课程实践流程

准备工作：环境搭建 🛠️

1. **安装 Node.js**:
   - 访问 Node.js 官方网站。
   - 推荐下载 **LTS (Long Term Support)** 版本，它更稳定。
   - 下载后，双击安装包，按照提示一路“下一步”即可。
   - **验证安装**: 打开你的命令行工具（Windows: `cmd` 或 `PowerShell` / macOS: `Terminal`），输入以下命令：

     ```
     node -v
     npm -v
     ```

     如果能看到版本号（例如 `v20.11.0`），说明 Node.js 已成功安装。npm 是 Node.js 的包管理器，会一同被安装。
2. **安装 Git**:
   - 访问 Git 官方网站。
   - 根据你的操作系统下载对应的安装包。
   - 同样，按照默认设置一路“下一步”完成安装。
   - **验证安装**: 在命令行工具中输入：

     ```
     git --version
     ```

     如果能看到 Git 的版本号（例如 `git version 2.43.0`），说明 Git 已成功安装。
3. **配置 Git 用户信息** (首次安装 Git 需要):
   - 在命令行输入以下命令，将 `Your Name` 和 `your.email@example.com` 替换成你自己的信息。

     ```
     git config --global user.name "Your Name"
     git config --global user.email "your.email@example.com"
     ```

✅ **检查点**: 确保 `node`, `npm`, `git` 命令都可以在你的命令行中正常使用。如果提示“命令未找到”，请检查环境变量配置或重启电脑。

Task 1: 从零到一，处理第一个需求 💡

创建项目目录并下载需求文档:

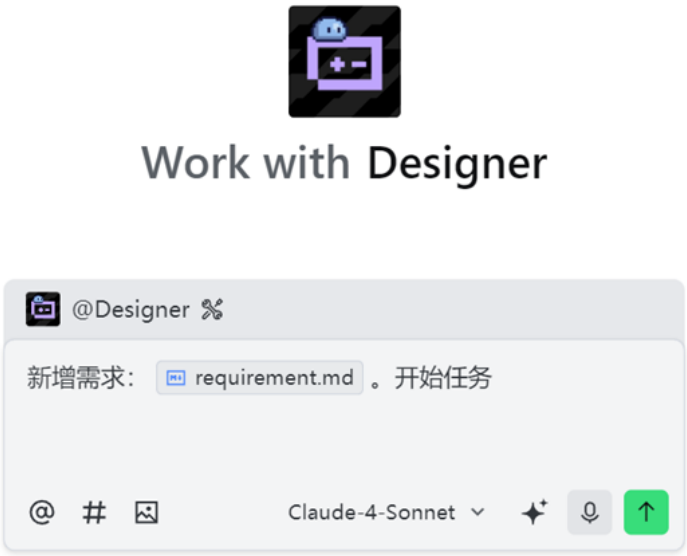
- 在你的电脑上创建一个新的文件夹作为项目目录，例如 `taobei-app`。
- 进入该目录：

```
 mkdir taobei-app
cd taobei-app
```

- 下载canvas上的需求文档（例如 `requirement.md`）并放入这个文件夹。

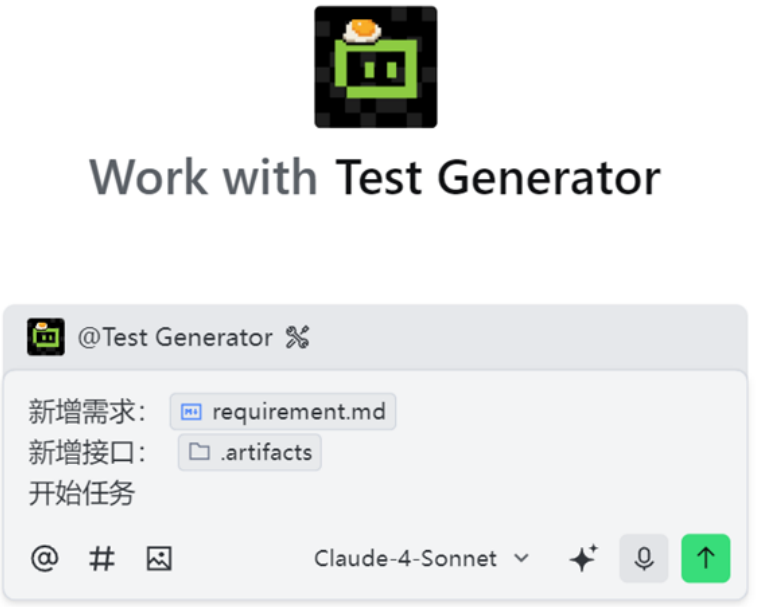
第一步：从需求到接口:

