Title : Developer-Tool-Kit

Unix Programming Lab

Course Code : 24B15CS214

ODD-Semester-3

Batch B5

Team members :

Sara Jain (2401030235)

Mukul Aggarwal (2401030239)

Letter of Transmittal

Amarjeet Kaur,
Department of Computer Science
Jaypee Institute of Information Technology
Sector–62, Noida

Subject: Submission of Project Report on "Developer-Tool-Kit"

Dear **sir/mam**

We are pleased to submit our project report titled **"Developer Tool Kit"**, developed as part of the UNIX Lab coursework for the B.Tech CSE program.

This project focuses on building a set of **five practical shell scripts** designed to serve as a **daily-use productivity toolkit for developers**. Each script addresses a common real-world requirement—such as automation, debugging assistance, task management, system checks, and workflow optimization—making development activities faster, simpler, and more efficient.

The report includes detailed explanations of the purpose of each script, system design, logic implementation, testing procedures, challenges encountered during development, and the potential enhancements that can be incorporated in future versions.

We sincerely hope that this project meets your expectations and demonstrates our understanding of UNIX shell scripting, automation, and practical problem-solving.

Sincerely,
Sara Jain – 2401030235
Mukul Aggarwal – 2401030239

INTRODUCTION

The **Developer Tool Kit** is a collection of five practical UNIX shell scripts designed to simplify and automate daily tasks performed by developers. These tools aim to improve productivity, reduce repetitive tasks, and enhance the command-line workflow. Each script focuses on a specific real-world requirement—from interacting with AI, to managing version control, organizing directory structures, setting up environments, and maintaining personal task lists.

The scripting approach ensures that all tools remain lightweight, fast, and portable across UNIX-based systems.

REPO LINK - https://github.com/SaraJain90/Developer-Tool-Kit

FEATURES

The toolkit consists of the following five scripts:

1) gpt – AI Assistant in Terminal

A command-line tool that allows developers to interact with **Gemini AI** directly from the terminal. It enables quick problem solving, code suggestions, debugging help, and more without opening a browser.
Usage:

```
gpt "<prompt here>"
```

2) mytree – Directory Structure Viewer

This script generates a clean and readable **tree-like representation** of the current directory. It helps developers quickly understand folder structures and file locations.
Usage:

```
mytree
```

3) setup – Virtual Environment & ML Toolkit Initializer

A convenience script that installs **Python**(latest version), **pip** and virtual environment. It then creates a **Python virtual environment** in the current folder and optionally installs commonly used **AI/ML libraries** such as NumPy, Pandas, Scikit-learn, etc, and **Jupyter Notebook and Lab** as well.
It simplifies the environment setup process for new projects.
Usage:
`setup`

4) gish – Git Automation Script

A smart wrapper combining:

- `git add .`
- `git commit -m "<message>"`
- `git push`

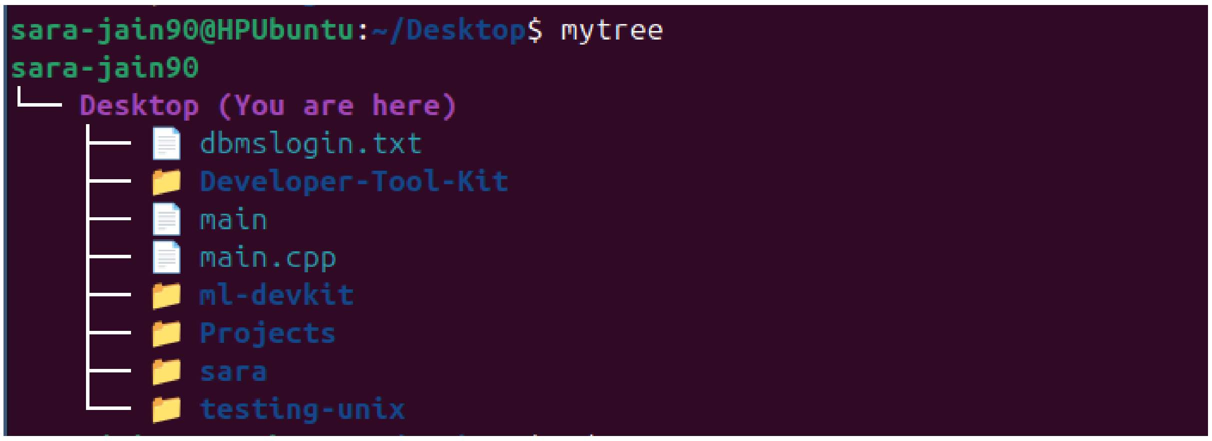
The script interactively asks the user for the commit message and performs all operations in a single step. This saves time and ensures consistent workflow during frequent version control operations.
Usage:
`gish`
(asks for commit message)

