CompTIA Linux+

Working with the Environment Variables

Introduction

Welcome to the **Working with the Environment Variables** Practice Lab. In this module you will be provided with the instructions and devices needed to develop your hands-on skills.

Environment Variables
Linux System
Permanent Environment Variables

Learning Outcomes

In this module, you will complete the following exercise:

- Exercise 1 - Working with the Environment Variables

After completing this lab, you will be able to:

- Read environment variables
- Set environment variables including the permanent ones

Exam Objectives

The following exam objectives are covered in this lab:

- **LPI:** 103.8 Basic file editing
- **CompTIA:** 2.3 Given a scenario, create, modify, and redirect files.

Note: *Our main focus is to cover the practical, hands-on aspects of the exam objectives. We recommend referring to course material or a search engine to research theoretical topics in more detail.*

Lab Duration

It will take approximately **1 hour** to complete this lab.

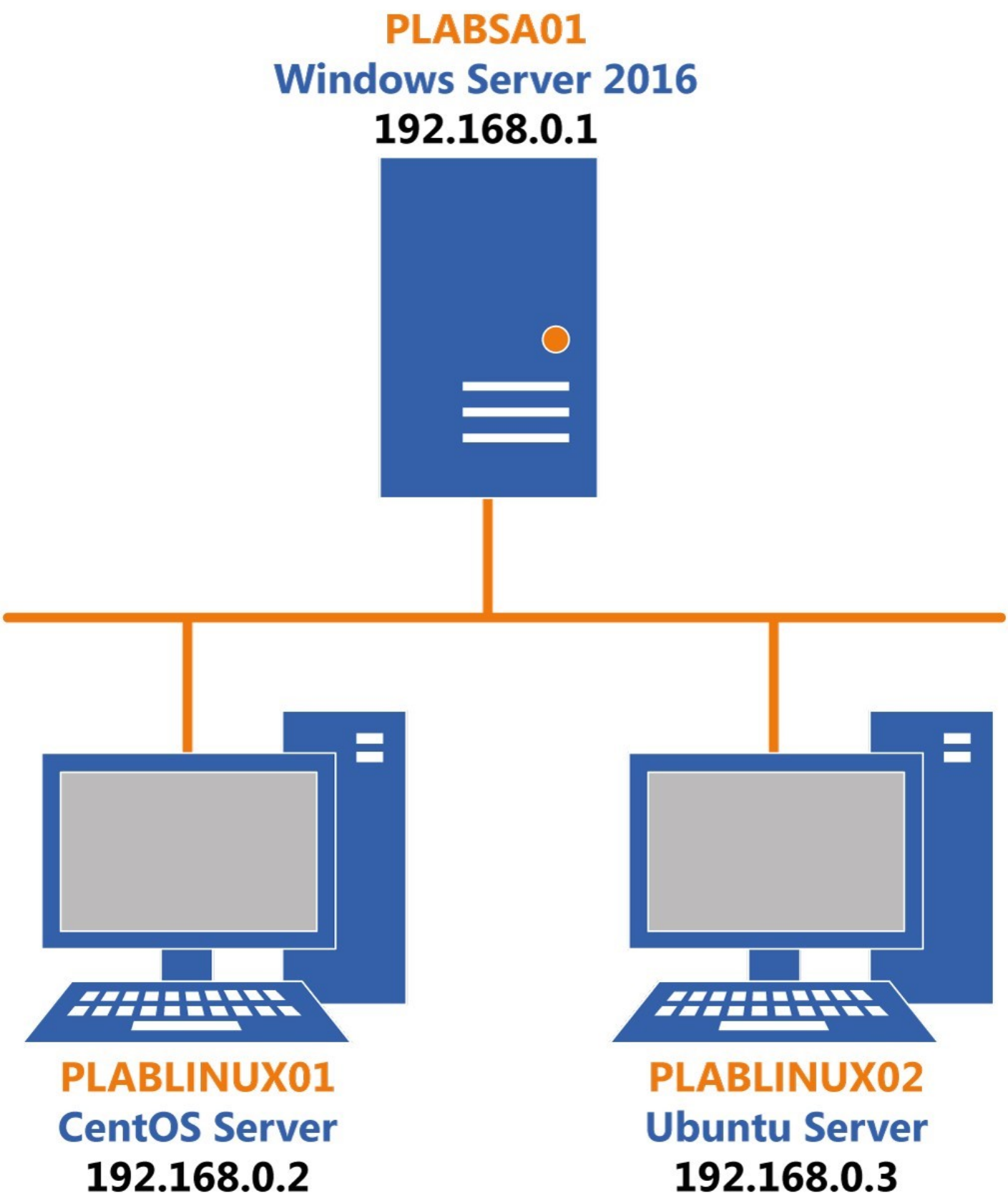
Help and Support

For more information on using Practice Labs, please see our **Help and Support** page. You can also raise a technical support ticket from this page.

Click Next to view the Lab topology used in this module.

Lab Topology

During your session, you will have access to the following lab configuration.



Depending on the exercises you may or may not use all of the devices, but they are shown here in the layout to get an overall understanding of the topology of the lab.

- **PLABSA01** (Windows Server 2016)
- **PLABLINUX01** (CentOS Server)
- **PLABLINUX02** (Ubuntu Server)

Click Next to proceed to the first exercise.

Exercise 1 - Working with the Environment Variables

A variable is mainly used for storing data on a temporary basis in memory. A variable can hold different types of data, such as characters, white spaces, alphanumeric characters, special characters and so on. Variables are of two types:

Environmental variables are defined for the current shell. These variables are then inherited and used by the child shells and the processes. On the other hand, shell variables work within the shell in which they are defined.

In this exercise, you will work with the environment variables.

Learning Outcomes

After completing this exercise, you will be able to:

- Log into a Linux System
- Read environment variables
- Set environment variables including the permanent ones

Your Devices

You will be using the following device in this lab. Please power these on now.

- **PLABLINUX01** (CentOS Server)

Task 1 - Read Environment Variables

Similar to any other operating system, Linux also uses variables, which can either be local or environment variables. The local variable is private within a shell, and it cannot be used outside the shell. This strictly means that any process that is not part of the shell cannot use the local variable, which exists only within the current shell. It is local to the extent that even the sub-shell cannot use it.

In this task, you will read the environment variables.

To read environment variables, perform the following steps:

Step 1

On the desktop, right-click and select **Open Terminal**.