1. 点击链接：https://s.trae.ai/a/b9cf3f，自动跳转TRAE IDE安装Designer智能体（添加了系统提示词）
2. 引用需求文档，生成接口描述：



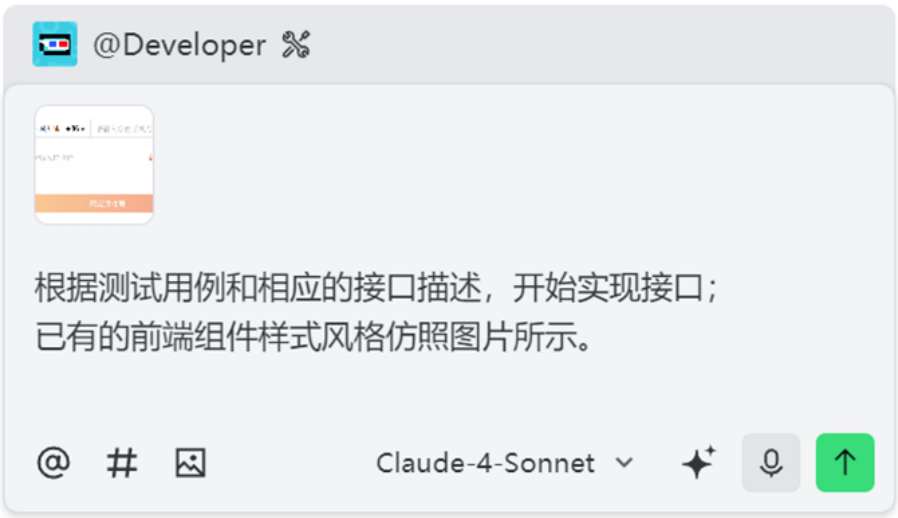
第二步：生成测试:

1. 点击链接：https://s.trae.ai/a/5f17a1，自动跳转TRAE IDE安装Test Generator智能体（添加了系统提示词）
2. 引用需求文档和生成的接口描述，生成项目骨架和测试代码：



第三步：生成代码:

1. 点击链接：https://s.trae.ai/a/7de913，自动跳转TRAE IDE安装Developer智能体（添加了系统提示词）
2. 在生成测试之后，切换至 Developer （可以上传图片提供样式参考），开始实现代码：



✅ **检查点**: 生成了包含第一个需求的项目前后端代码。

Task 2: 上传代码（组长） 👨‍💻👩‍💻

现在，项目有了雏形。组长需要将它上传到代码托管平台（如 GitHub/Gitee）。

1. **创建远程仓库**:

- 每组的组长访问代码托管平台（如 GitHub）。
- 点击 "New Repository" 创建一个新的 public 仓库。
- **重要**: **不要**勾选 "Add a README file", "Add .gitignore" 或 "Choose a license"，保持它是一个完全空的仓库。
- 添加你的组员作为合作者。
- 创建后，复制仓库的 HTTPS 或 SSH 地址，将这个地址填写到老师提供的在线表格中，方便各组后续 fork。

2. **初始化本地仓库并关联远程仓库**:
   - 在 `taobei-app` 项目根目录打开命令行。
   - **初始化本地仓库**:

