# Git 基础知识关卡

## 1. 安装 Git

### 1.1 Windows 系统

根据自己的 Windows 版本下载并依向导安装，在终端输入指令 git --version 检查是否安装成功。

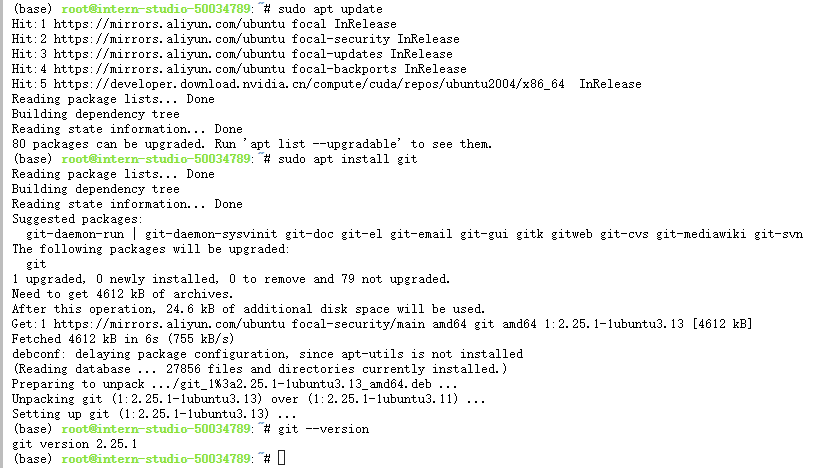
### 1.2 Linux 系统

通过包管理器安装 Git：

sudo apt update

sudo apt install git

输入指令 git --version 检查安装版本。

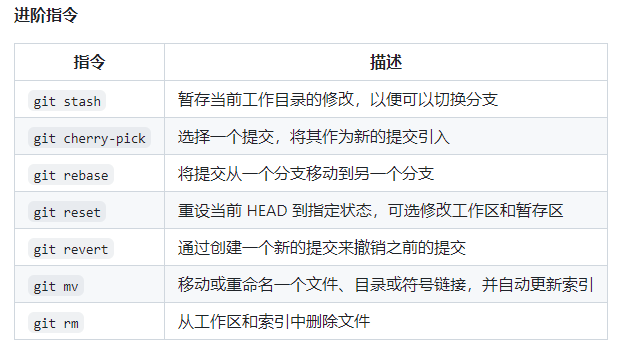


## Git 托管平台

* Github是全球最大的代码托管平台之一，拥有丰富的开源项目和活跃的开发者社区。它提供了版本控制、项目管理、协作开发等功能，并支持多种编程语言。
* Gitlab是一个自托管或基于云的平台，提供了完整的 DevOps 工具链，包括代码托管、持续集成/持续部署（CI/CD）、问题跟踪等。
* Getee是国内的代码托管平台，提供了代码托管、项目管理、协作开发等功能，对国内开发者来说，访问速度可能更快，也更符合国内的使用习惯。

## 3. 常用 Git 操作





可以使用命令行的帮助文档（git command -h,例如 git pull -h）来获取更多信息。

## 4. 食用小 tips

### 4.1 全局设置 vs. 本地设置

* **全局设置**：这些设置影响你在该系统上所有没有明确指定其他用户名和电子邮件的 Git 仓库。这是设置默认用户名和电子邮件的好方法。
* **本地设置**：这些设置仅适用于特定的 Git 仓库。这对于你需要在不同项目中使用不同身份时很有用，例如区分个人和工作项目。

### 4.2 如何配置

**全局设置用户信息**

打开终端或命令提示符，并输入以下命令来设置全局用户名和电子邮件地址：

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"这里的 "Your Name" 和 "your.email@example.com" 应替换为你自己的姓名和电子邮件。

**本地设置用户信息**

首先，确保你当前处于你想要配置的 Git 仓库的目录中。然后，输入以下命令来仅为该仓库设置用户名和电子邮件地址：

git config --local user.name "Your Name"

git config --local user.email "your.email@example.com"