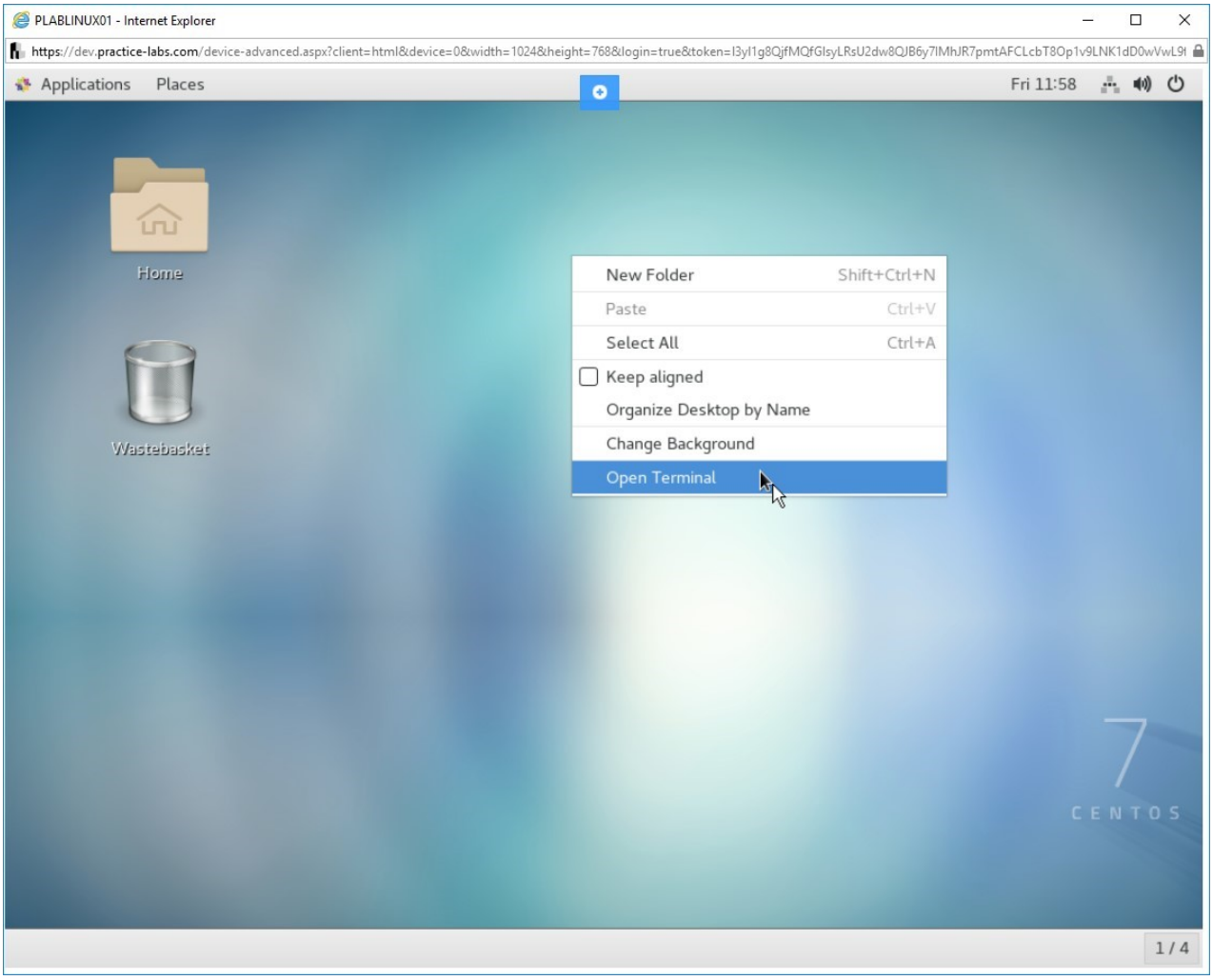


Figure 1.1 Screenshot of PLABLINUX01: Selecting the Open Terminal option from the context menu.

Step 2

The terminal window is displayed. Following are the commands to view and list the environmental variables:

- **printenv**: Prints full or partial environment.
- **env**: Prints all exported environment
- **set**: Prints the name and value of each shell variable available.

To print a full or partial environment, type the following command:

```
printenv
```

Press **Enter**.

> ***Note:*** *You may want to maximize the terminal window.*

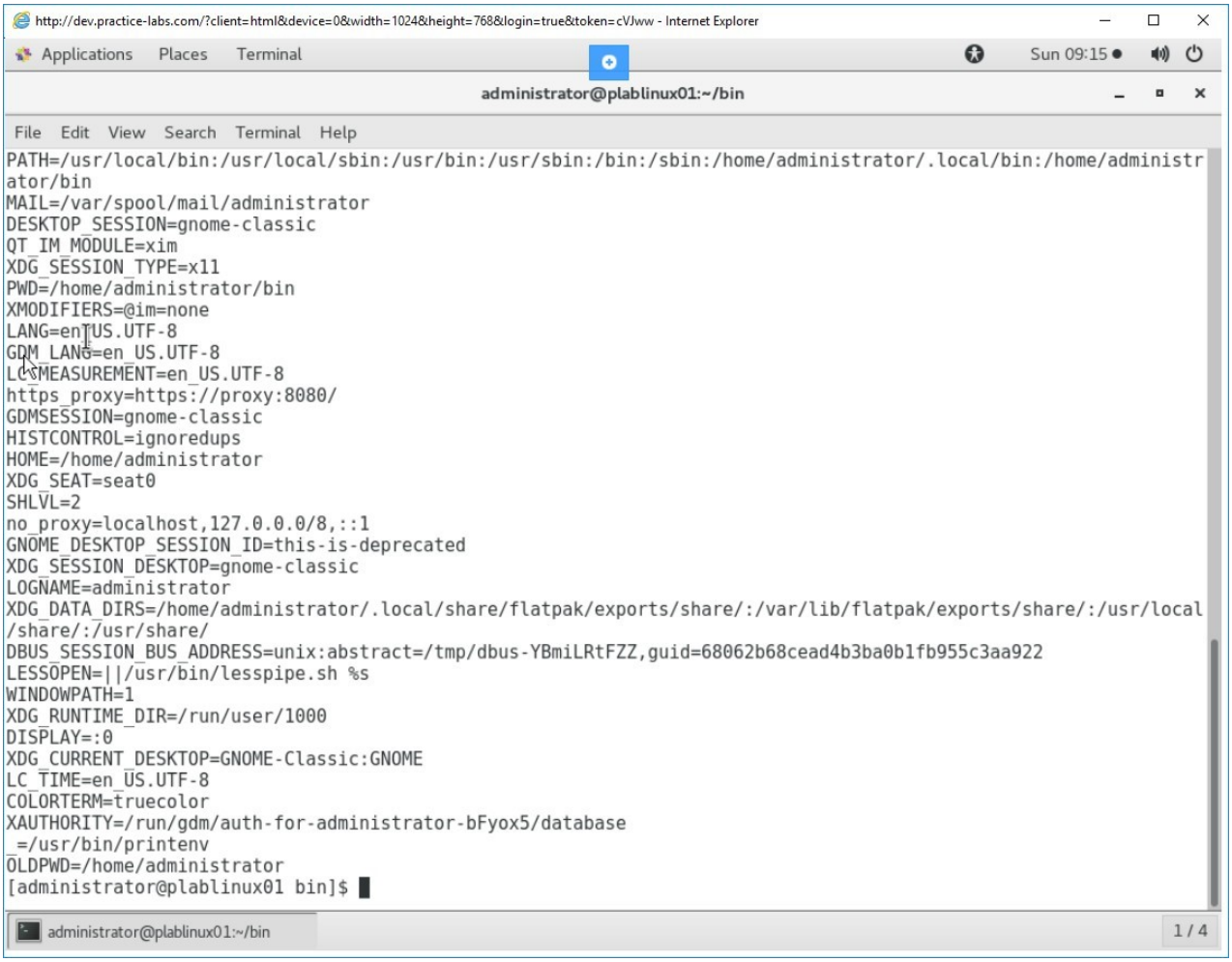


Figure 1.2 Screenshot of PLABLINUX01: Executing the printenv command.

