**Project Title: Custom Shell Environment Setup Script**

**Overview:**

This script automates common user management tasks on a Linux system, such as creating, removing, and configuring users. It also allows setting up a predefined shell environment with custom tools and configurations.

**Prerequisites:**

* Operating system: Linux
* Sudo Privileges: Sudo(superuser)
* Installed Utilities: Bash,useradd, userdel, passwd etc
* System Configuration: Home Directory Structure, Password Policies

**Technologies** **Used:**

* Bash
* Linux commands
* Custom bash environment

**Features:**

1. **Create a New User**  
   Allows an administrator to create a new user on the system with a home directory, default shell, and a predefined environment setup (e.g., custom .bashrc file and project directories).
2. **List All Users**  
   Displays all users in the system by reading the /etc/passwd file.
3. **Remove an Existing User**  
   Removes a user and their associated home directory from the system.
4. **Configure User Environment**  
   Configures a selected user’s shell environment with default tools and settings (such as a custom .bashrc and project directories).

**Script Functions:**

1. **create\_user()**
   * **Purpose**: To create a new user with a custom environment setup.
   * **Steps**:
     + Prompts for a username.
     + Uses sudo useradd to add the user with a home directory and bash shell.
     + If successful, sets a password for the user and calls setup\_default\_tools() to apply predefined environment configurations.
     + **Error Handling**: If user creation fails, an appropriate message is shown.
2. **list\_users()**
   * **Purpose**: Lists all users on the system.
   * **Steps**:
     + Reads and displays the contents of /etc/passwd, which contains all user accounts.
3. **remove\_user()**
   * **Purpose**: Removes a user from the system.
   * **Steps**:
     + Prompts for a username.
     + Uses sudo userdel -r to delete the user and their home directory.
     + **Error Handling**: If user removal fails, an appropriate message is shown.
4. **configure\_user()**
   * **Purpose**: Configures a shell environment for a selected user.
   * **Steps**:
     + Lists all users and prompts for a username.
     + Checks if the user exists using the id command.
     + If the user exists and has a home directory, it applies custom configurations by calling setup\_default\_tools().
5. **setup\_default\_tools()**
   * **Purpose**: Applies custom environment configurations for a given user.
   * **Steps**:
     + Takes the username as an argument.
     + Sets up a custom .bashrc in the user’s home directory, defining:
       - Custom aliases (e.g., ll, grep, cls, etc.).
       - Custom environment variables (e.g., MY\_PROJECT\_DIR).
       - A custom shell prompt.
     + Ensures proper ownership and permissions of the .bashrc file and user’s project directory (~/projects).

Here’s the detailed documentation for the shell script project you provided:

**User Management and Configuration Script**

**Purpose**

This script automates common user management tasks on a Linux system, such as creating, removing, and configuring users. It also allows setting up a predefined shell environment with custom tools and configurations.

**Features**

1. **Create a New User**  
   Allows an administrator to create a new user on the system with a home directory, default shell, and a predefined environment setup (e.g., custom .bashrc file and project directories).
2. **List All Users**  
   Displays all users in the system by reading the /etc/passwd file.
3. **Remove an Existing User**  
   Removes a user and their associated home directory from the system.
4. **Configure User Environment**  
   Configures a selected user’s shell environment with default tools and settings (such as a custom .bashrc and project directories).

**Script Functions**

1. **create\_user()**
   * **Purpose**: To create a new user with a custom environment setup.
   * **Steps**:
     + Prompts for a username.
     + Uses sudo useradd to add the user with a home directory and bash shell.
     + If successful, sets a password for the user and calls setup\_default\_tools() to apply predefined environment configurations.
     + **Error Handling**: If user creation fails, an appropriate message is shown.
2. **list\_users()**
   * **Purpose**: Lists all users on the system.
   * **Steps**:
     + Reads and displays the contents of /etc/passwd, which contains all user accounts.
3. **remove\_user()**
   * **Purpose**: Removes a user from the system.
   * **Steps**:
     + Prompts for a username.
     + Uses sudo userdel -r to delete the user and their home directory.
     + **Error Handling**: If user removal fails, an appropriate message is shown.
4. **configure\_user()**
   * **Purpose**: Configures a shell environment for a selected user.
   * **Steps**:
     + Lists all users and prompts for a username.
     + Checks if the user exists using the id command.
     + If the user exists and has a home directory, it applies custom configurations by calling setup\_default\_tools().
     + **Error Handling**: If the user or home directory does not exist, an appropriate message is shown.
5. **setup\_default\_tools()**
   * **Purpose**: Applies custom environment configurations for a given user.
   * **Steps**:
     + Takes the username as an argument.
     + Sets up a custom .bashrc in the user’s home directory, defining:
       - Custom aliases (e.g., ll, grep, cls, etc.).
       - Custom environment variables (e.g., MY\_PROJECT\_DIR).
       - A custom shell prompt.
     + Ensures proper ownership and permissions of the .bashrc file and user’s project directory (~/projects).
     + **Error Handling**: Ownership and permission settings ensure that files are correctly set up for the user.

**Predefined Environment Settings (in .bashrc)**

The script configures the following settings in the user's .bashrc file:

1. **PATH Extension**: Adds /usr/local/my\_custom\_bin to the PATH for custom binaries.
2. **Custom Aliases**:
   * ll: Long listing format for ls.
   * grep: Enables color in grep output.
   * cls, c: Clears the terminal.
   * Safe rm, cp, mv: Prompts before actions.
3. **Custom Environment Variables**:
   * EDITOR=nano: Sets the default editor to nano.
   * MY\_PROJECT\_DIR=~/projects: Defines a custom project directory environment variable.
4. **Custom Prompt (PS1)**: Sets the prompt to show username, hostname, and current directory.

**Limitations**

* The script must be run by a specific user (lap), limiting its flexibility.
* It assumes that sudo is configured and available for the current user.
* User creation and removal depend on the system’s configuration (i.e., password complexity policies).

**Conclusion:**

This user management and configuration script is a powerful tool for administrators, providing essential functions like user creation, deletion, and environment configuration. It allows setting up custom user environments efficiently, saving time on repetitive administrative tasks. With basic validation, clear prompts, and predefined shell environments, the script is functional and easy to extend.

Further improvements, such as enhanced error handling, better input validation, logging, and backup options, can increase the robustness of the script. Advanced features like graphical menus, integration with external systems, and role-based setups can make the script more versatile for enterprise-level environments.

By implementing these improvements, you can make the script more user-friendly, efficient, and adaptable to different use cases and system environments.