el que tenemos en la nube, y “git push -u origin main”.



Por último tenemos que destacar el archivo `.gitignore`, que sirve para indicar los directorios que no se subirán a github, en nuestro caso “doc”.



Ahora nos vamos a nuestro repositorio de github en la nube y podemos observar que se ha subido correctamente todo menos la carpeta doc.



## 4) Comandos git más importantes

Aparte de los comandos anteriormente mencionados existen muchos otros con diversos fines:

- git pull (fusiona todos los cambios que se han hecho en el repositorio remoto con el directorio de trabajo local en la rama de trabajo actual)
- git remote -v (te permite ver todos los repositorios remotos)
- git branch (muestra un listado de todas las ramas disponibles en tu proyecto)
- git status (muestra la lista de los archivos que se han cambiado junto con los archivos que están por ser preparados o confirmados)
- git clone <url de tu repositorio>(para descargar el repositorio remoto)

The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a file tree with folders like '1er TRIM' and files like 'Prueba01\prueba'. The main editor area displays a 'README.md' file with the following content:

```
1er TRIM > Prueba01 > prueba > 1 README.md
1 //Para subirlo a GitHub:
2 git init
3 git add .
4 git commit -m "primera subida"
5 git remote add origin https://github.com/DanielMNpdv/
6 git push -u origin master
7
8 //Otros Comandos git
9 git clone (para descargar el repositorio desde la nube)
10 git pull (fusiona todos los cambios que se han hecho en el repositorio remoto con el
    directorio de trabajo local)
```

Below the editor, the TERMINAL pane shows the execution of these commands in a PowerShell session:

```
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS F:\Daniel\GBDLC> cd '.\1er TRIM\Prueba01\'
PS F:\Daniel\GBDLC\1er TRIM\Prueba01> git init
Initialized empty Git repository in F:\Daniel\GBDLC\1er TRIM\Prueba01\.git/
PS F:\Daniel\GBDLC\1er TRIM\Prueba01> dir
PS F:\Daniel\GBDLC\1er TRIM\Prueba01> git clone DanielMNpdv@host:/DanielMNpdv/prueba
Cloning into 'prueba'...
ssh: Could not resolve hostname host: Name or service not known
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
PS F:\Daniel\GBDLC\1er TRIM\Prueba01> git clone https://github.com/DanielMNpdv/prueba.git
Cloning into 'prueba'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0
Receiving objects: 100% (9/9), done.
PS F:\Daniel\GBDLC\1er TRIM\Prueba01> dir

Directorio: F:\Daniel\GBDLC\1er TRIM\Prueba01

Mode                LastWriteTime         Length Name
----                -
d-----          23/09/2021    21:50             prueba

PS F:\Daniel\GBDLC\1er TRIM\Prueba01>
```

The status bar at the bottom indicates the current branch is 'master' and the file encoding is UTF-8.

Para utilizar correctamente el comando git clone primero debemos inicializar el repositorio vacío con git init y luego ya podremos clonarlo, esto nos descargará el repositorio que tenemos en la nube en nuestra carpeta local.

github.com/DanielMNpdv/prueba

Search or jump to... Pull requests Issues Marketplace Explore

DanielMNpdv / prueba Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

DanielMNpdv subida doc ee47c7e 5 minutes ago 4 commits

doc	subida doc	5 minutes ago
src	Primera subida	2 hours ago
.gitignore	subida doc	5 minutes ago
README.md	subida doc	5 minutes ago

README.md

```
//Para subirlo a GitHub: git init git add . git commit -m "subida doc" git remote add origin
https://github.com/DanielMNpdv/ git push -u origin main

//Otros Comandos git clone (para descargar el repositorio desde la nube) git pull (fusiona todos los cambios que se
han hecho en el repositorio remoto con el directorio de trabajo local) git checkout () git remote -v git remote add
origin git branch git merge

//Comandos a introducir para empezar a trabajar (-g es global y se hace 1 sola vez):

npm init --yes npm install -g typescript npm install tsc-init -g npm install mongoose npm install @types/mongoose
npm i @types/node@14.14.6 npx tsc -w

//Se genera el tsconfig.json y en el hacemos las modificaciones de siempre: "target": "es6", "outDir": "./build", y en la
linea 68 ponemos lo siguiente: "exclude": [ "node_modules" ]

//En el .gitignore ponemos: build node_modules

//Creamos src y dentro index.ts y la carpeta model y view, en model iran las clases y en view irán los programas .ts que
iremos creando, crearemos tambien la carpeta doc y dentro de ella crearemos un pdf explicando lo que has
aprendido, en el readme explicaras tu proyecto

//Primero debes crear los scripts .ts y luego los tienes que compilar con (opcion --watch para compilar en tiempo real):
tsc ./index.ts --out ./build/index.js --watch

//Para ejecutar lo que hemos compilado usamos: node build npm run build -- o -- node build/index

//Para ejecutar e instalar un trabajo de alguien npm install npm run build npm start

//Creamos la carpeta view y metemos los archivos que hayamos creado.

//Repositorio profe stackblitz https://stackblitz.com/edit/heroesapi008

//Repositorio profe Github https://github.com/sgbdpdv2021?tab=repositories

//Para conectar desde mongo compass mongodb+srv://usuario1:usuario1@cluster0.49asp.mongodb.net/test
```

Por último vamos a explicar el uso del comando git pull, en este caso he creado una carpeta nueva en mi repositorio remoto llamada doc.