同样，替换 "Your Name" 和 "your.email@example.com" 为该特定项目中使用的姓名和电子邮件。

### 4.3 验证设置

**查看全局配置**：

git config --global --list

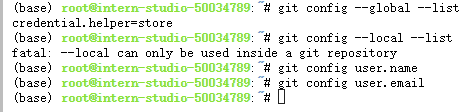
**查看仓库配置**：

git config --local --list

**查看特定配置项**：

git config user.name

git config user.email



### 4.4 Git 四步曲

Git的日常使用中，下面四步曲是常用的流程，尤其是在团队协作环境中。

**添（Add）**

* **命令**：git add <文件名> 或 git add .
* **作用**：将修改过的文件添加到本地暂存区（Staging Area）。这一步是准备阶段，你可以选择性地添加文件，决定哪些修改应该被包括在即将进行的提交中。

**提（Commit）**

* **命令**：git commit -m '描述信息'
* **作用**：将暂存区中的更改提交到本地仓库。这一步是将你的更改正式记录下来，每次提交都应附带一个清晰的描述信息，说明这次提交的目的或所解决的问题。

**拉（Pull）**

* **命令**：git pull
* **作用**：从远程仓库拉取最新的内容到本地仓库，并自动尝试合并到当前分支。这一步是同步的重要环节，确保你的工作基于最新的项目状态进行。在多人协作中，定期拉取可以避免将来的合并冲突。

**推（Push）**

* **命令**：git push
* **作用**：将本地仓库的更改推送到远程仓库。这一步是共享你的工作成果，让团队成员看到你的贡献。

## 5. 常用插件

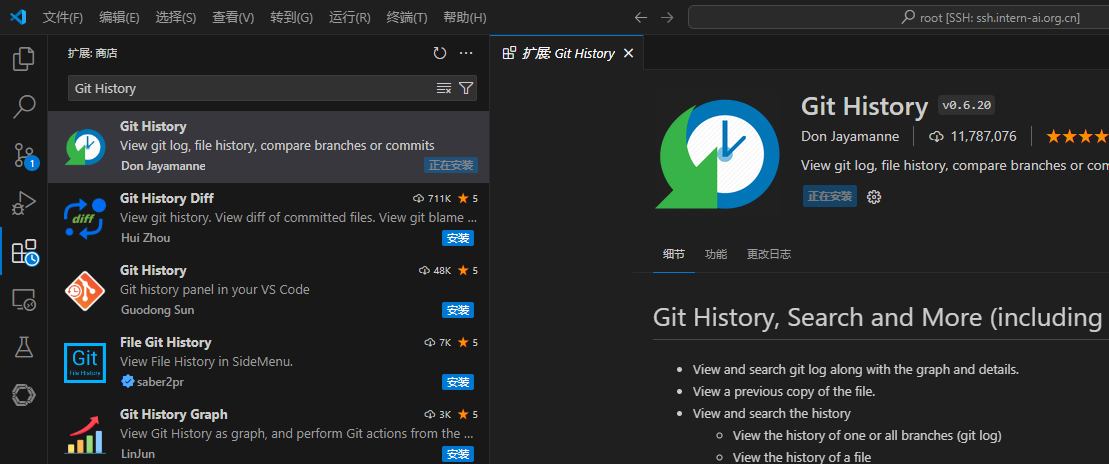
* [GitLens](https://marketplace.visualstudio.com/items?itemName=eamodio.gitlens): 在代码行上显示 Git 提交信息。



* [Git Graph](https://marketplace.visualstudio.com/items?itemName=mhutchie.git-graph): 类似于 SourceTree 的可视化版本控制插件。



