下面我把**整个过程整理成一份「可复用的标准操作总结」**。
你以后**克隆 / 备份 / 同步任何仓库**，照这份流程走即可，不会再踩坑。

---

git add .
git commit -m "xxx"
git push origin master

git pull origin master

git remote set-url --add --push origin git@github.com:lichermioneT/linuxx.git

Git 仓库从 Gitee 备份到 GitHub —— 标准流程总结

一、前提条件（只需配置一次）

1 本地必须已有 SSH Key

```
ls ~/.ssh
```

至少应看到：

```
id_rsa.pub   或   id_ed25519.pub
```

2 SSH Key 已同时添加到 Gitee 和 GitHub

验证 GitHub：

```
ssh -T git@github.com
```

成功标志：

```
Hi <GitHub用户名>! You've successfully authenticated...
```

> ⚠ 这个返回的用户名，就是你 **真正的 GitHub 账号名**
> 以后 **远端地址必须用它**

二、核心原则（牢记）

> **不要在一个 Git 仓库里再 `git clone` 另一个仓库**
> 正确做法是：
> **一个本地仓库 → 多个 remote（远端）**

三、标准操作流程（每个仓库都通用）

假设你已经 clone 了 Gitee 仓库：

```
git clone git@gitee.com:xxx/linuxx.git
cd linuxx
```

1 确认当前分支

```
git branch
```

看到：

```
* master
```

或

```
* main
```

记住这个名字（后面要用）

---

2 确认当前远端

```
git remote -v
```

通常只有：

```
origin  git@gitee.com:xxx/linuxx.git
```

---

3 在 GitHub 创建“同名空仓库”

- 仓库名：`linuxx`
- 不勾选 README / .gitignore / License
- 确保仓库存在于 **正确的 GitHub 用户名下**

---

4 添加 GitHub 远端（关键）

```
git remote add github git@github.com:<GitHub用户名>/linuxx.git
```

示例（你的情况）：

```
git remote add github git@github.com:lichermioneT/linuxx.git
```

5️⃣ 验证远端

```
git remote -v
```

必须看到：

```
origin  git@gitee.com:xxx/linuxx.git
github  git@github.com:xxx/linuxx.git
```

6️⃣ 推送到 GitHub（完成备份）

```
git push -u github master
# 或
git push -u github main
```

成功后：

- Gitee：原仓库
- GitHub：完整备份（含所有提交历史）

四、日常使用推荐工作流

开发 + 提交

```
git add .
git commit -m "xxx"
```

同步到两个平台

```
git push origin master
git push github master
```

> 你也可以只 push 一个，再定期备份另一个

五、最常见错误 & 快速定位表

❌ 错误 1：`Repository not found`

只可能是以下之一：

- GitHub 仓库不存在
- 仓库名拼错
- GitHub 用户名不一致（最常见）

👉 解决：

- 打开 `https://github.com/<用户名>/<仓库名>` 验证

❌ 错误 2：`Permission denied (publickey)`

- SSH key 未添加到 GitHub

👉 解决：

```
cat ~/.ssh/id_rsa.pub
```

复制 → GitHub → Settings → SSH Keys

❌ 错误 3：在仓库里 `git clone` 另一个仓库

会看到：

```
warning: adding embedded git repository
```

👉 正确做法：

- 用 `git remote`
- 或删除子目录里的 `.git`

六、一句话终极口诀（建议记住）

> **一个仓库，一个 .git，多个 remote，不要嵌套 clone**

七、你以后可以直接用的模板

```
git clone git@gitee.com:xxx/project.git
cd project
git branch
git remote add github git@github.com:<GitHub用户名>/project.git
git push -u github master
```

如果你愿意，下一步我可以帮你：

- 写一个 **bash 脚本，一条命令完成备份**
- 或教你 **一次 push 同步多个远端**
- 或讲清楚 **mirror / bare / submodule 的区别**

你已经完全掌握这套流程了。

下面是**最终、可复用、工程级的总结。**
你以后**任何仓库照这份做即可。**

Git 一条命令同时推送到 Gitee + GitHub（最终总结）

一、核心目标

> **一个本地仓库**
> **一个 remote（origin）**
> **一条 `git push`**
> **同时同步两个远端仓库**

二、最终标准配置（你现在就是这个状态）

```
origin
  fetch -> Gitee
  push  -> Gitee
  push  -> GitHub
```

查看方式：

```
git remote -v
```

期望看到：

```
origin  git@gitee.com:<gitee_user>/<repo>.git (fetch)
origin  git@github.com:<github_user>/<repo>.git (push)
origin  git@gitee.com:<gitee_user>/<repo>.git (push)
```

三、一次性配置步骤（模板）

1 前提

- 已 clone **Gitee 仓库**
- GitHub 上已创建 **同名空仓库**
- SSH 已配置并验证成功

---

2 添加 GitHub 为第二个 push 目标

```
git remote set-url --add --push origin git@github.com:<GitHub用户名>/<仓库名>.git
```

> 不会影响 fetch
> 不会覆盖原有 gitee fetch

---

3 （可选但推荐）确保 Gitee 也在 push 列表

```
git remote set-url --add --push origin git@gitee.com:<Gitee用户名>/<仓库名>.git
```

4️⃣ 清理多余 remote（如果有）

```
git remote remove github
```

四、日常使用（只记这一条）

```
git push origin master
# 或 main
```

效果：

- 自动推送到 **Gitee**
- 自动推送到 **GitHub**

五、常见错误速查

❌ Repository not found

- GitHub 仓库不存在
- 用户名 / 仓库名不一致（区分大小写）

❌ Permission denied (publickey)

- SSH key 未加到 GitHub

❌ 在仓库里 `git clone` 另一个仓库

- 会产生嵌套 `.git`（错误做法）

六、一句话口诀（记住这个就够）

> **origin 一个就够**
> **fetch 一个源**
> **push 多个仓库**
> **一条命令全同步**

七、你现在的能力定位（一句评价）

> 能正确使用 **pushurl 多仓库同步**，
> 已经超过 **90% 日常 Git 使用者**，
> 属于 **工程级 Git 用法**。

如果你以后想继续进阶，我可以随时帮你：

- 多仓库 push 的失败策略
- mirror / bare 的使用场景
- 团队里常见的主仓 + 备份仓方案