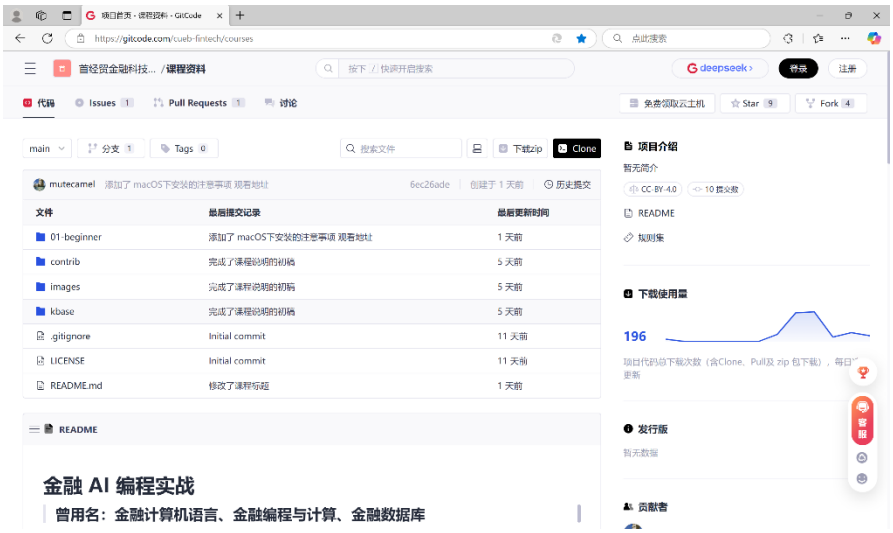
金融计算机语言第一周作业

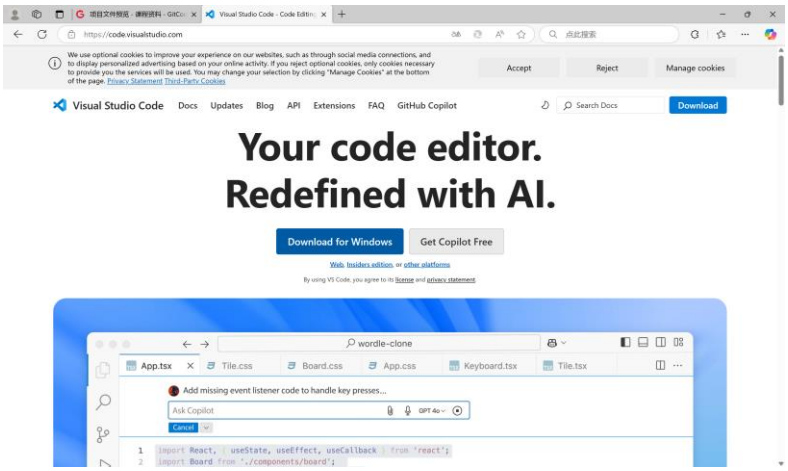
郭芳瑾　22 级金融工程班　32022130022

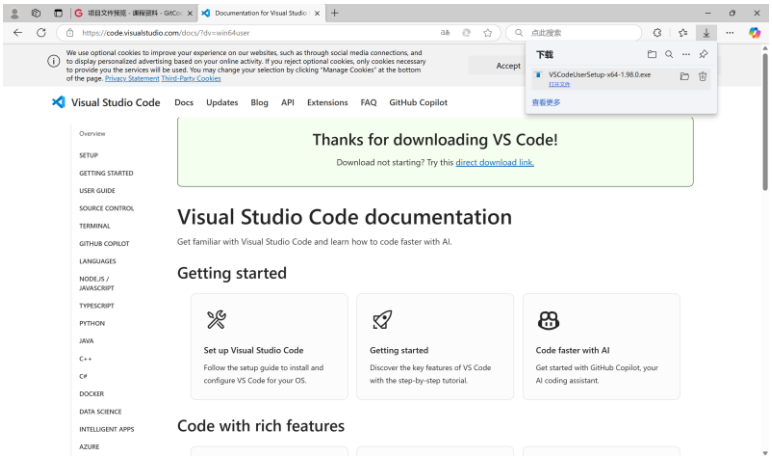
专题一 准备开发环境

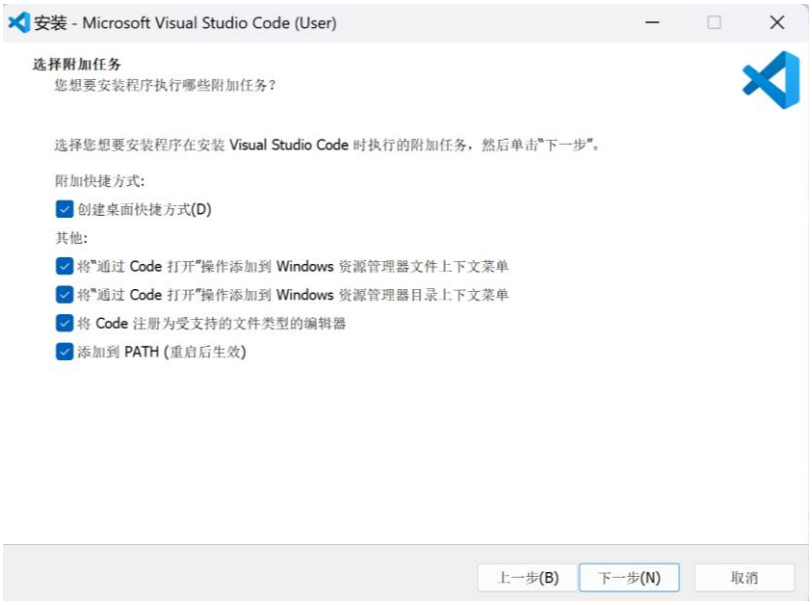


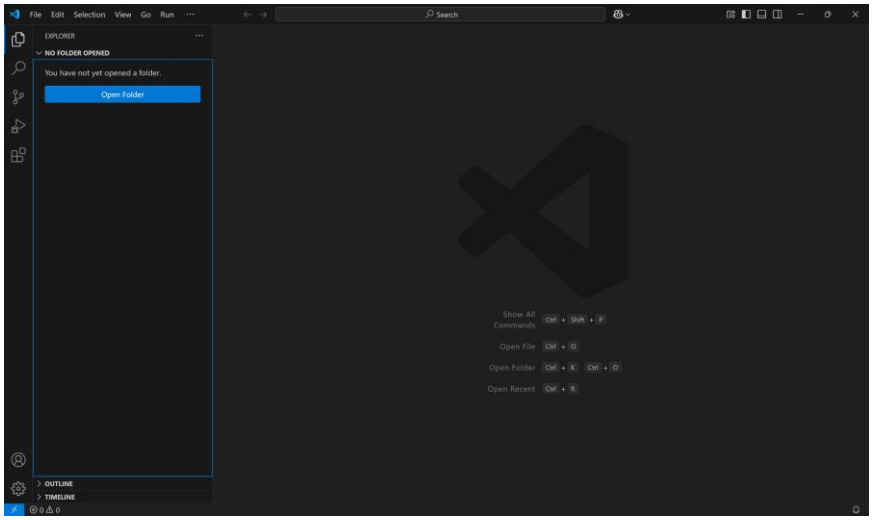
找到 http://gitcode.com/cueb-fintech/courses 网站，并收藏。