Step 3

You can find several environment variables listed as the output. Scroll up and find HOSTNAME.

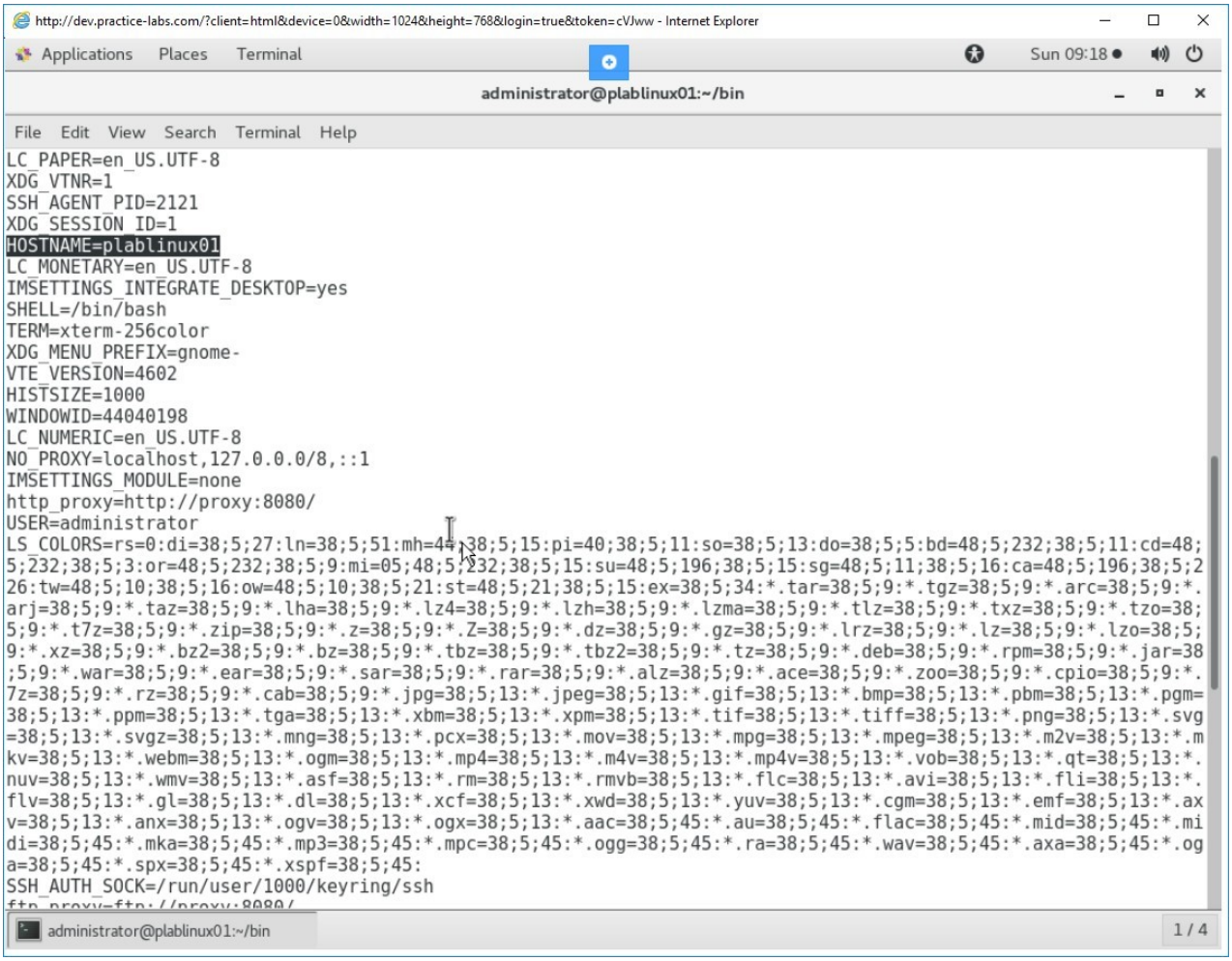


Figure 1.3 Screenshot of PLABLINUX01: Locating the HOSTNAME variable.

Step 4

Clear the screen by entering the following command:

```
clear
```

You can use printenv to display the value of a single variable. To do this, type the following command:

```
printenv HOSTNAME
```

Press **Enter**. Notice that the output of this command is the same as the value assigned to HOSTNAME. Similarly, you can verify the value of each of the variables.

