## 实验报告：DevOps 基础知识与个人网站开发

---

## 实验目的

1. 掌握 DevOps 基础概念和原则，学习 DevOps 文化和流程，理解持续集成（CI）、持续交付（CD）等核心理念。

2. 熟悉 Linux 基础命令，掌握文件操作、权限管理等常用命令，为后续的开发和运维任务打下基础。

3. 学习 Git 版本控制，通过 Git 管理项目代码版本，掌握代码提交、分支管理及与 GitHub 仓库的协作流程。

4. 设计并开发一个静态个人网站，包括创建主页（index.html）、关于页（about.html）和联系页（contact.html），并通过 GitHub 托管项目。

---

## 实验内容

1. DevOps 基础知识学习

- 学习 DevOps 文化与核心原则，了解如何通过自动化、持续集成和监控提升软件交付效率。

- 熟悉常用的 Linux 命令，包括文件操作、权限管理、系统更新等。

- 学习 Git 的基础操作，包括初始化仓库、提交更改、创建分支及推送代码至远程仓库。

2. 静态个人网站项目开发

- 项目需求：

- 主页（index.html）：展示个人简介、技能和项目经验。

- 关于页（about.html）：提供详细的个人介绍，包括教育背景和爱好。

- 联系页（contact.html）：创建静态联系表单，用户可以通过此页面发送消息（无需后端逻辑）。

- 技术栈：HTML、CSS、Git、GitHub

- 版本控制：使用 Git 管理代码，并将项目托管到 GitHub 公共仓库。

---

## 实验步骤

第一部分：基础环境配置

1. 安装 Ubuntu 虚拟机

- 下载并安装 VMware Workstation。

- 下载 Ubuntu 系统镜像并安装在 VMware 中。

2. GitHub 账号配置

- 在 Windows 和 Ubuntu 中生成 SSH 密钥：

```bash

ssh-keygen -t rsa -b 4096 -C "your-email@example.com"

```

- 将生成的公钥（`~/.ssh/id\_rsa.pub`）添加到 GitHub 的 SSH Key 中，以便通过 SSH 协议进行认证。

第二部分：创建项目目录及初始化 Git 仓库

1. 在 Ubuntu 虚拟机中创建项目目录

```bash

mkdir personal-website

cd personal-website

```

2. 初始化 Git 仓库

```bash

git init

```

第三部分：开发静态个人网站

1. 创建 HTML 文件

- 创建必要的文件：

```bash

touch index.html about.html contact.html

```

- 使用文本编辑器（如 nano）编辑文件内容：

```bash

nano index.html

```

2. 示例：`index.html` 内容

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>My Personal Website</title>

</head>

<body>

<h1>Welcome to My Personal Website</h1>

<p>Hi, I'm Amy! This is my portfolio website.</p>

<nav>

<a href="about.html">About Me</a> |

<a href="contact.html">Contact</a>

</nav>

</body>

</html>

```

第四部分：使用 Git 进行版本控制

1. 查看当前文件状态

```bash

git status

```

2. 添加文件并提交到 Git 仓库

```bash

git add .

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

```

- 问题：提交代码时提示 `Please tell me who you are` 错误。

- 解决方案：配置 Git 用户名和邮箱：

```bash

git config --global user.name "Amy-by"

git config --global user.email "your-email@example.com"

```

3. 将代码推送到 GitHub

- 添加远程仓库：

```bash

git remote add origin git@github.com:Amy-by/personal-website.git

```

- 推送代码到 GitHub：

```bash

git push -u origin master

```

第五部分：在 Windows VS Code 中连接 Ubuntu 虚拟机

1. 查看 Ubuntu 虚拟机 IP 地址

```bash

ip addr show

```

- 获取 IP 地址（例如：`192.168.218.201`）。

2. 在 Windows 中安装 VS Code 及 Remote - SSH 插件

- 按 `Ctrl + Shift + P` 打开命令面板，选择 `Remote-SSH: Connect to Host...`。

- 输入 `ssh gec@192.168.218.201`，连接到 Ubuntu 虚拟机。

- 成功连接后，可以在 VS Code 中直接编辑 Ubuntu 中的文件。

---

## 实验中出现的问题及解决方法

1. 问题一：`appstreamcli` 错误

- 现象：执行 `sudo apt update` 时出现 `GLib-ERROR` 相关报错。

- 解决方法：尝试以下命令修复包管理器：

```bash

sudo apt update --fix-missing

sudo apt install --reinstall appstream

sudo apt upgrade

```

2. 问题二：GitHub 连接失败

- 现象：推送代码时提示 `Repository not found`。

- 原因：远程仓库 URL 配置错误。

- 解决方法：重新设置远程仓库：

```bash

git remote set-url origin git@github.com:Amy-by/personal-website.git

```

3. 问题三：VS Code 连接 Ubuntu 虚拟机失败

- 现象：尝试通过 SSH 连接时提示 `Connection timed out`。

- 解决方法：确保虚拟机网络连接正常，并关闭防火墙或启用端口转发。

---

## 实验总结

- 收获：

- 了解了 DevOps 文化及其核心原则，掌握了持续集成与交付的基本概念。

- 熟悉了常用 Linux 命令，掌握了基本的文件操作和权限管理。

- 掌握了 Git 版本控制及 GitHub 远程仓库管理流程。

- 成功开发了一个静态个人网站，并使用 Git 进行版本控制与托管。

- 改进方向：

- 下一阶段可以考虑将静态网站转变为动态网站，使用 Flask 或 Django 等框架进行后端开发。

- 学习并配置 CI/CD 流水线，实现自动化部署。

- 进一步优化 Ubuntu 环境，解决系统更新中的潜在问题。

---

GitHub 项目仓库链接：[personal-website](https://github.com/Amy-by/personal-website)

通过本次实验，掌握了 Web 开发的基本技能，为后续更复杂的 DevOps 项目打下了坚实的基础。