1. 安装 VS Code（代码编辑器）



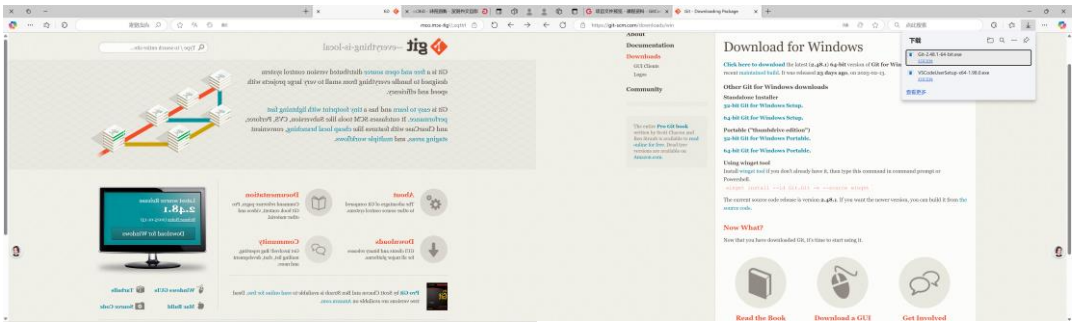




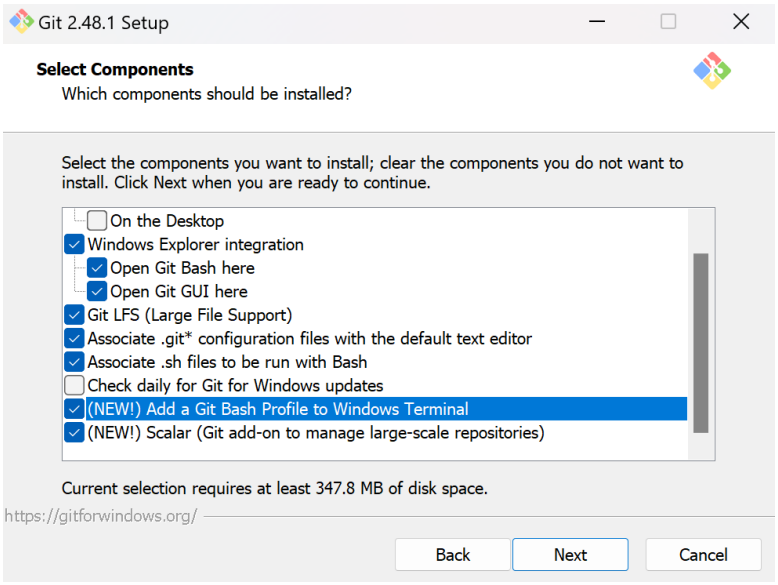


按操作步骤完成 VS Code 的安装，并注册了用户。

2. 安装 Git（代码版本管理软件）

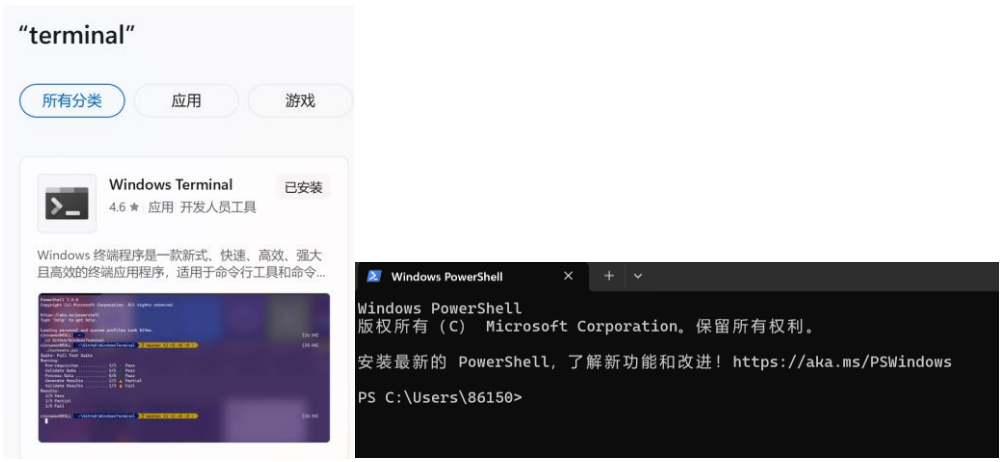


Standalone Installer – self contain; Portable -绿色版本（便携，适安装在 U 盘上）

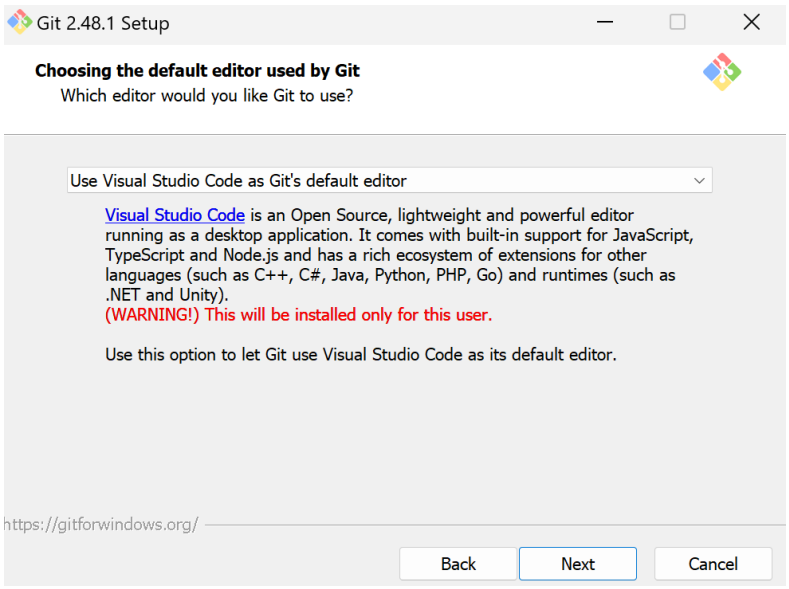