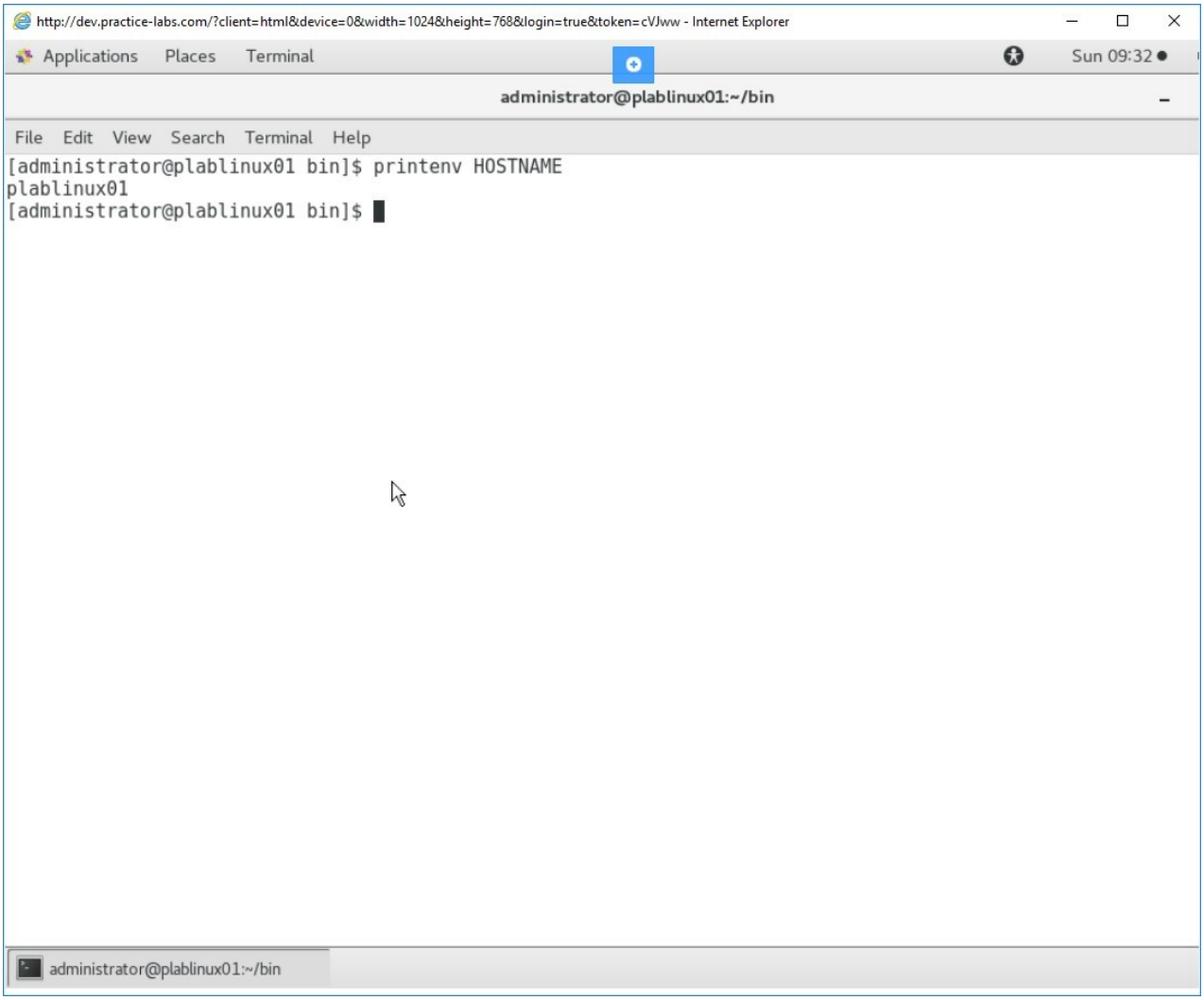


Figure 1.4 Screenshot of PLABLINUX01: Checking the HOSTNAME variable value with the printenv command.

Step 5

Clear the screen by entering the following command:

```
clear
```

You can also use the printenv command with the grep command. To do this, type the following command:

```
printenv | grep HOSTNAME
```

Press **Enter**.

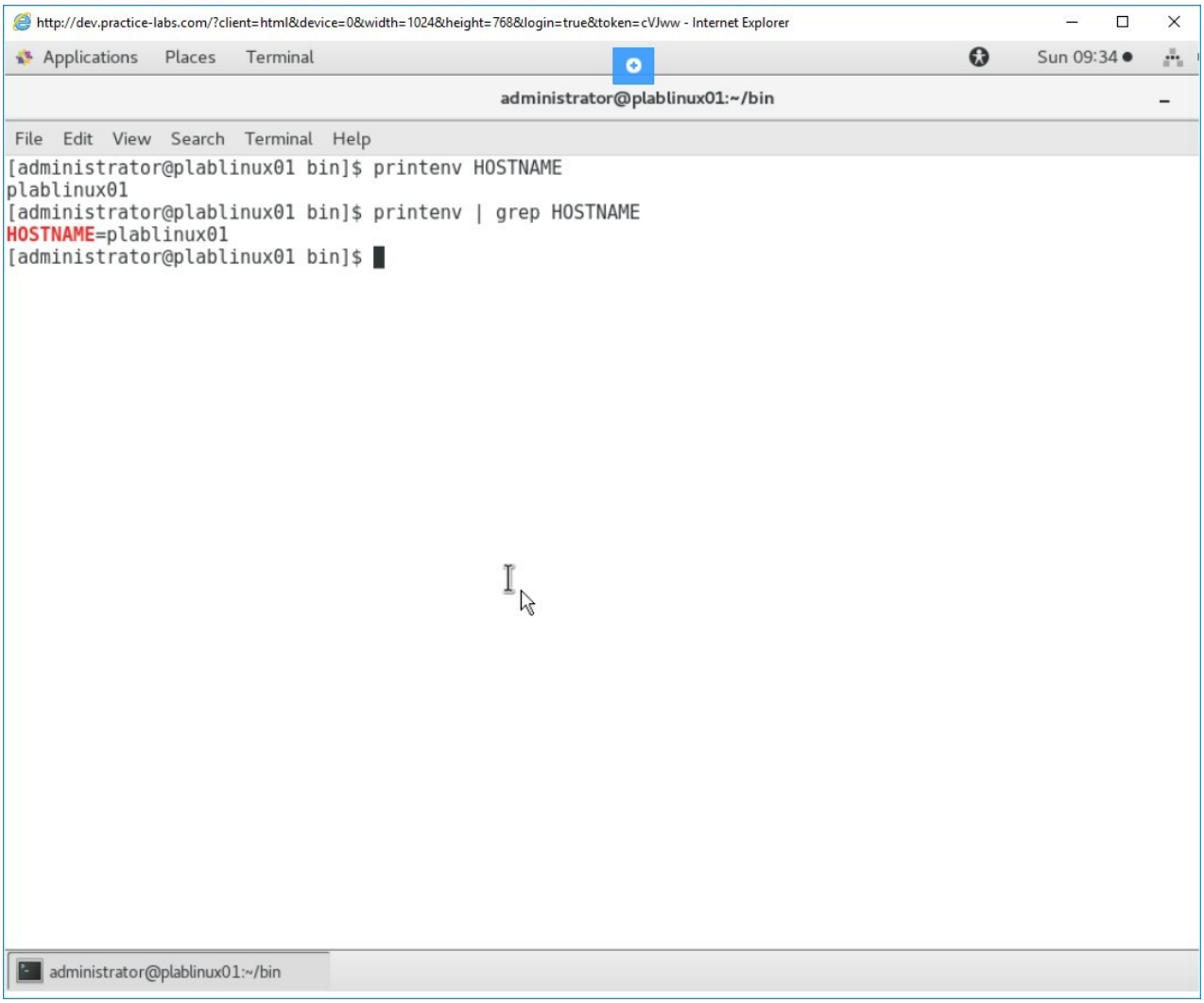


Figure 1.5 Screenshot of PLABLINUX01: Finding the HOSTNAME value with the grep command.

Step 6

Clear the screen by entering the following command:

```
clear
```

You can choose to display the value of the HOSTNAME variable. To do this, type the following command:

```
echo $HOSTNAME
```

Press **Enter**.

