

## 1. 克隆仓库到本地

```
1 git clone git@github.com:BoBo111a/yolo-accident-detector.git
2 cd yolo-accident-detector
```

## 2. 创建个人开发分支（每天开始开发前）

```
1 git checkout -b feature/yourname # 推荐分支命名规范
2 # 例如: git checkout -b feature/zd
```

## 3. 开发与本地提交

```
1 # 修改代码后提交
2 git add .
3 git commit -m "完成xxx功能（具体描述变更内容）"
```

## 4. 推送分支到远程

```
1 git push origin feature/yourname
2 # 例如 git push origin feature/zd
3 # 首次推送需要加 -u 参数:
4 git push -u origin feature/yourname
5 # 例如 git push -u origin feature/zd
```

## 5. 创建Pull Request (PR)

### PR 就像「代码的请假条」

想象小组在共同编辑一份**电子文档**（比如腾讯文档）：

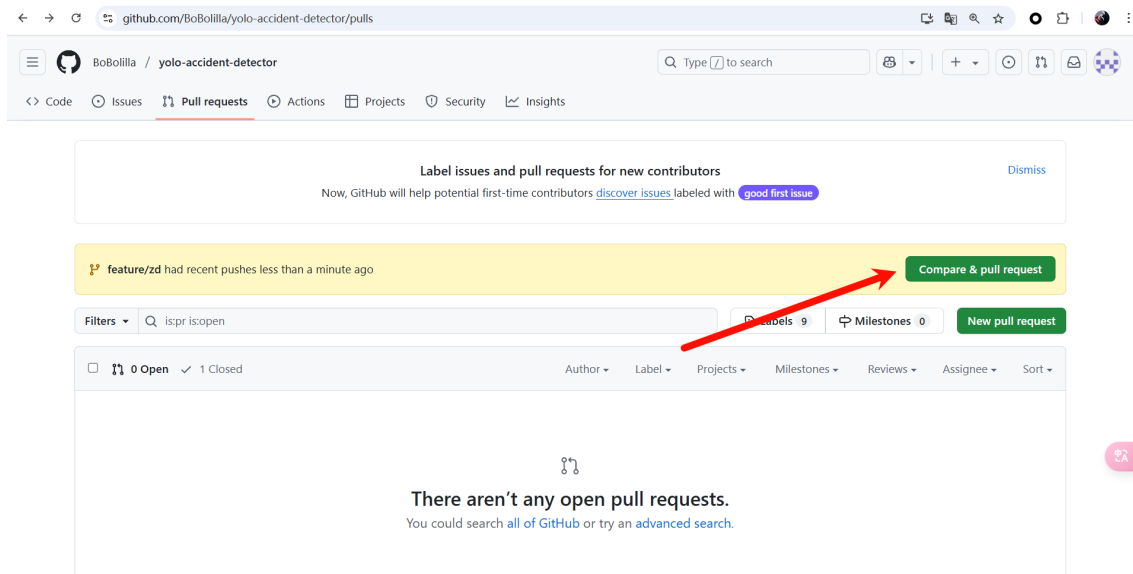
#### 1. 直接改主文档 ❌

- 如果所有人同时乱改，文档会一团糟（代码冲突/错误）

#### 2. PR 的工作方式 ✅

- 你：把要改的内容**先复制一份**（你的分支），在自己的副本里修改
- 改完后，提交一个「申请」说：“*我改好了，请组长检查一下再合并到主文档*” → 这就是 PR！

- 在GitHub网页端操作
- 选择 `Compare & pull request`



- 这里的 `compare` 选择你自己创建的 `feature/yourname` 分支对比，对比更改后填写PR描述即可，PR描述就是你负责的部分完成了什么功能，效果怎样

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff comparisons here.](#)

base: master ← compare: feature/zd ✓ Able to merge. These branches can be automatically merged.

Add a title

Feature/zd

Add a description

Write Preview

添加使用git管理项目的工作流文档---》UseGit.md和UseGit.pdf

Markdown is supported Paste, drop, or click to add files

Create pull request

Remember, contributions to this repository should follow its [contributing guidelines](#).

Reviews

No reviews

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

[Contributing](#)

- 关联相关Issue（如有）

## 6. 代码审查与合并

### 1. 其他成员审查代码：

- 在PR页面的Files changed标签页进行行级评论
- 使用 `/test` 命令触发CI（如果配置了自动化测试）

### 2. 合并代码：

```
1 # 当PR通过审查后（组长操作）
2 git checkout master
3 git pull origin master # 确保本地master最新
4 git merge --no-ff feature/branch-name # 保留分支历史
5 git push origin master
```

## 同步最新代码（组员日常操作）

```
1 # 每天开始工作前
2 git checkout master
3 git pull origin master
4
5 # 回到开发分支合并最新代码
6 git checkout feature/your-branch
7 git merge master # 处理可能出现的冲突
```

## 冲突解决流程

当出现冲突时：

```
1 # 查看冲突文件
2 git status
3
4 # 手动编辑标记了<<<<<<的文件
5 # 解决后提交
6 git add .
7 git commit -m "解决与master分支的合并冲突"
```

## 常用命令速查表

操作	命令
查看分支	<code>git branch -av</code>
删除已合并分支	<code>git branch -d branch-name</code>
暂存修改	<code>git stash</code>
查看远程仓库	<code>git remote -v</code>
撤销本地修改	<code>git checkout -- filename</code>

使用 `git log --graph --oneline` 查看提交历史

使用 `git shortlog -sn` 查看成员贡献统计。