p.s. Windows explorer 资源管理器

- **Windows 终端—Terminal**

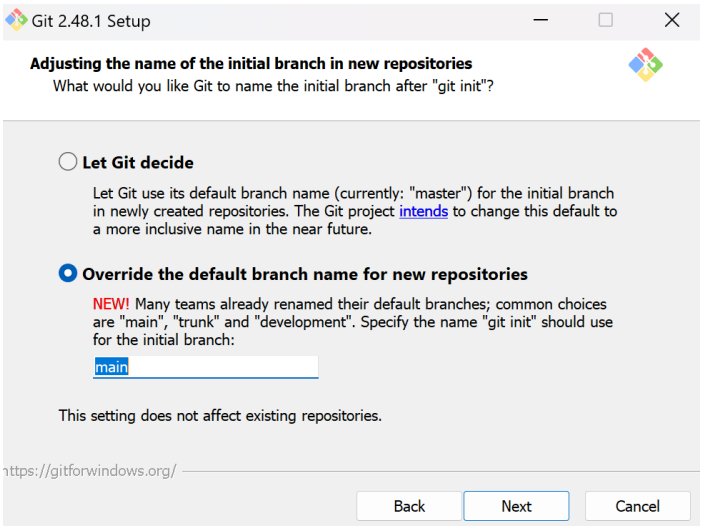


- **编辑器 editor**（Vim 好用，但是难，可学）



- **Git 主分支：**master **；　初始分支 initial branch**

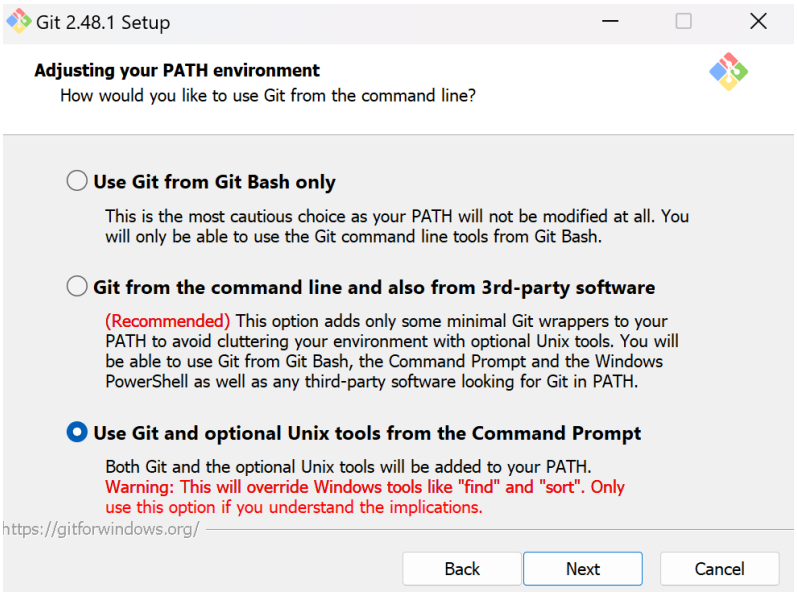
版本控制系统允许有多分支，不同分支状态和用途等不同（修复 bug，增加功能，成熟稳定生产），分支可以切换，还可以合并。在各分支任务完成后，可以将其合并到主分支中，形成一个无缺陷且具有新功能的主分支。