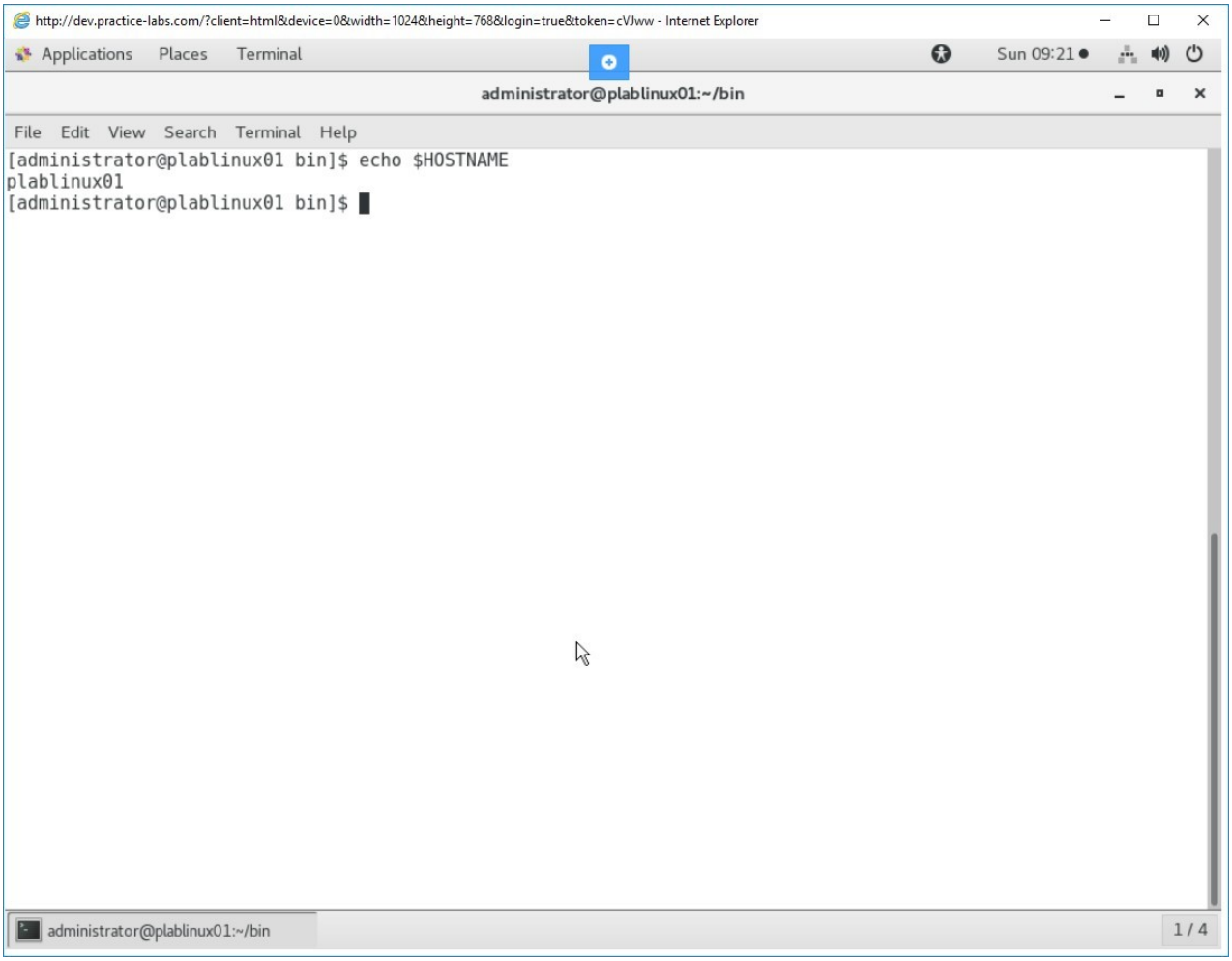


Figure 1.6 Screenshot of PLABLINUX01: Displaying the HOSTNAME variable value.

Step 7

Clear the screen by entering the following command:

```
clear
```

The **env** command displays a list of environment variables that have been exported. However, it does not include all bash variables. Type the following command:

```
env
```

Press **Enter**.

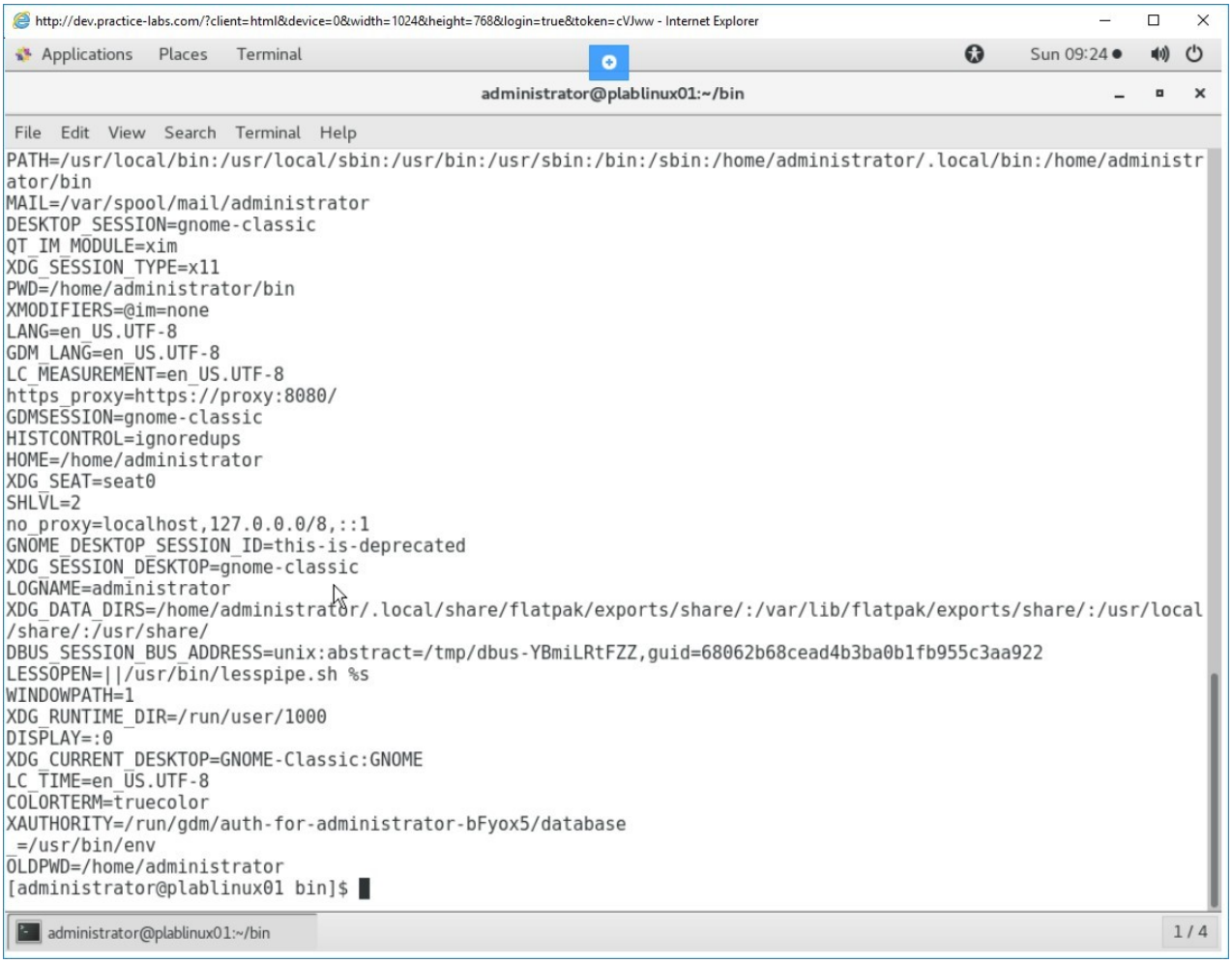


Figure 1.7 Screenshot of PLABLINUX01: Executing the env command.