5) todo – Lightweight To-Do Manager

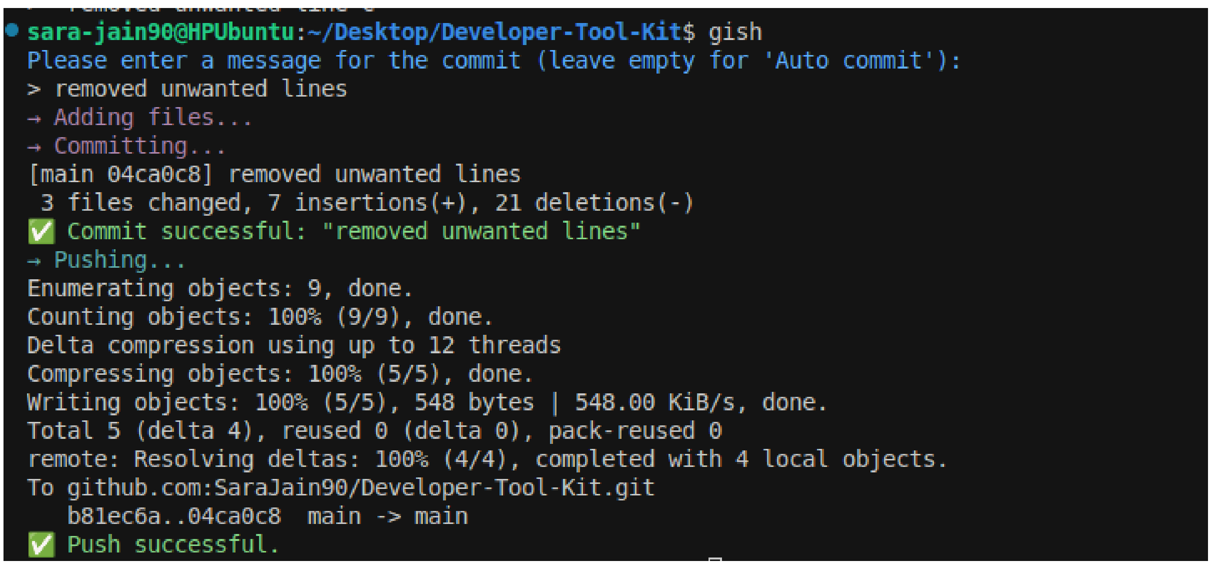
A local task-tracking tool that helps developers maintain personal to-do lists inside their project directory.
Operations include creating tasks, marking them as done, displaying them, and initializing the manager.
Usage:

- `todo init`
- `todo "<task message>"`
- `todo done <task_id>`
- `todo display`

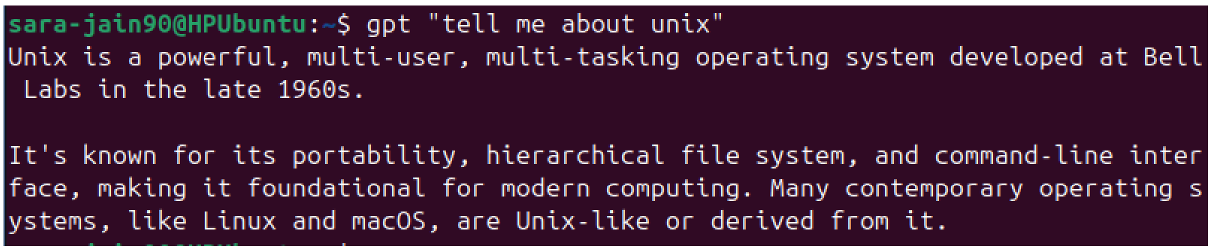
WORKING SCREENSHOTS



```
sara-jain90@HPUbuntu:~/Desktop$ mytree
sara-jain90
└── Desktop (You are here)
    ├── 📄 dbmslogin.txt
    ├── 📁 Developer-Tool-Kit
    ├── 📄 main
    ├── 📄 main.cpp
    ├── 📁 ml-devkit
    ├── 📁 Projects
    ├── 📁 sara
    └── 📁 testing-unix
```