使用新仓库 main

- **环境变量 PATH environment**

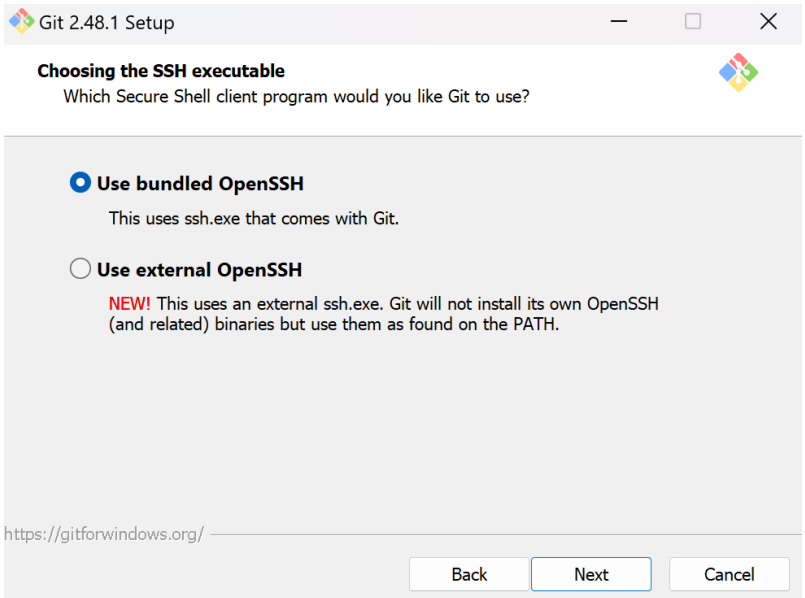
解决终端中的命令从哪里找到的问题，添加存放命令的路径。



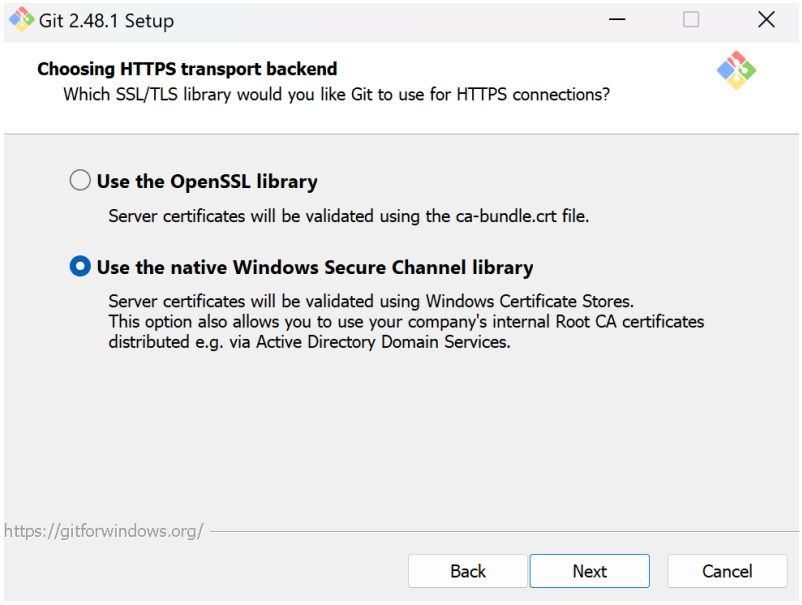
Unix tool 命令行工具（自动化）

- **SSH**（互联网加密通讯的工具）

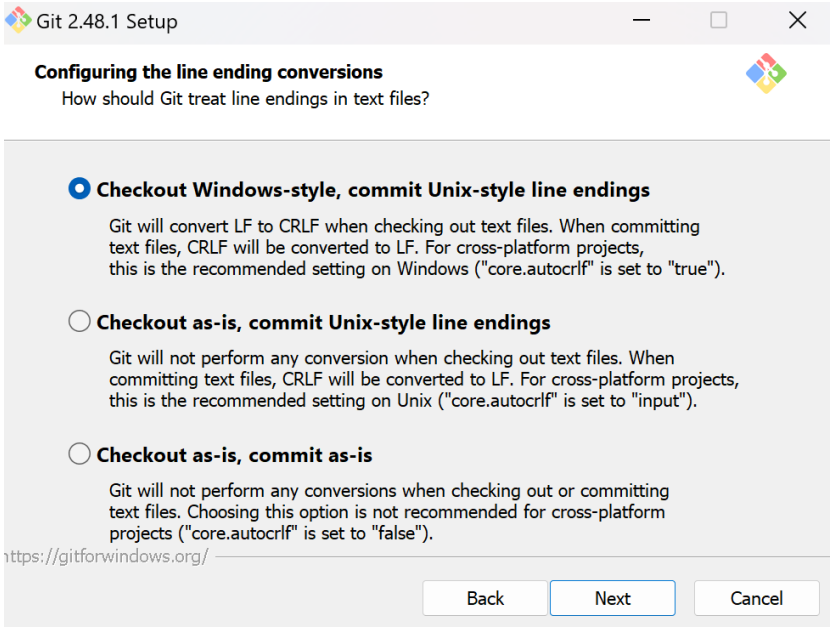
从本地 Git 仓库与 Gitcode 平台惊醒通讯时使用，在进行代码协作的沟通过程中，需要对代码进行加密。



- **通讯后端 HTTPS transport backend**

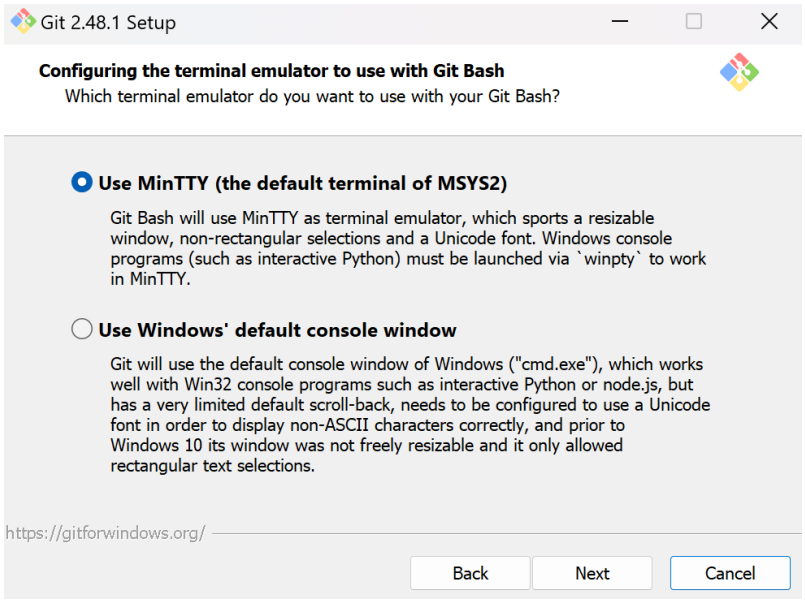


- **文本换行 line ending conversions** (避免协作时的格式问题)