Step 8

You can find several environment variables listed as the output. Scroll up and find **HOSTNAME**. Notice the value assigned to it.

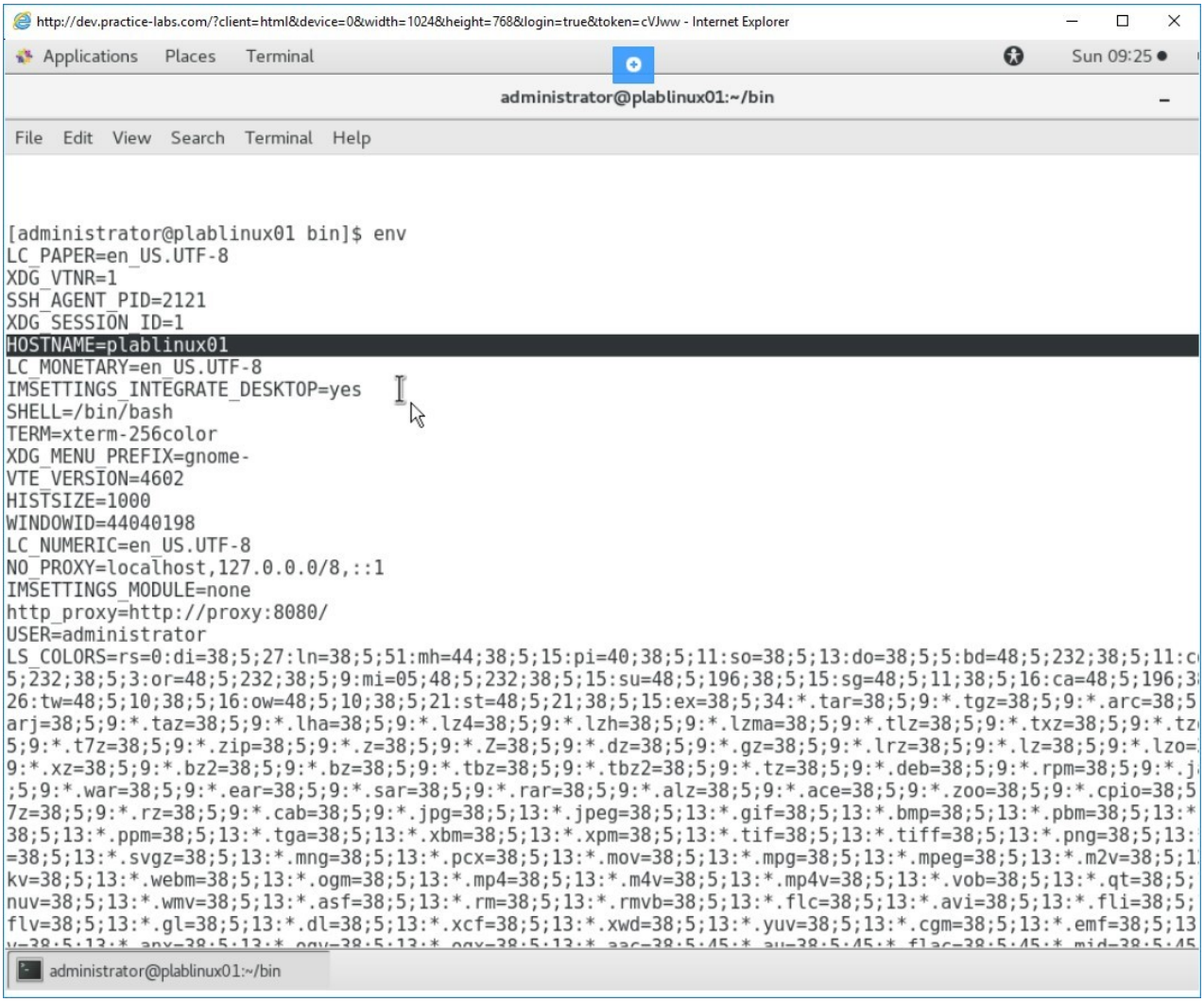


Figure 1.8 Screenshot of PLABLINUX01: Locating the HOSTNAME variable value.

Step 9

Clear the screen by entering the following command:

```
clear
```

The **set** command displays the variables and their values if not argument is supplied to it. With an argument, the set command can be used to change the value of a variable. Type the following command:

```
set
```

Press **Enter**.

