# 第1-2天：基础知识

## 1. 学习节点：DevOps文化和原则、Linux基础

在这两天内，你将了解DevOps的基本概念，并熟悉一些常用的Linux命令。这将帮助你为后续的项目和任务奠定基础。

---

## 2. 项目需求：创建一个简单的“个人网站”项目

目标：设计并开发一个包含静态页面的个人网站，同时使用Git进行版本控制。

具体需求：

- 网站结构：

- 主页（index.html）：介绍你的个人信息，如技能、项目经验等。

- 关于页（about.html）：更详细的个人介绍，如教育背景、爱好等。

- 联系页（contact.html）：提供联系表单，用户可以通过该页面发送消息（静态页面即可，无需后端逻辑）。

- 设计要求：

- 使用HTML和CSS设计网页。

- 网站要有一个简洁且易于阅读的布局（可以参考一些现成的模板）。

- 页面应包括导航栏，确保用户可以在不同页面之间轻松切换。

- 版本控制：

- 初始化Git本地仓库。

- 创建多个Git提交（commit），分别保存不同开发阶段的进度，如“创建主页”、“添加CSS样式”、“创建关于页”等。

- 将本地仓库上传至GitHub，并确保你的代码托管在一个公开仓库中（便于展示和未来迭代）。

---

## 3. 涉及知识点：（详细内容已经GPT归档）

A. DevOps的基本理念和流程

- DevOps文化：强调开发（Dev）与运维（Ops）之间的协作，目标是提高软件交付的速度和质量。

- DevOps的核心原则：

- 持续集成（CI）：频繁将代码集成到主分支，自动测试，尽早发现问题。

- 持续交付（CD）：确保代码可以随时部署到生产环境。

- 自动化：尽可能减少手动操作，提高效率。

- 监控与反馈：对系统进行持续监控，快速响应和改进。

B. Linux基础命令

- 文件操作：`ls`, `cd`, `pwd`, `mkdir`, `touch`, `cp`, `mv`, `rm`

- 权限管理：`chmod`, `chown`

- 查看文件内容：`cat`, `less`, `more`, `nano`

- 系统管理：`sudo`, `apt update`, `apt install`

C. Git基本操作

- 克隆仓库：`git clone <repository\_url>`

- 初始化仓库：`git init`

- 查看状态：`git status`

- 添加更改：`git add <file>` 或 `git add .`

- 提交更改：`git commit -m "commit message"`

- 推送到远程仓库：`git push origin main`

- 创建和切换分支：`git branch <branch\_name>`, `git checkout <branch\_name>`

---

## 4. 迭代方向：

- 第一阶段（当前）：创建一个静态网站。学习并掌握HTML、CSS基础，并熟练使用Git进行版本控制。

- 第二阶段（未来的扩展）：

- 将静态网站转变为动态网站：

- 学习使用Flask或Django（Python Web框架）。

- 添加后端功能，如动态内容渲染、用户表单提交（如“联系页”）。

- 实现数据库支持，如使用SQLite或MySQL。

- 部署网站到云服务器（如AWS、Azure、Heroku）并设置CI/CD流水线，实现自动化部署。

---

## 作业要求：

1. 个人网站项目：

- 完成“主页”、“关于页”和“联系页”的开发，并确保在本地测试正常。

- 将代码提交至GitHub，提供GitHub仓库的链接。

2. 学习总结（文档形式）：

- 简要总结DevOps的基本理念及其重要性。

- 列出常用的Linux命令及其用途（至少5个）。

- 简述Git的基本操作流程，并说明其在项目管理中的重要性。

3. 提交方式：

- 提交一个包含项目代码的GitHub仓库链接。

- 提交一份PDF格式的学习总结（可以直接写在README.md文件中）。

完成这两天的学习任务后，你将掌握网站开发的基本流程，以及DevOps文化的基础知识。这些技能将为你未来的运维和开发工作提供支持。

# 任务总结

1. 学习节点：

- DevOps文化和原则：

- 强调开发与运维协作，目标是提高软件交付速度和质量。

- 核心理念包括：持续集成（CI）、持续交付（CD）、自动化、监控与反馈。

- Linux基础：

- 了解文件操作命令（如`ls`, `cd`, `mkdir`, `cp`等）。

- 掌握权限管理（如`chmod`, `chown`）。

2. 项目需求：创建一个简单的“个人网站”

- 网站功能：

- 主页（index.html）：介绍个人信息。

- 关于页（about.html）：详细介绍教育背景、技能等。

- 联系页（contact.html）：静态表单用于用户填写联系信息。

- 技术栈：

- 使用HTML和CSS设计页面。

- 通过Git进行版本控制，创建并管理项目。

3. Git基本操作：

- 初始化仓库：`git init`

- 查看状态：`git status`

- 提交更改：`git add .` → `git commit -m "message"`

- 推送到GitHub：`git push origin main`

4. 项目迭代方向：

- 短期目标：掌握静态网站开发及Git操作。

- 长期目标：将项目升级为动态网站，学习Flask或Django，实现后端功能并部署到云端。

---

作业要求

1. 项目代码：完成静态网站并提交至GitHub。

2. 学习总结：简述DevOps理念、Linux常用命令及Git基本操作流程。提交至README.md或PDF文件。

通过这两天的学习，你将掌握Web开发基础和版本控制技能，为后续更复杂的项目打下基础。

# 实际开发流程

1. 下载ubuntu和vmwave
2. GitHub账号配置（SSH密钥）

分为windows和linux，生成ssh密钥，并且在github上添加密钥

1. 创建项目目录

mdkir

初始化 Git 仓库

git init

1. 创建静态个人网站

-创建编辑HTML 文件

touch index.html about.html contact.html

nano index.html

1. 使用Git进行版本控制

1. 查看当前文件状态

git status

2. 添加文件到 Git 仓库

git add .

3. 提交文件

git commit -m "Initial commit - Added static website files"

遇到问题：Git 还没有配置我的用户身份信息,按照提示创建完成.

我的用户名是Amy-by.

1. 将ubuntu虚拟机中的文件连接到windows的vscode中运行