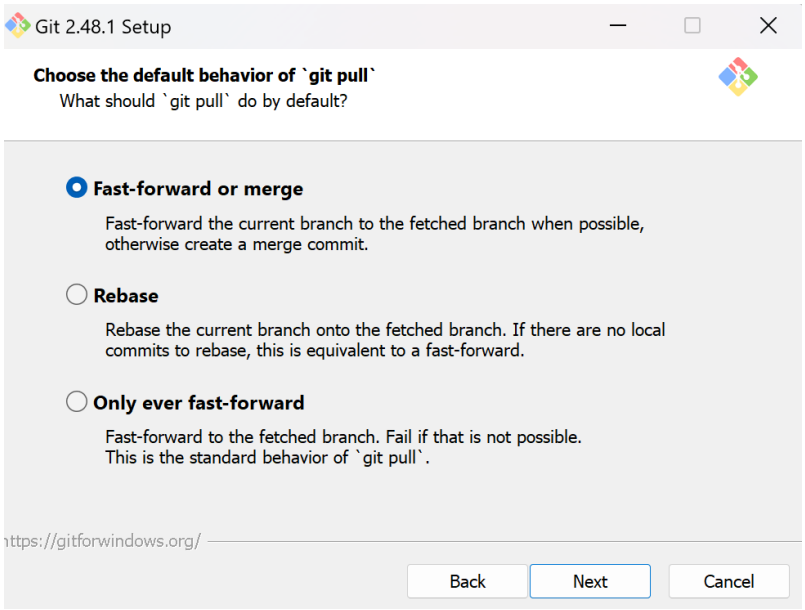


- **配置终端模拟器**

MinTTY：可以很好的模拟 unix 终端，符合跨平台标准。

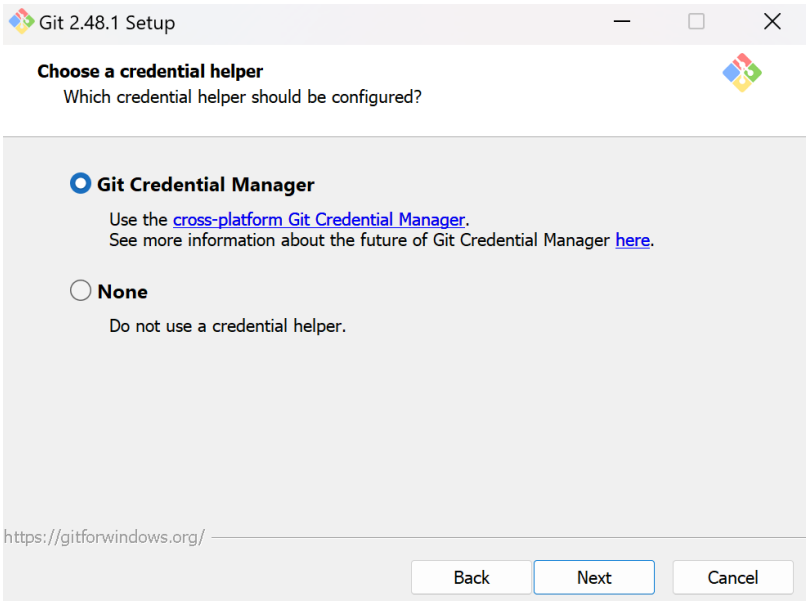


- **拉取代码时的合并方式’git pull’**

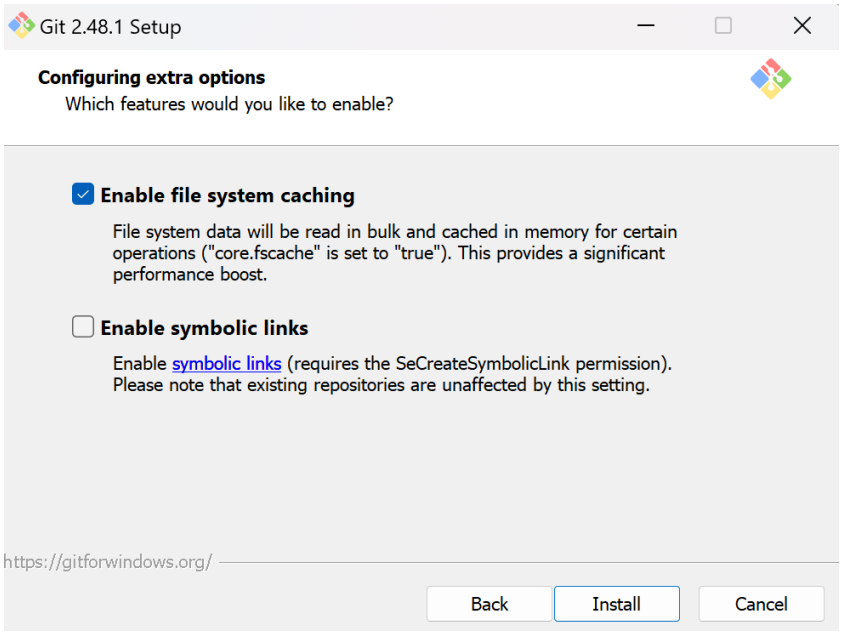


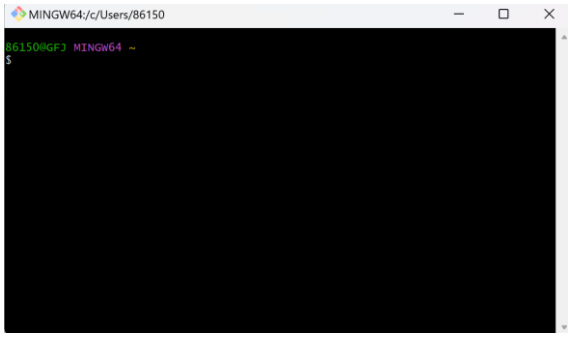
- **口令 credential**

密码的保存，需要安全的保存机制



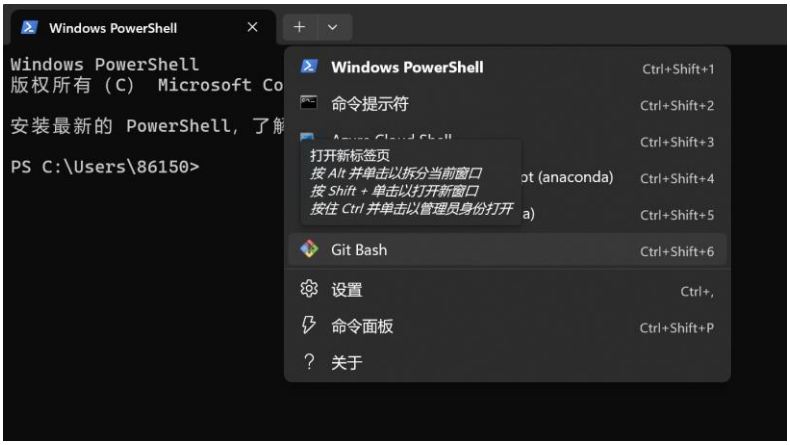
- 文件系统的缓存，符号链接（Windows 没有）



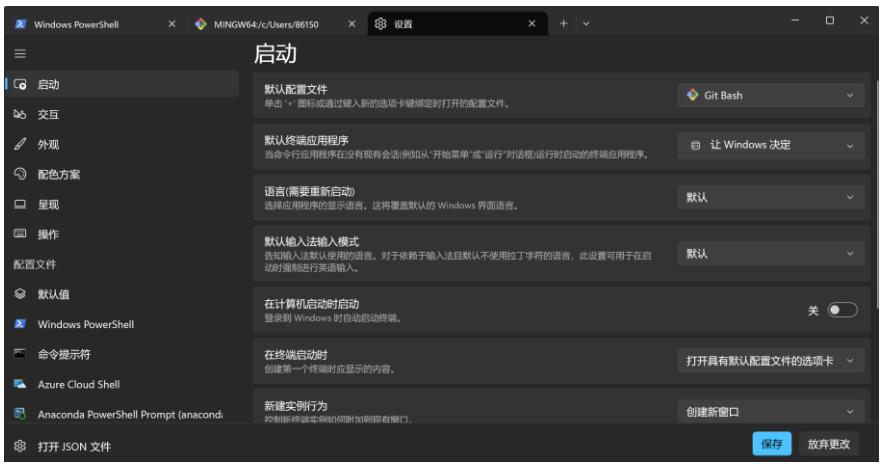


（乌班图子系统）

Windows 终端



修改启动默认配置文件：

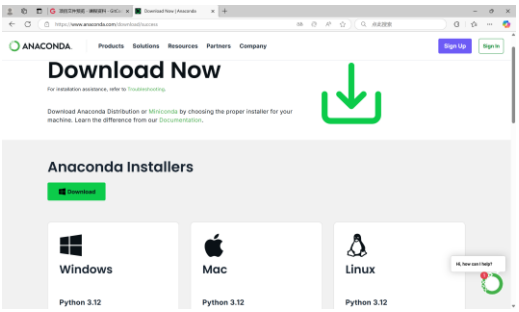


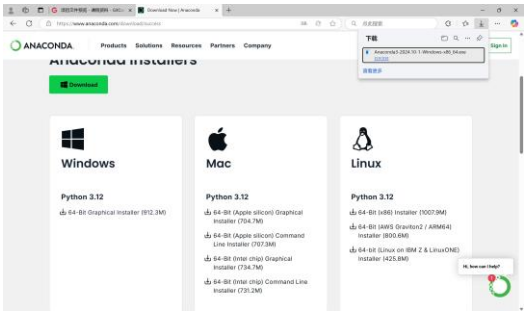
修改外观：

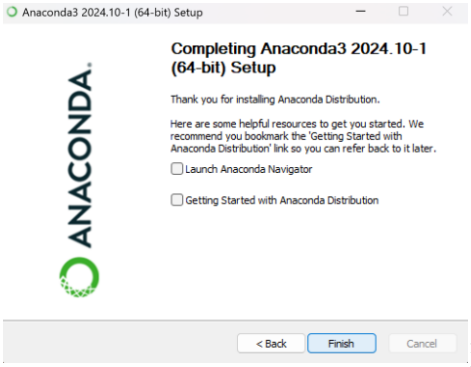
完成 Git 的安装和设置。

3. 安装 Anaconda（Python 解释器）

Anaconda 是开源的 Python 发行版，里面有很多数据科学相关的库和工具，能方便做数据分析，机器学习这些事情。

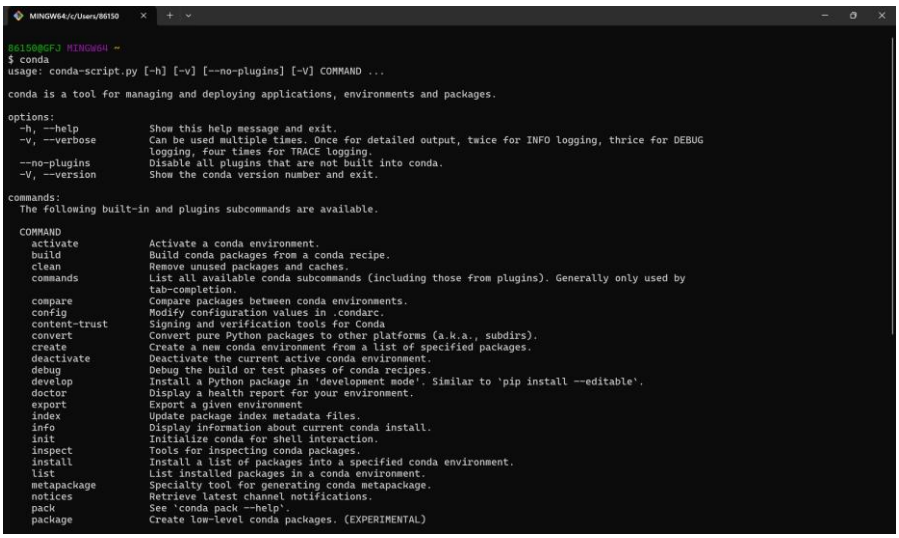






完成安装

使用：




```
86150@GFJ MINGW64 ~
$ which conda
/d/anaconda/Scripts/conda
```

PATH 环境变量：

```
86150@GFJ MINGW64 ~
$ echo $PATH
/c/Users/86150/bin:/mingw64/bin:/usr/local/bin:/usr/bin:/bin:/mingw64/bin:/usr/bin:/c/Users/86150/bin:/c/WINDOWS/system32:/c/WINDOWS:/c/WINDOWS/System32/Wbem:/c/WINDOWS/System32/WindowsPowerShell/v1.0:/c/WINDOWS/System32/OpenSSH:/cmd:/mingw64/bin:/usr/bin:/d/anaconda:/d/anaconda/Library/mingw-w64/bin:/d/anaconda/Library/usr/bin:/d/anaconda/Library/bin:/d/anaconda/Scripts:/c/Users/86150/AppData/Local/Microsoft/WindowsApps:/d/Microsoft VS Code/bin:/usr/bin/vendor_perl:/usr/bin/core_perl