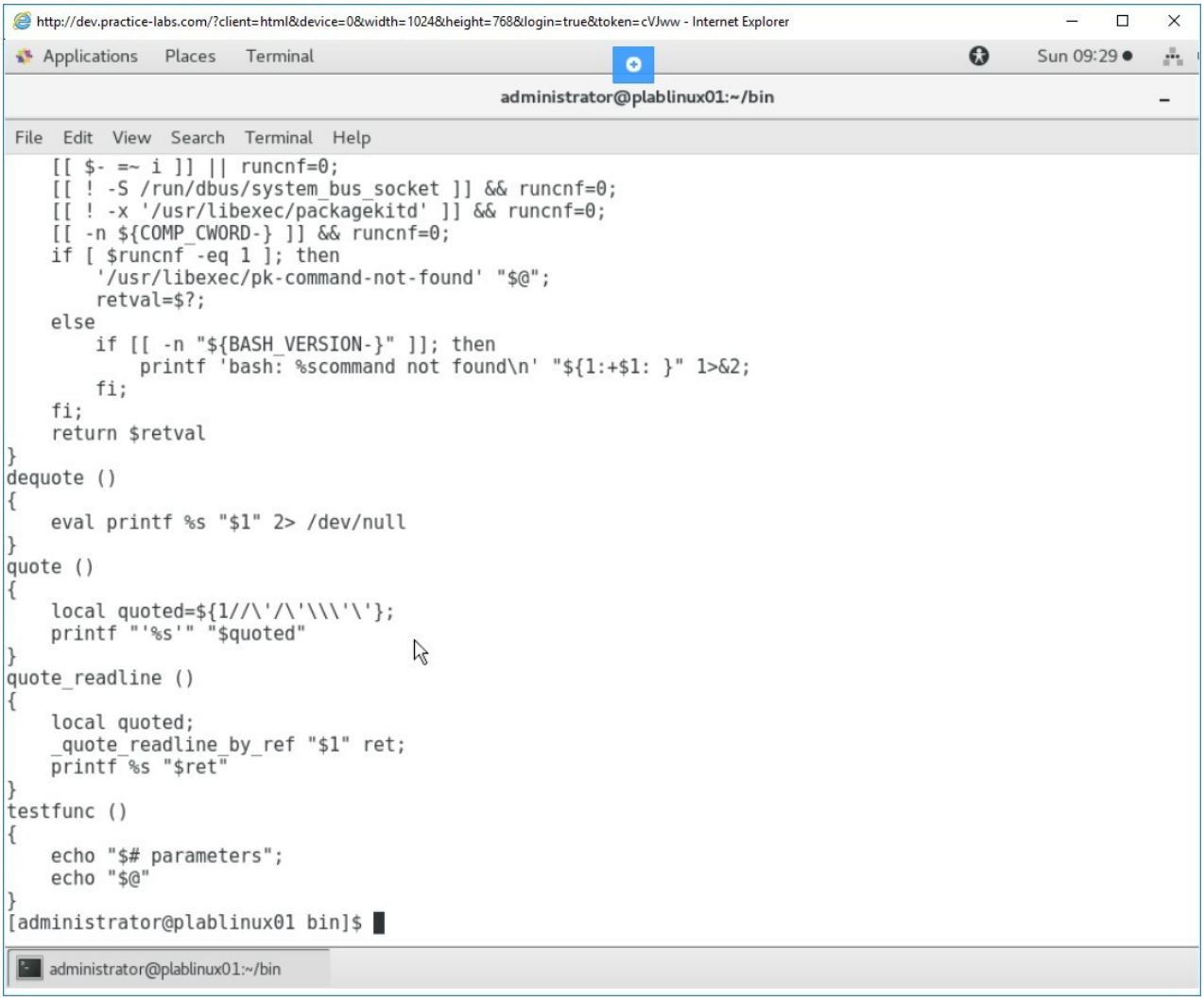


Figure 1.9 Screenshot of PLABLINUX01: Executing the set command.

Task 2 - Setting Environment Variables Including the Permanent Ones

Environment variables can pass the stored value from the current shell to the sub-shell. A variable that is defined in the sub-shell loses its values after the sub-shell terminates. There are a number of environment variables that are predefined by the system and user startup files. Examples include PATH, PS1, PS2, and so on.

To set the environment variables including the permanent ones, perform the following steps:

Step 1

Clear the screen by entering the following command:

```
clear
```

Before an environmental variable is defined, a local variable needs to be created. You can choose to define any variable and define a value along with it. Type the following command:

```
PLAB=ls-l
```

Press **Enter**.

Note: *Variable names are case sensitive. As a standard practice, you should define them in uppercase.*

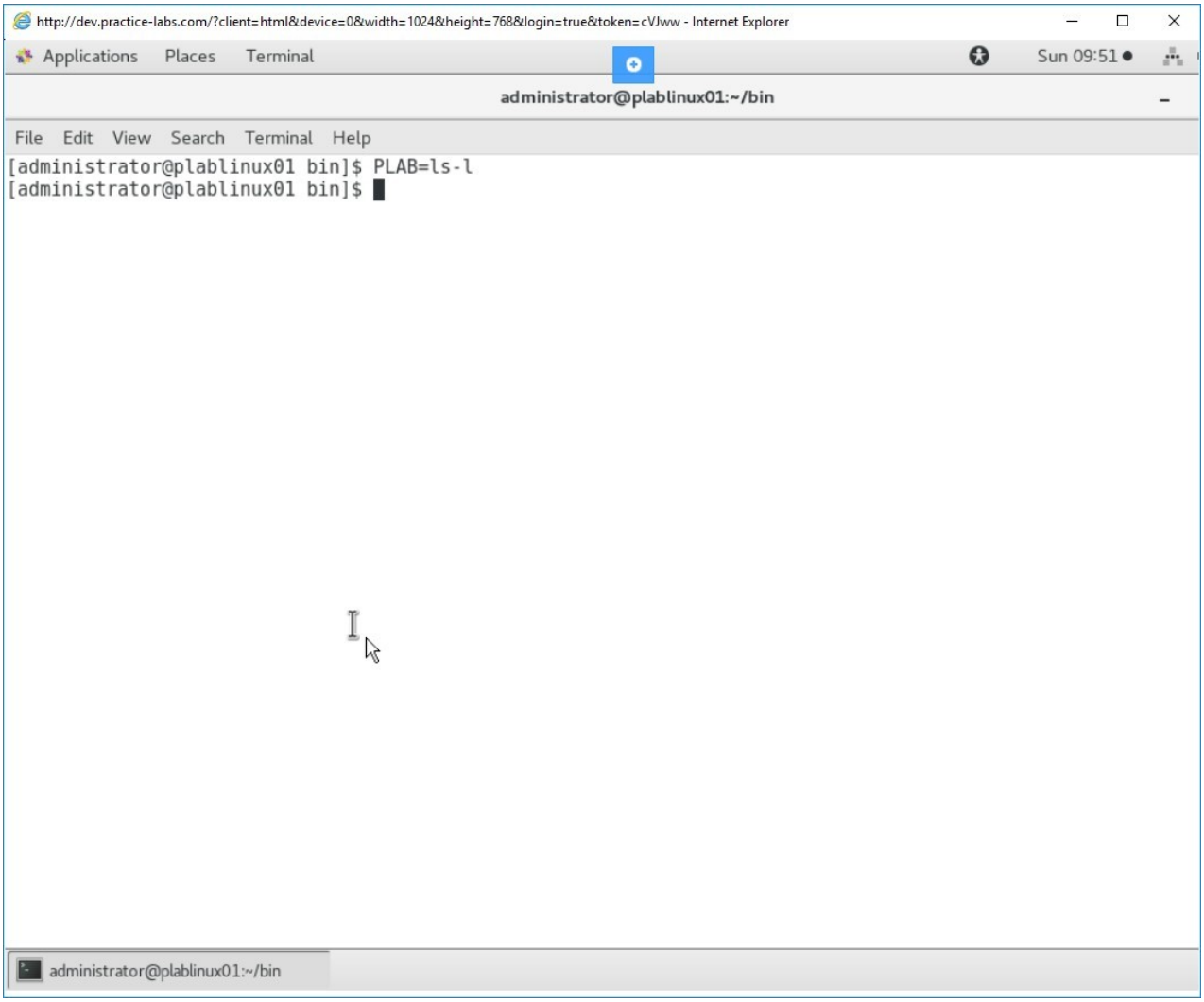


Figure 1.10 Screenshot of PLABLINUX01: Defining a local variable.

Step 2

To verify if the local variable is created, type the following command:

```
echo $PLAB
```

Press **Enter**. Notice the variable along with its value is returned in the output.