The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left displays a project structure with a folder named 'PRUEBA' containing a 'src' subfolder. Inside 'src', there are two files: 'prueba1.txt' and 'prueba2.txt', both marked with a green 'M' icon indicating they are modified. Other files in the project include '.gitignore' and 'README.md'. The main editor area shows the 'prueba1.txt' file with the text 'prueba1' on the first line. Below the editor, the TERMINAL panel is active, displaying the output of a 'git commit' and 'git push' command. The commit message is 'subida pruebas'. The push command fails with a 'rejected' error because the remote repository contains work that is not present in the local repository. The terminal output includes the following text:

```
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

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:   src/prueba1.txt
        modified:   src/prueba2.txt

PS C:\Users\pruebas\Desktop\codigo\prueba> git commit -m "subida pruebas"
[main 12a9d5f] subida pruebas
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\pruebas\Desktop\codigo\prueba> git push -u origin main
To https://github.com/DanielMNpdv/prueba.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/DanielMNpdv/prueba.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

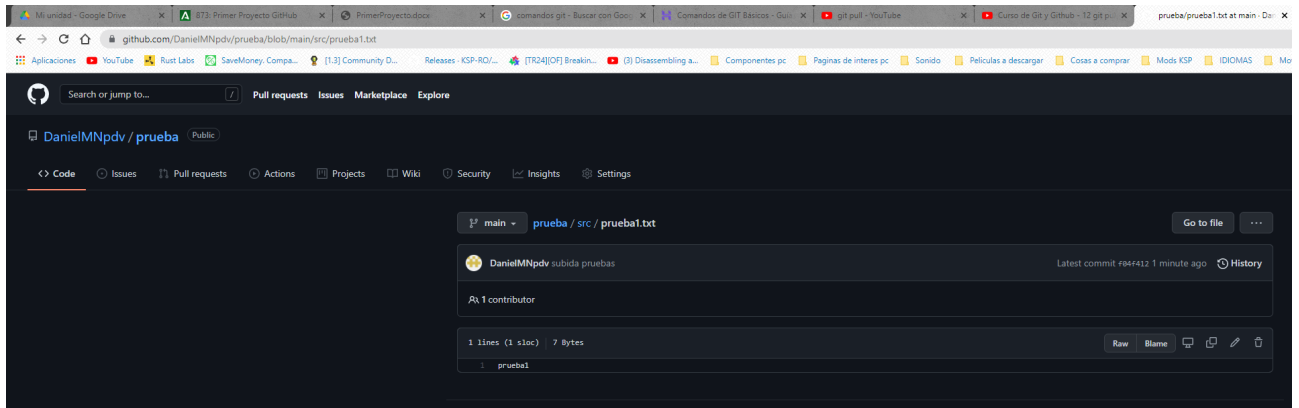
The status bar at the bottom of the window shows 'main' as the current branch, with 0 commits ahead and 0 behind. The bottom of the screen shows the Windows taskbar with icons for the Start menu, File Explorer, Google Chrome, and Visual Studio Code.

Al mismo tiempo he modificado los archivos prueba1.txt y prueba2.txt, pero como podemos observar no podemos subir las modificaciones, esto es debido a que el contenido del repositorio remoto tiene modificaciones que no se han hecho desde este repositorio local (carpeta doc).

The screenshot shows the Visual Studio Code interface within an Oracle VM VirtualBox window titled 'W7\_pruebas [Corriendo]'. The Explorer sidebar on the left shows a project structure with a 'PRUEBA' folder containing 'doc' and 'src' subfolders. The 'src' folder contains 'prueba1.txt', 'prueba2.txt', and '.gitignore'. The 'doc' folder contains 'Comandos\_github.pdf'. The Explorer sidebar also shows 'README.md' and '.gitignore' at the root level. The main editor area shows the 'prueba1.txt' file with the content '1 prueba1'. The Terminal panel at the bottom shows the output of a 'git pull' command. The output indicates that the local repository is up-to-date with the remote repository. The terminal text is as follows:

```
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
PS C:\Users\pruebas\Desktop\codigo\prueba> git pull
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 1), reused 8 (delta 1), pack-reused 0
Unpacking objects: 100% (8/8), 982.93 KiB | 3.37 MiB/s, done.
From https://github.com/DanielMNpdv/prueba
   beb4dcd..ee47c7e  main      -> origin/main
Auto-merging README.md
Merge made by the 'recursive' strategy.
 .gitignore      | 2 +-
 README.md       | 2 +-
 doc/Comandos_github.pdf | Bin 0 -> 1134548 bytes
3 files changed, 2 insertions(+), 2 deletions(-)
create mode 100644 doc/Comandos_github.pdf
PS C:\Users\pruebas\Desktop\codigo\prueba> git push -u origin main
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 566 bytes | 566.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DanielMNpdv/prueba.git
   ee47c7e..7042fa3  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
PS C:\Users\pruebas\Desktop\codigo\prueba> git pull
```

Para poder subir los cambios hechos a los archivos .txt tenemos que hacer un git pull para sincronizar ambos directorios.



Por último aquí podemos ver que los cambios se han subido correctamente.