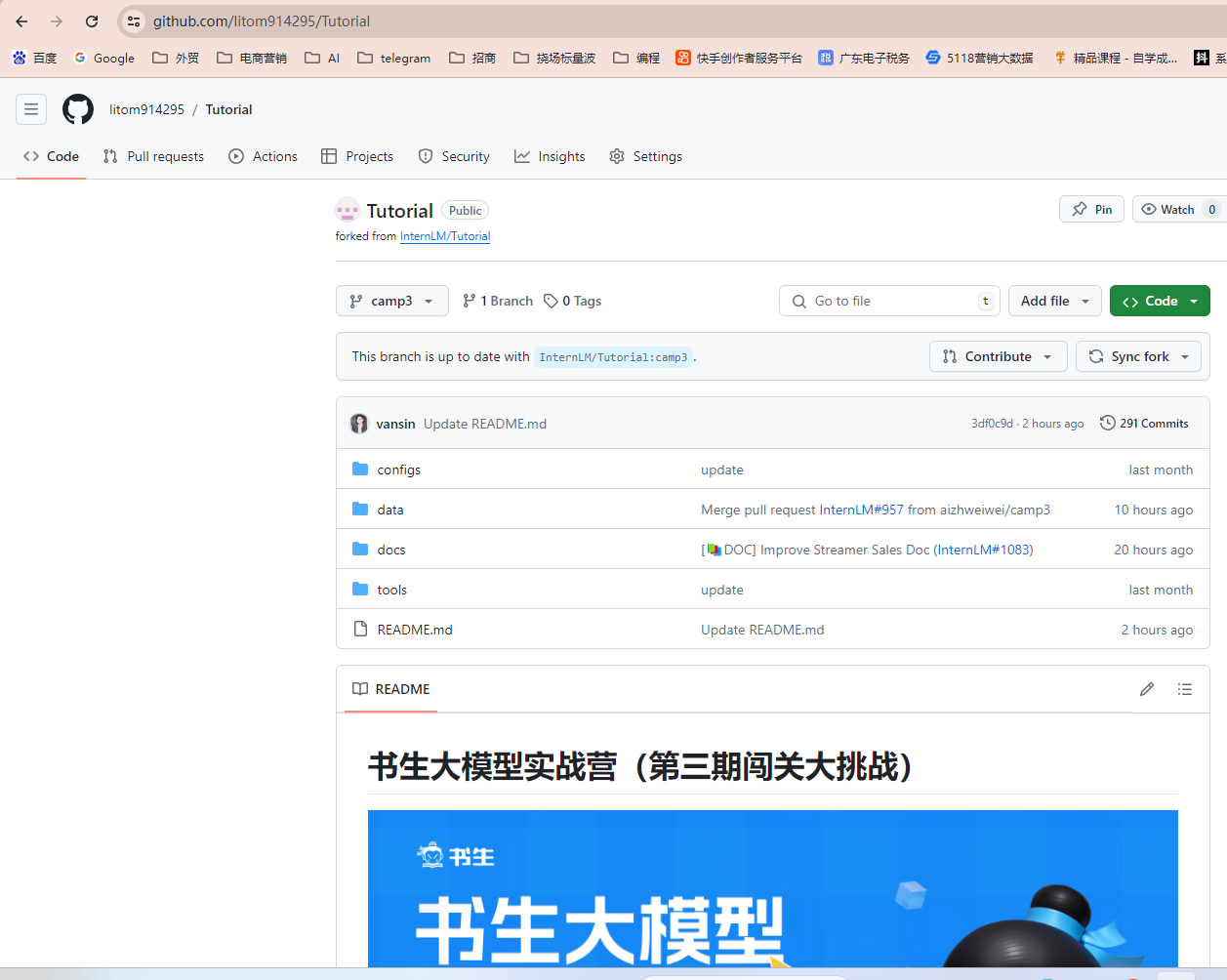
* [Git History](https://marketplace.visualstudio.com/items?itemName=donjayamanne.githistory): Git 日志查看器。