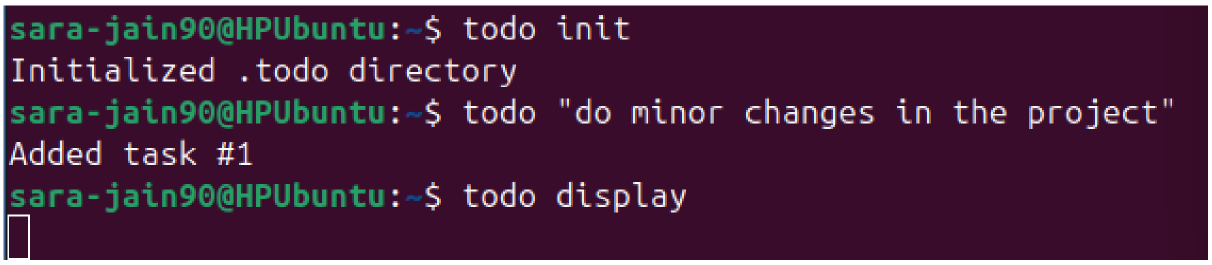


```
● sara-jain90@HPUbuntu:~/Desktop/Developer-Tool-Kit$ gish
  Please enter a message for the commit (leave empty for 'Auto commit'):
  > removed unwanted lines
  → Adding files...
  → Committing...
  [main 04ca0c8] removed unwanted lines
   3 files changed, 7 insertions(+), 21 deletions(-)
  ✅ Commit successful: "removed unwanted lines"
  → Pushing...
  Enumerating objects: 9, done.
  Counting objects: 100% (9/9), done.
  Delta compression using up to 12 threads
  Compressing objects: 100% (5/5), done.
  Writing objects: 100% (5/5), 548 bytes | 548.00 KiB/s, done.
  Total 5 (delta 4), reused 0 (delta 0), pack-reused 0
  remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
  To github.com:SaraJain90/Developer-Tool-Kit.git
     b81ec6a..04ca0c8  main -> main
  ✅ Push successful.
```

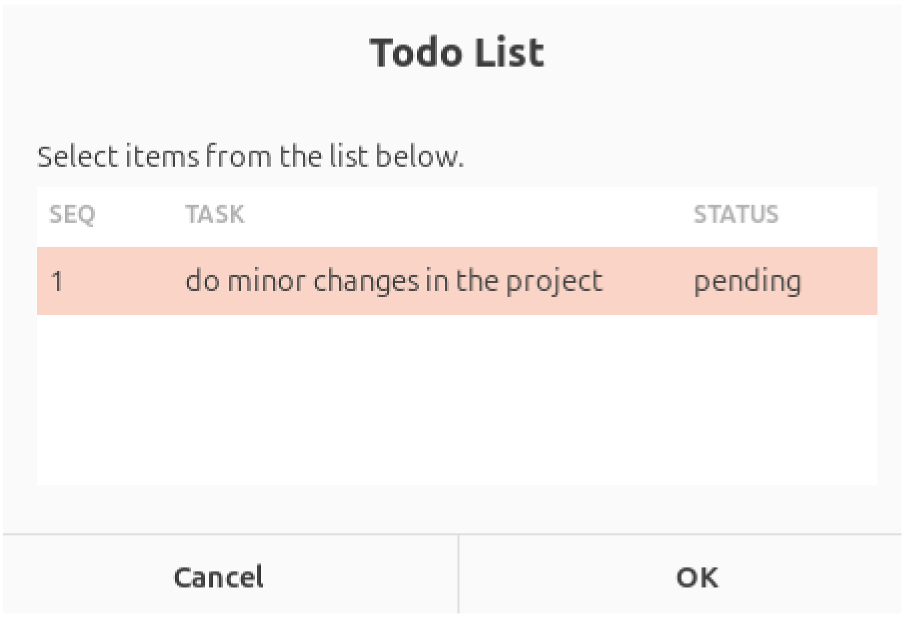


```
sara-jain90@HPUbuntu:~$ gpt "tell me about unix"
Unix is a powerful, multi-user, multi-tasking operating system developed at Bell
 Labs in the late 1960s.

It's known for its portability, hierarchical file system, and command-line inter
face, making it foundational for modern computing. Many contemporary operating s
ystems, like Linux and macOS, are Unix-like or derived from it.
```