```

路径优先级，文件从前往后找

Python 提示符：

```
86150@GFJ MINGW64 ~
$ python
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

应用：

```
86150@GFJ MINGW64 ~
$ ipython
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.27.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: print("hello")
hello

In [2]:
Do you really want to exit ([y]/n)? y

86150@GFJ MINGW64 ~
$ python
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hello")
hello
>>> quit()
```

Ctrl+d，按 y 退出 python/ 输入 quit()

新建目录：

```
86150@GFJ MINGW64 ~
$ pwd
/c/Users/86150

86150@GFJ MINGW64 ~
$ mkdir repo

86150@GFJ MINGW64 ~
$ ls
 「开始」菜单@
 AppData/
'Application Data'@
 Contacts/
 Cookies@
 Desktop/
 Documents/
 Downloads/
 Favorites/
 Links/
'Local Settings'@
 Music/
'My Documents'@
 NetHood@
 NTUSER.DAT
 ntuser.dat.LOG1
 ntuser.dat.LOG2
 NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.0.regtrans-ms
 NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.1.regtrans-ms
 NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.2.regtrans-ms
```

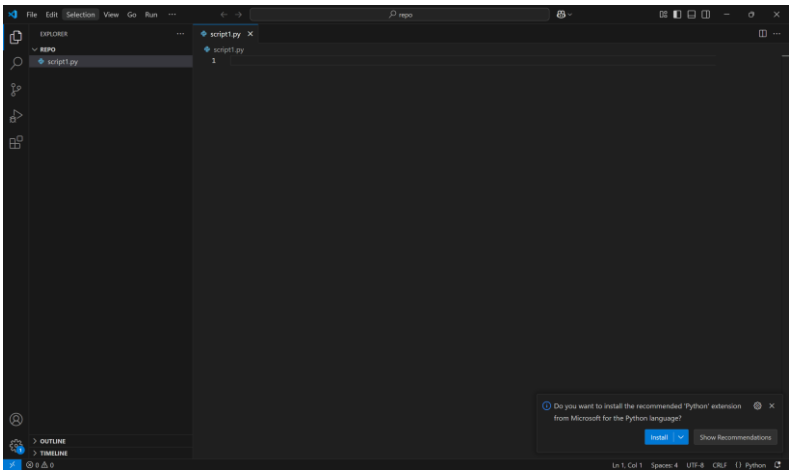
```
ntuser.dat.LOG1
ntuser.dat.LOG2
NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.0.regtrans-ms
NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.1.regtrans-ms
NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.2.regtrans-ms
NTUSER.DAT{41d29056-4f7f-11ef-9d6a-9591a138dd00}.TxR.blf
NTUSER.DAT{41d29057-4f7f-11ef-9d6a-9591a138dd00}.TM.blf
NTUSER.DAT{41d29057-4f7f-11ef-9d6a-9591a138dd00}.TMContainer00000000000000000001.regtrans-ms
NTUSER.DAT{41d29057-4f7f-11ef-9d6a-9591a138dd00}.TMContainer00000000000000000002.regtrans-ms
ntuser.ini
OneDrive/
Pictures/
PrintHood@
PycharmProjects/
Recent@
repo/
'Saved Games'/
Searches/
SendTo@
Templates@
Untitled.ipynb
untitled.py
Untitled1.ipynb
Untitled2.ipynb
Videos/
'WPS Cloud Files'/
WPSDrive/
```

打开 vs code：

```
86150@GFJ MINGW64 ~
$ cd repo

86150@GFJ MINGW64 ~/repo
$ code .
```

建立脚本：



1. 生成随机序列密码：

```
86150@GFJ MINGW64 ~/repo
$ cat script1.py
import random
import string
def generate_password(lenght):
    all_character=string.ascii_letters+string.digits+string.punctuation
    password="".join(random.choice(all_character)for i in range (lenght))
    return password
print(generate_password(12))

86150@GFJ MINGW64 ~/repo
$ python script1.py
}_D\{fc`(kdD

86150@GFJ MINGW64 ~/repo
$ python script1.py
^`J9eX/'znm"