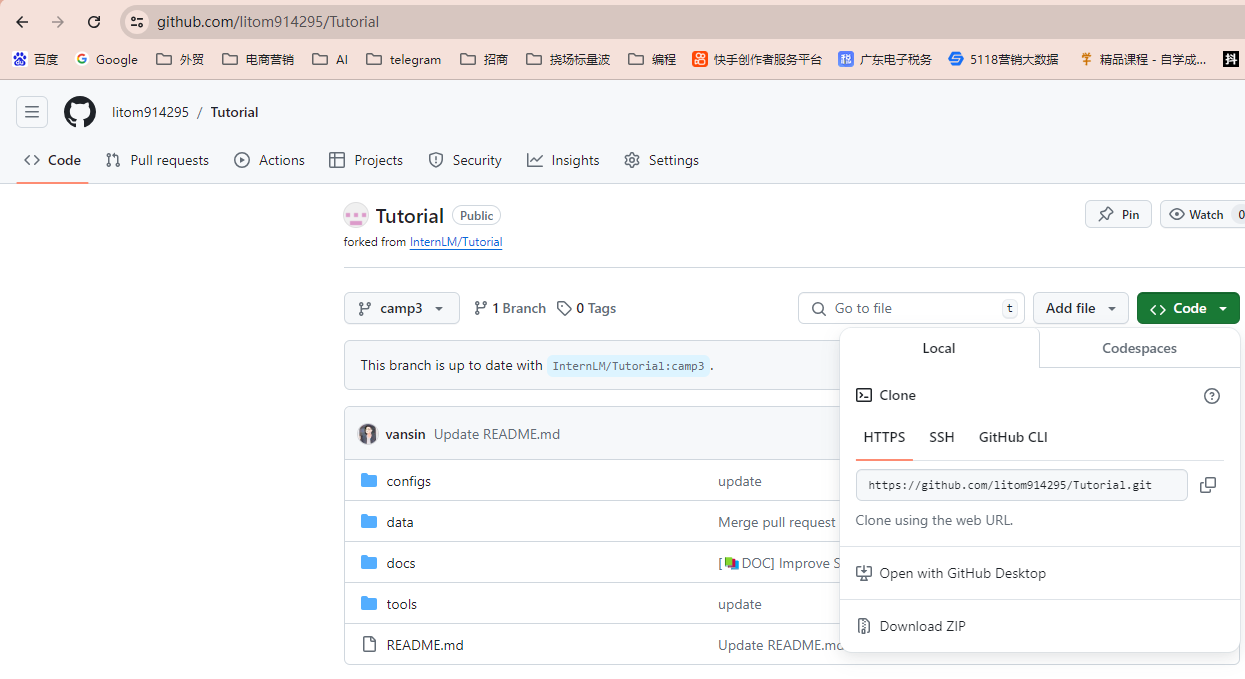
## 6. 常规开发流程

* **Fork 目标项目**





* **获取仓库链接**



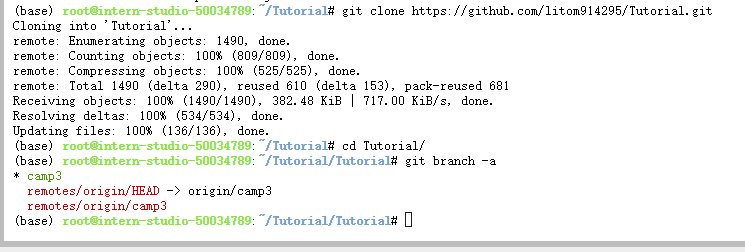
git clone <https://github.com/litom914295/Tutorial.git>

