Selected files

1 printable files

LAB 5 .md

LAB 5 .md

Objective: Build a starter project environment automatically.

starter_kit.sh

```
#!/bin/bash

# Create project directories
mkdir -p project/scripts project/docs project/data

# Add placeholder README.md in each folder
for folder in project project/scripts project/docs project/data; do
    echo "# README for $(basename "$folder")" > "$folder/README.md"
done
```

Print completion message

echo "Starter Kit Ready!" How it works: *mkdir -p* ensures parent directories are created if they don't exist. The for loop adds a README.md in each folder with a placeholder title. Finally, it prints Starter Kit Ready!. LAB_extra.md

Sample Output


LAB_extra

Purpose of Script

The `starter_kit.sh` script automates the creation of a starter project structure. It:

- Creates a `project/` folder with subfolders: `scripts/`, `docs/`, `data/`.

- Adds a placeholder `README.md` file in each folder.
- Prints a confirmation message after successful execution.

This saves time and ensures consistent project folder setup.

Example Run

$ bash starter_kit.sh Starter Kit Ready! **Folder structure after running:** project/

- ├── README.md
- ├── scripts/
- │ └── README.md
- ├── docs/
- │ └── README.md
- └── data/
- └── README.md

Extra Questions

1. What does mkdir -p do? ·*mkdir* is the command to make a new directory (folder). ·The *-p* flag stands for “parents”, and it has two main effects: ·Creates parent directories if they don’t exist. ·Without -p, *mkdir* project scripts would give an error if project doesn’t already exist. ·Doesn’t complain if the directory already exists. Normally, mkdir foldername fails if the folder already exists. With -p, it just moves on quietly.

2. Why is automation useful in DevOps? -DevOps focuses on collaboration, speed, and reliability in software development and deployment. Automation is a core part because it:

- Saves Time
- Repetitive tasks like creating environments, deploying code, or running tests can be automated.
- Reduces manual work so teams can focus on actual development.
- Reduces Human Error
- Manual tasks are prone to mistakes (typos, missing steps).
- Scripts and automation tools ensure tasks are executed consistently every time.
- Speeds Up Delivery
- Automation allows continuous integration and continuous deployment (CI/CD).
- Software updates can reach users faster and more reliably.
- Ensures Consistency Across Environments
- Development, testing, and production environments can be automatically configured the same way.
- Avoids the “it works on my machine” problem.
- Scales Easily
- Tasks that are simple for one system can be automatically applied to hundreds or thousands of servers.