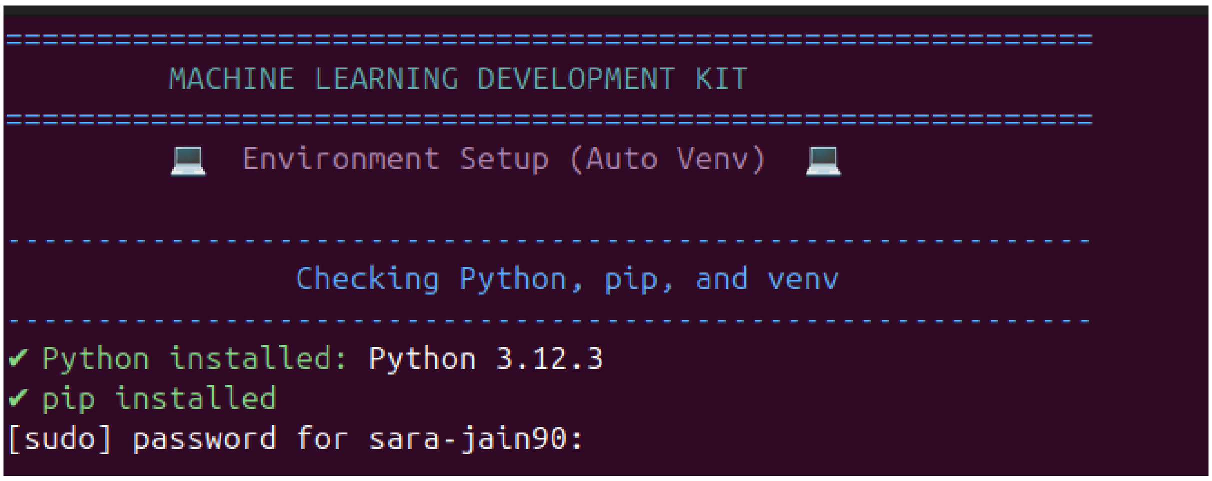
```
sara-jain90@HPUbuntu:~$ todo init
Initialized .todo directory
sara-jain90@HPUbuntu:~$ todo "do minor changes in the project"
Added task #1
sara-jain90@HPUbuntu:~$ todo display
```



Todo List

Select items from the list below.

SEQ	TASK	STATUS
1	do minor changes in the project	pending



```
============================================================
        MACHINE LEARNING DEVELOPMENT KIT
============================================================
        💻  Environment Setup (Auto Venv)  💻

------------------------------------------------------------
                Checking Python, pip, and venv
------------------------------------------------------------
✔ Python installed: Python 3.12.3
✔ pip installed
[sudo] password for sara-jain90:
```

PROJECT DESIGN AND IMPLEMENTATION

The **Developer Tool Kit** follows a modular shell-scripting approach, where each script is designed with a specific purpose while maintaining compatibility and simplicity. The scripts were implemented using **Bash**, leveraging key UNIX concepts such as file handling, process control, command substitution, and user interaction.

1. Script Architecture

Each script follows a structured design pattern:

- Input handling
- Core processing logic
- Validation and error handling
- Output formatting

This ensures consistency across the toolkit and simplifies future enhancements.

---

2. Implementation Details

a) gpt Script

- Uses environment variables to store the Gemini API key.
- Sends user prompt to the API using curl.
- Displays AI-generated output directly in the terminal.
- Includes error handling for missing keys and failed responses.

b) mytree Script

- Displays a clean, **color-coded directory tree** of the current working folder with icons for files and folders.
- Shows the **full path hierarchy** from the user's HOME directory, highlighting the current folder as "You are here".
- Uses intelligent indentation logic (├──, └──, │) to produce a structured, readable tree layout.
- Automatically ignores hidden/system files, ensuring a neat and clutter-free output.

c) setup Script

- Automatically creates a Python virtual environment using `python3 -m venv`.
- Activates the venv and asks user to install preferred AI/ML libraries, and Jupyter Lab and Notebook.
- Installs packages via pip, ensuring a clean and controlled environment.
- Also installs the VS Code and git if not already installed.

d) gish Script

- Uses interactive `read` command to take commit message from user.
- Executes git commands sequentially with error checks.

- Displays colored success/error messages for better usability.

e) todo Script

- Initializes a `.todo` directory and a `todo.db` file.
- Allows adding, listing, and completing tasks.
- Uses timestamps and indexes for task identification.
- Provides persistent storage within the working directory.

CONCLUSION

The **Developer Tool Kit** successfully demonstrates the power and flexibility of UNIX shell scripting in automating real-world developer workflows. By combining multiple productivity tools—AI integration, directory visualization, environment setup, version control automation, and task management—the project showcases practical application of Bash scripting concepts.

The toolkit is lightweight, easy to use, and highly extensible, making it a valuable utility for developers who frequently work on the command line. Future enhancements could include GUI wrappers, cloud sync for todos, or integration with other AI models and additional developer utilities.