cd Tutorial/

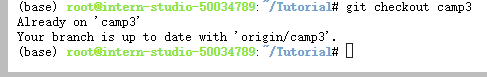
git branch -a

git checkout -b camp3 origin/camp3

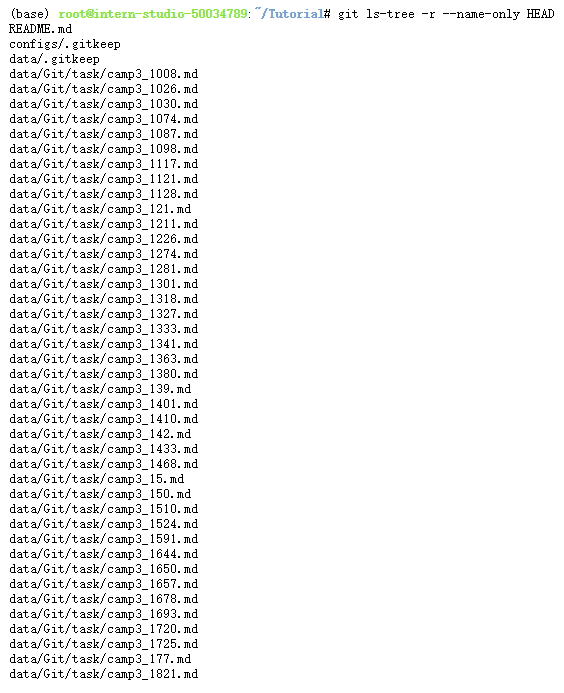
* **查看分支**



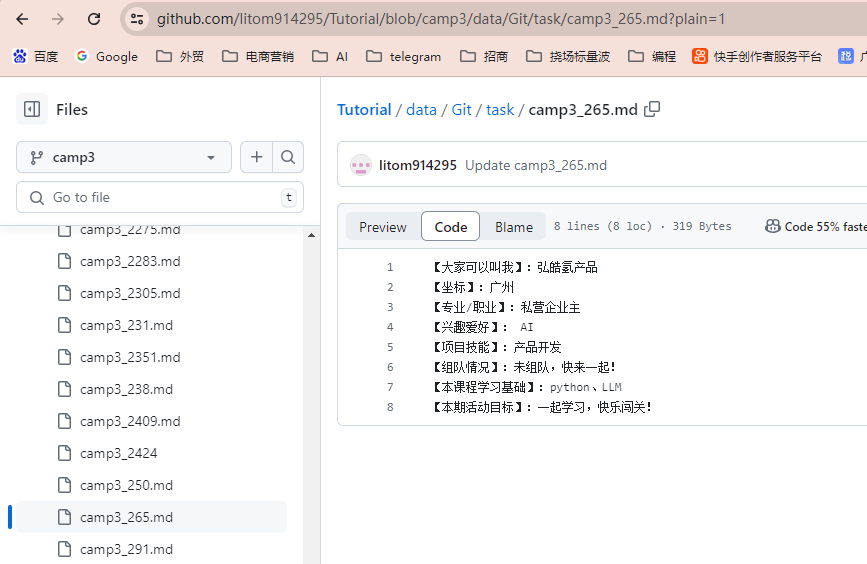
* **切换到第三期的分支**