86150@GFJ MINGW64 ~/repo
$ python script1.py
gaS('~yqY=T^
```

2. 若出错则会显示文件未找到等。（文件路径错误/文件名输入错误）

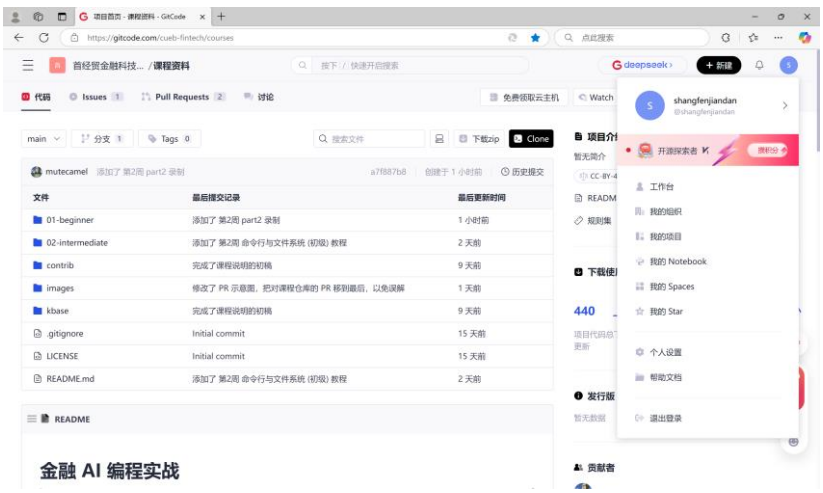
大模型的使用

使用豆包，可以解答一般问题。

e.g. SSH 密钥是什么？

SSH 密钥是一种用于安全登录远程服务器的加密技术，它比传统密码更安全，通过公钥和私钥配对来验证身份。

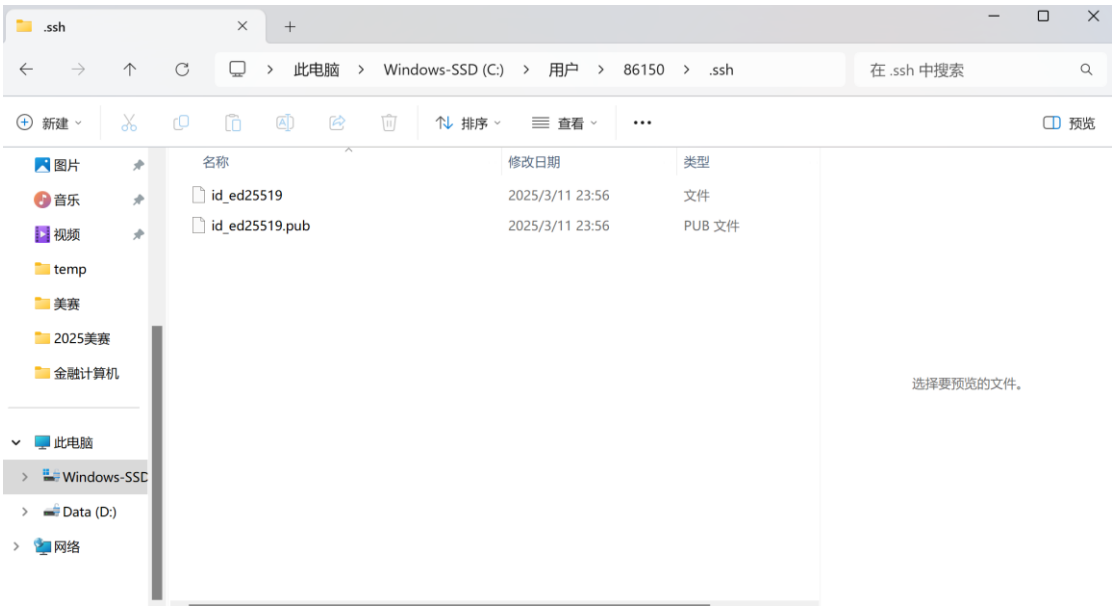
4. Git Code 用户注册



5. 配置 SSH 密钥，添加进 GitCode 安全设置

配置 SSH 密钥：

```
86150@GFJ MINGW64 ~
$ ssh-keygen -t ed25519 -C "15001090950@163.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/86150/.ssh/id_ed25519):
Created directory '/c/Users/86150/.ssh'.
Enter passphrase for "/c/Users/86150/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/86150/.ssh/id_ed25519
Your public key has been saved in /c/Users/86150/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:x1kcqCoTRwQYbV5Foys+zI05Ts/lQnA/lUQpeUrjwo4 15001090950@163.com
The key's randomart image is:
+--[ED25519 256]--+
|  .+.o.o+o.o.    |
|  . o o.=.=. .   |
|   o +.o B .o    |
|    + =.+.oo     |
|     .B.+S.+     |
|   +E=+ o.       |
|    X+. ..       |
|   o =.o         |
|    . o..        |
+----[SHA256]-----+
```



```
86150@GFJ MINGW64 ~
$ cd

86150@GFJ MINGW64 ~
$ pwd
/c/Users/86150

86150@GFJ MINGW64 ~
$ cd .ssh

86150@GFJ MINGW64 ~/.ssh
$ ls
id_ed25519  id_ed25519.pub

86150@GFJ MINGW64 ~/.ssh
$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMOGKHkswy3YMrrYABJpSDJnuGGjuEsF5GTSoEQnpxZJ 15001090950@163.com
```

密钥对：公钥用于加密，私钥用于解密。（**非对称加密技术**）

密钥加入 Gitcode：

测试 SSH 密钥：

```
86150@GFJ MINGW64 ~/.ssh
$ ssh -T git@gitcode.com
The authenticity of host 'gitcode.com (116.205.2.91)' can't be established.
RSA key fingerprint is SHA256:aTlsy+4ARMC7nWyy5eKIqUkotk8yv7Jd+XXoP4EXj1Y.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'gitcode.com' (RSA) to the list of known hosts.
remote: Welcome to GitCode, shangfenjiandan
```

```
86150@GFJ MINGW64 ~/.ssh
$  ssh -T git@gitcode.com
remote: Welcome to GitCode, shangfenjiandan
```

打开提交

查看全局配置：

```
86150@GFJ MINGW64 ~
$ git config --global user.name shangfenjiandan
git config --global user.email shangfenjiandan@noreply.gitcode.com

86150@GFJ MINGW64 ~
$ git config --list --global
core.editor="D:\Microsoft VS Code\bin\code" --wait
user.name=shangfenjiandan
user.email=shangfenjiandan@noreply.gitcode.com
```

克隆到本地：

```
86150@GFJ MINGW64 ~
$ pwd
/c/Users/86150

86150@GFJ MINGW64 ~
$ cd repo

86150@GFJ MINGW64 ~/repo
$ git clone git@gitcode.com:shangfenjiandan/week01.git
Cloning into 'week01'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 0), reused 7 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (7/7), 8.75 KiB | 1.25 MiB/s, done.
```