Git团队开发教程（给非仓库所有者的开发者）

1. **设置环境**

首先，你需要确保在你的本地机器上安装了 Git。你可以通过在命令行或终端中运行 `git --version` 来检查这一点。如果你没有 Git，你可以从官方 Git 网站上安装它。

2. **克隆存储库**

一旦安装了 Git，下一步是将存储库克隆到你的本地机器上。你可以使用以下命令来完成此操作：

```
git clone <repository_url>
```

`<repository_url>` 可以从你的 GitHub 存储库主页面上的 "Clone" 按钮获取。

3. **在开始工作之前始终拉取**

这是教程的主要部分。在你每天开始编码或创建新的分支之前，你应该始终确保你的本地代码与远程存储库的主分支是最新的。你可以使用以下命令来完成此操作：

```
git checkout main
git pull origin main
```

这将更新你本地的主分支版本，使其与远程存储库匹配。

4. **为每个任务创建新分支**

为每个任务创建新分支是一种良好的做法。这样，每个任务的更改都是独立的，并且可以在合并到主分支之前分别进行审查。你可以使用以下命令创建并切换到新的分支：

```
git checkout -b <branch_name>
```

`<branch_name>` 应该是一个简短的、描述性的名称，表示你在此分支上要进行的任务。

5. **提交并推送更改**

在你进行了一些更改之后，你应该提交这些更改，并将它们推送到远程存储库。你可以使用以下命令来完成此操作：

```
git add .
git commit -m "<commit_message>"
git push origin <branch_name>
```

`<commit_message>` 应该是一个简短的、描述性的消息，解释在此提交中做了什么更改。

6. **创建拉取请求**

一旦你推送了你的更改，你应该去 GitHub 网站并创建一个拉取请求。这是一个请求将你的更改合并到主分支的请求。你应该在拉取请求的描述中提供对你的更改的详细解释和任何其他相关信息。

7. **定期更新他们的分支**

在工作的分支上，你应该定期更新它，以包含主分支的任何新更改。这减少了合并时出现冲突的可能性。你可以使用以下命令更新你的分支：

```
git checkout <branch_name>
git pull origin main
```

再次，`<branch_name>` 是你当前正在工作的分支的名称。

当然！我将为VSCode和PyCharm生成一个简单的教程，重点介绍如何将这些IDE与GitHub一起使用。

VSCode GitHub 整合教程

1. **安装 Git**

   首先，你需要确保你的计算机上已经安装了Git。你可以通过打开一个终端窗口并输入 `git --version` 来验证这一点。

2. **安装 VSCode**

   如果你还没有这样做，请从官方网站下载并安装VSCode。

3. **为 Visual Studio Code 安装 GitHub 扩展**

   你可以在扩展视图中找到这个（在Mac上是 `Ctrl+Shift+X` 或 `Cmd+Shift+X`），搜索 "GitHub Pull Requests and Issues"，然后安装它。

4. **打开仓库**

   使用VSCode中的 `File > Open Folder` 菜单选项打开已克隆的仓库文件夹。你应该能在窗口的左下角看到仓库的名称。

5. **分支和提交**

   - 你可以通过 `Source Control` 标签（在Mac上是 `Ctrl+Shift+G` 或 `Cmd+Shift+G`）来创建一个新的分支，点击 `...` 按钮，然后选择 `Create Branch`。
   - 对你的文件进行更改，然后返回到 `Source Control` 标签。在这里，你会看到一列变更。
   - 在文本框中输入提交消息，然后按 `Ctrl+Enter`（在Mac上是 `Cmd+Enter`）来提交更改。

6. **推送更改**

   - 要推送你的更改，再次点击 `...` 按钮，然后选择 `Push`。
   - 如果你是第一次推送并且创建了一个新的分支，VSCode会要求你设置上游。选择 `Yes`，分支将被推送到GitHub。

7. **拉取请求**
    - 通过点击VSCode侧边的活动栏上的GitHub图标，打开GitHub视图。
    - 要创建一个新的拉取请求，点击 "Pull Requests" 旁边的 + 图标。选择你当前的分支与主分支进行比较。

PyCharm GitHub 整合教程

1. **安装 Git**

   和 VSCode 一样，你的计算机上必须安装 Git。

2. **安装 PyCharm**

   从 JetBrains 官方网站下载并安装 PyCharm。

3. **连接 GitHub 到 PyCharm**
    - 前往 `File > Settings`（在Mac上是 `PyCharm > Preferences`），然后到 `Version Control > GitHub`。
    - 在这里，你可以通过使用令牌或使用你的凭据登录来添加你的GitHub账户。

4. **克隆仓库**
    - 从 `VCS > Get from Version Control` 菜单，你可以克隆仓库。你只需要粘贴仓库URL并选择克隆的目录。

5. **分支和提交**
    - 要创建一个新的分支，前往 `VCS > Git > Branches`，然后选择 `New Branch`。
    - 在进行更改后，前往 `VCS > Commit` 或按 `Ctrl+K`（在Mac上是 `Cmd+K`）。
    - 在新窗口中，写下你的提交消息并按 `Commit`。

6. **推送更改**
    - 要将你的更改推送到 GitHub，你可以前往 `VCS > Git > Push` 或按 `Ctrl+Shift+K`（在Mac上是 `Cmd+Shift+K`）。

7. **拉取请求**
    - 打开 GitHub 视图（`View > Tool Windows > GitHub`）。
    - 要创建拉取请求，点击 "Pull Requests" 旁边的 + 图标，填写详细信息，并提交。

记住要定期从主分支拉取最新的更改到两个IDE中，以保持你的本地代码最新。在 PyCharm 中，使用 `VCS > Git > Pull`。在 VSCode 中，使用 `Ctrl+Shift+P`（在 Mac 上是 `Cmd+Shift+P`）打开命令面板并选择 `Git: Pull`。

管理指南（给仓库所有者）

1. **执行拉取请求（PR）审查过程**

实施一项政策，即每个 PR 必须在合并之前至少由一个其他人审查。这确保了有超过一个人理解每个更改，并帮助捕捉错误。

2. **使用一致的分支模型**

执行一致的分支模型。例如，你可能使用一个系统，其中有一个主分支，每个新特性有特性分支，每个 bug 修复有 bug 分支。

3. **使用问题跟踪器**

充分利用 GitHub 的问题跟踪器。这有助于跟踪需要完成的事情以及每项任务的责任人。它还提供了每次更改决策过程的记录。

4. **定期沟通**

与你的团队保持定期沟通。这可能是每日站立会议，