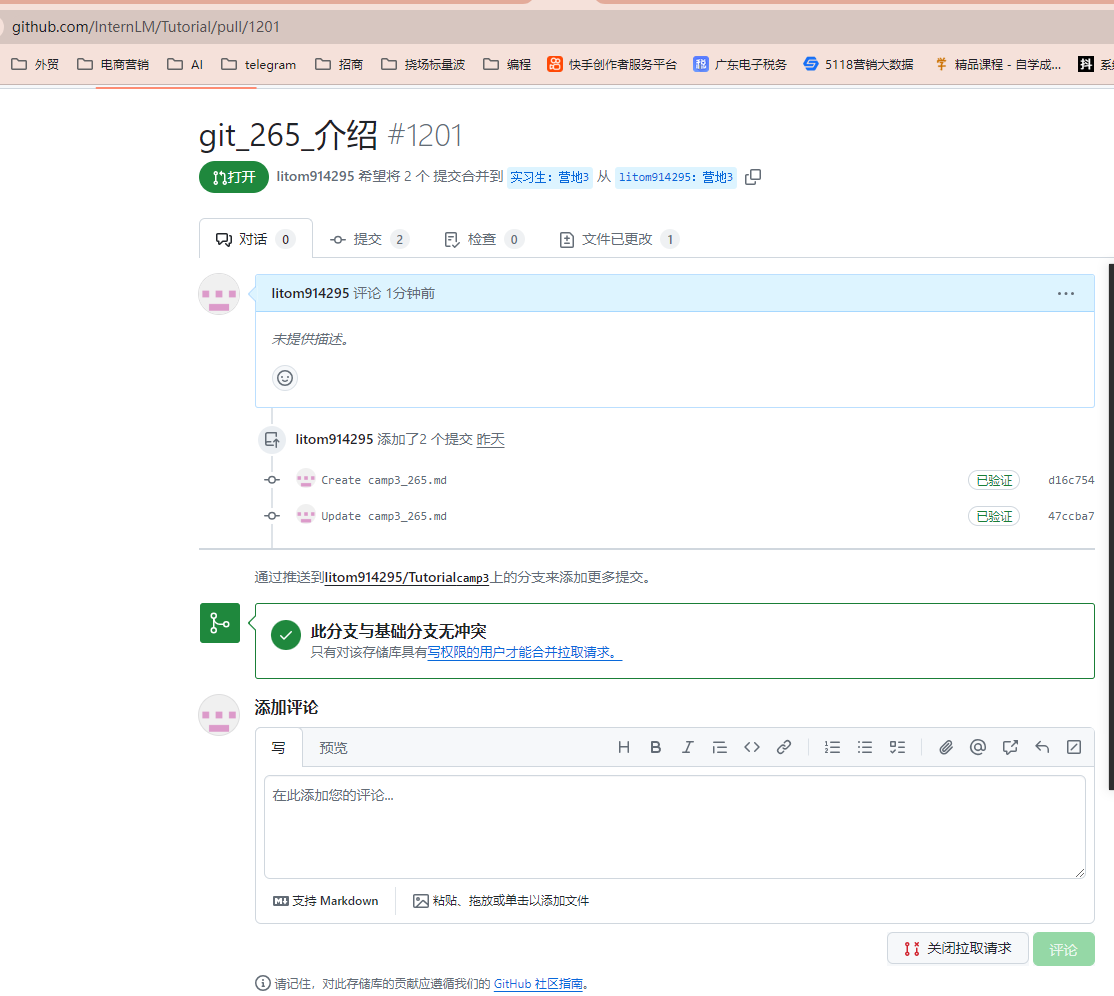
* **查看分支内容**



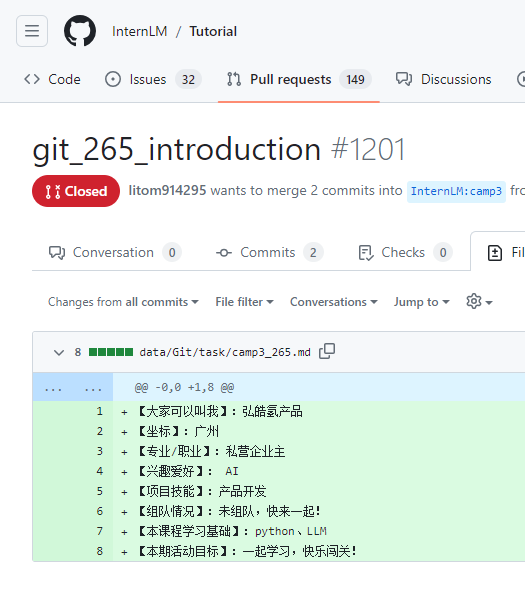
1721368547123



在 自己的github 页面将修改的内容 PR 到 Tutorial，并修改title后提交：



查看修改的内容



## 任务2: 实践项目：构建个人笔记仓库

<https://github.com/litom914295/LLMLearn>