✅LAB 5 - Starter Kit and Automation

🎯 Purpose

The `starter_kit.sh` script is a shell automation tool that helps set up a basic project structure quickly and consistently. It is especially useful for data science or software projects that require organized folders from the beginning.

This script creates a folder named `project/` with three subdirectories: `scripts/`, `docs/`, and `data/`. Each subdirectory also contains a placeholder `README.md` file to describe its purpose.

By automating this repetitive task, the script saves time and reduces setup errors.

📁 What It Creates

After running the script, this folder structure is created:

```
project/
├── scripts/
│  └── README.md
├── docs/
│  └── README.md
└── data/
└── README.md
```

Each `README.md` file contains a simple title indicating the folder's purpose.

🧪 Example Run

```
#!/bin/bash
# ==========================================================
# Data Science Starter Pack - Full Setup Script
# Run as: bash starter_pack.sh
# ==========================================================

set -e

# === Base directory ===
BASE_DIR="/home/vibhu0077/ds_starter"
APP_DIR="$BASE_DIR/app"
VENV_DIR="$BASE_DIR/.venv"

echo "📂 Base Directory: $BASE_DIR"
echo "📂 App Directory: $APP_DIR"
echo "🛠 Virtual Environment: $VENV_DIR"
```

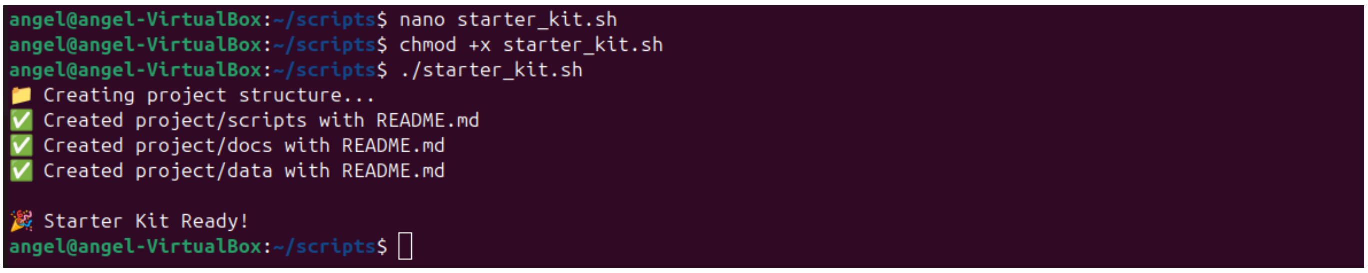
```
# === Step 1: Create base and app directories ===
if [ ! -d "$BASE_DIR" ]; then
  echo "➡ Creating base directory..."
  mkdir -p "$BASE_DIR"
else
  echo "✅ Base directory already exists."
fi

if [ ! -d "$APP_DIR" ]; then
  echo "➡ Creating app directory..."
  mkdir -p "$APP_DIR"
else
  echo "✅ App directory already exists."
fi

cd "$BASE_DIR"


# Step 1: Install required packages
sudo apt-get update
#sudo apt-get install -y python3 python3-venv python3-pip


# === Step 2: Create virtual environment ===
if [ ! -d "$VENV_DIR" ]; then
  echo "➡ Creating hidden virtual environment..."
  python3 -m venv "$VENV_DIR"
else
  echo "✅ Virtual environment already exists."
fi

# === Step 3: Activate environment ===
echo "➡ Activating virtual environment..."
source "$VENV_DIR/bin/activate"

# === Step 4: Upgrade pip ===
echo "➡ Upgrading pip..."
pip install --upgrade pip

# === Step 5: Install packages ===
echo "➡ Installing Streamlit and libraries..."
pip install streamlit pandas numpy matplotlib seaborn

# === Step 6: Create sample Streamlit app ===
APP_FILE="$APP_DIR/app.py"
echo "➡ Creating Streamlit app at $APP_FILE..."

cat > "$APP_FILE" << 'EOF'
import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
st.set_page_config(page_title="Starter Pack Dashboard", layout="wide")

st.title("📊 Data Science Starter Pack - Visualizations")

tab1, tab2, tab3, tab4 = st.tabs(["Random Data", "Line Plot", "Histogram",
"Heatmap"])

# Tab 1: Show random dataset
with tab1:
    st.header("Random Data Preview")
    df = pd.DataFrame(np.random.randn(20, 5), columns=list("ABCDE"))
    st.write(df)

# Tab 2: Line Plot
with tab2:
    st.header("Line Plot Example")
    x = np.linspace(0, 10, 100)
    y = np.sin(x)
    fig, ax = plt.subplots()
    ax.plot(x, y, label="sin(x)")
    ax.legend()
    st.pyplot(fig)

# Tab 3: Histogram
with tab3:
    st.header("Histogram Example")
    data = np.random.randn(1000)
    fig, ax = plt.subplots()
    ax.hist(data, bins=30, alpha=0.7)
    st.pyplot(fig)

# Tab 4: Heatmap
with tab4:
    st.header("Heatmap Example")
    corr = df.corr()
    fig, ax = plt.subplots()
    sns.heatmap(corr, annot=True, cmap="coolwarm", ax=ax)
    st.pyplot(fig)
EOF

# === Step 7: Success message ===
echo "✅ Setup complete!"
echo "To activate environment later, run:"
echo "source $VENV_DIR/bin/activate"
echo ""
echo "🚀 To start the Streamlit app, run:"
echo "streamlit run $APP_FILE"
```