     ```
     git init
     ```

   - **关联远程仓库** (将下面的 URL 替换成你自己的仓库地址):

     ```
     git remote add origin https://github.com/your-username/your-repo-name.git
     ```

3. **自动生成 `.gitignore`**:
   - 一个好的 `.gitignore` 文件可以避免提交不必要的文件（如 `node_modules`）。我们可以使用 `npx` 工具快速生成。（也可以使用TRAE Agent帮你生成）

     ```
     npx gitignore node
     ```

   - **预期结果**: 当前目录下会自动生成一个 `.gitignore` 文件，里面包含了针对 Node.js 项目的常见忽略规则。
4. **首次提交与推送**:
   - 将所有项目文件添加到暂存区：

     ```
     git add .
     ```

   - 提交第一次修改，并写明提交信息：

     ```
     git commit -m "feat: initial project structure and basic features from AI agent"
     ```

   - **推送到远程仓库**:

     ```
     # -u 参数会将本地的 main 分支与远程的 origin/main 分支关联起来，后续推送可简化为 git push
     git push -u origin main
     ```

   - **预期结果**: 刷新你的远程仓库页面，代码已经成功上传

✅ **检查点**: 组长的远程仓库已有初始代码。

Task 3: 跨组协作，提交 PR (Pull Request) 🤝

软件开发不仅要组内协作，更要学会与他人协作。这个环节，我们将为**下一组**的项目贡献代码.

1. **Fork 下一组的项目**:
   - 从在线表格中找到**下一组**的仓库地址（最后一组 fork 第一组）。
   - 打开该仓库页面，点击右上角的 **Fork** 按钮。
   - **预期结果**: 你会在自己的账号下创建一个该项目的“副本”。
2. **克隆你的 Forked 仓库到本地**:
   - 进入你刚刚 Fork 的仓库（注意 URL 是你自己的用户名），复制地址。
   - 在本地克隆这个项目：

     ```
     git clone <你fork的仓库地址>
     ```

3. **为下一组的项目做贡献**:
   - 分别进入backend和frontend目录，`npm install` 安装依赖。
   - 现在，你可以为这个项目做一些改进。可以选择以下任一方式：
     - **增量开发**：在他们的项目基础上，实现一个注册功能？关于注册的需求在requirement_new.md，你可以复制其内容并覆盖项目中原本的requirement.md，然后重复上述 从需求到接口 - 生成测试 - 生成代码的过程。
     - **修复 Bug**: 运行他们的项目，看看有没有明显的 Bug？尝试修复它
   - 你可以**手动修改代码**，也可以**使用 AI 智能体**帮你完成。
4. **推送改动并提交 PR**:
   - 完成代码修改后，提交你的改动：

     ```
     git add .
     git commit -m "fix: resolve login button style issue"
     git push
     ```

     - **注意**: 这里的 `push` 是推送到**你自己 Fork 的仓库**。
   - 推送成功后，刷新你 Fork 的仓库页面，你会看到一个黄色的提示条，点击 **"Contribute" -> "Open pull request"**。
   - 填写清晰的标题和描述，说明你做了什么改进，然后点击 **"Create pull request"**。
   - **预期结果**: 一个 PR (Pull Request) 就成功提交到下一组的原仓库了！

5. **审查并合并 PR (Code Review)**:
    - 现在，回到**你们自己组**的仓库。
    - 在 "Pull requests" 标签页，你应该能看到其他组为你们提交的 PR。
    - **作为组长/组员**，点击进入 PR，仔细审查改动内容 ("Files changed" 标签页)。
    - 你可以对代码行进行评论，提出修改意见。
    - 如果代码没有问题，看起来很棒，就点击 **"Merge pull request"** -> **"Confirm merge"**。
    - **预期结果**: 其他组贡献的代码将成功合并到你们的项目中

🎉 **恭喜你！** 你已经完整体验了一次包含 AI 协同、Git 分支与协作、Code Review 和跨团队贡献的现代化软件开发全流程！