✅ Command:

```
./starter_kit.sh
```

📸Output image of the code :



```
angel@angel-VirtualBox:~/scripts$ nano starter_kit.sh
angel@angel-VirtualBox:~/scripts$ chmod +x starter_kit.sh
angel@angel-VirtualBox:~/scripts$ ./starter_kit.sh
📁 Creating project structure...
✅ Created project/scripts with README.md
✅ Created project/docs with README.md
✅ Created project/data with README.md

🎉 Starter Kit Ready!
angel@angel-VirtualBox:~/scripts$
```

✅ Summary

This lab demonstrates how automation with shell scripts can improve efficiency during project setup. The starter_kit.sh script ensures a consistent starting structure for any new project with minimal manual effort.

?Extra Questions

Q1. What does `mkdir -p` do?

The `mkdir` command is used to create directories in Unix/Linux systems.

The `-p` flag stands for **"parents"**, and it has two key benefits:

1. **Creates parent directories as needed**:

   If the parent directories do not exist, `mkdir -p` will create them automatically.

   Example:

   ```
   mkdir -p project/scripts
   ```

This command will:

- `Create the project/ directory if it doesn't exist.`
- `Then create the scripts/ subdirectory inside it.`
- `No error if directory exists:`
- `If the directory already exists, mkdir -p will not throw an error. It just moves on silently.`

Q2. Why is Automation Useful in DevOps?

⚙ What is DevOps?

DevOps is a set of practices that combines software development (**Dev**) and IT operations (**Ops**). Its goal is to shorten the software development lifecycle and deliver high-quality software continuously.

🚀 Why Automation Matters

Automation is one of the **core pillars** of DevOps. It enables faster, more consistent, and more reliable workflows by reducing manual effort.

✅ Benefits of Automation in DevOps

1. **Speed and Efficiency**

- Repetitive tasks like testing, building, and deploying are done automatically.
- Saves time and reduces manual workload.

2. **Consistency and Reliability**

- Automation ensures the same process runs every time, reducing human errors.
- Configuration and deployment steps are repeatable and version-controlled.

3. **Scalability**

- Automation can handle large-scale environments effortlessly.
- Infrastructure can be replicated across multiple environments with tools like Terraform or Ansible.

4. **Continuous Integration / Continuous Deployment (CI/CD)**

- Automation is critical for CI/CD pipelines.
- Code changes are tested and deployed automatically, allowing for rapid and safe releases.

5. **Improved Collaboration**

- Dev and Ops teams share the same tools and workflows.
- Automated pipelines reduce friction and encourage better communication.

🧪 Real-World Automation Examples

Task	Automated Tool Example
Code Testing	GitHub Actions, Jenkins
Deployment	Docke