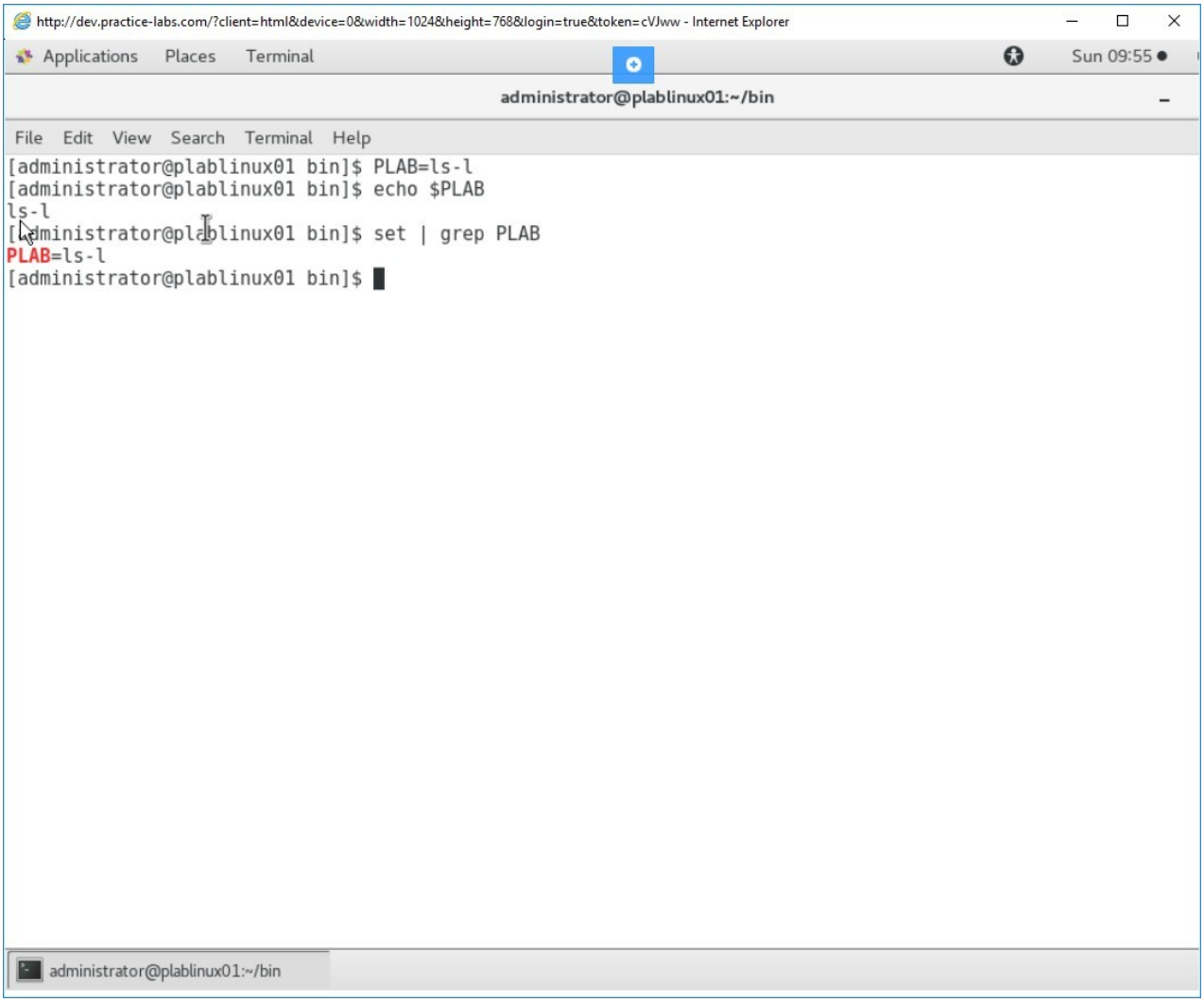


Figure 1.11 Screenshot of PLABLINUX01: Displaying the local variable value.

Step 3

To verify the existence of the variable with the set command, type the following command:

```
set | grep PLAB
```

Press **Enter**.

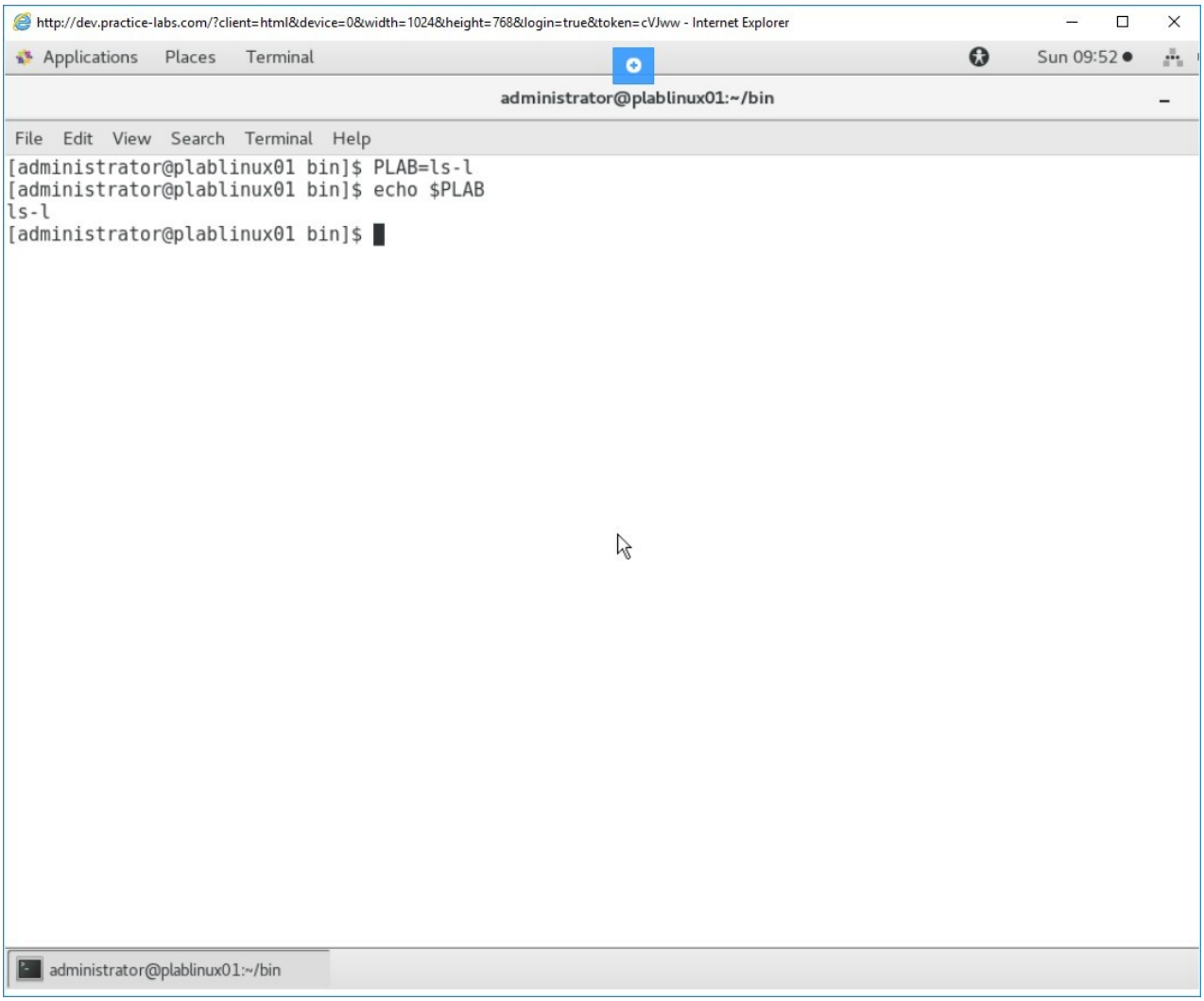


Figure 1.12 Screenshot of PLABLINUX01: Displaying the local variable value with the set and grep commands.

Step 4

To verify if the local variable is created with the printenv command, type the following command:

```
printenv | grep PLAB
```

Press **Enter**. Notice no output is returned.

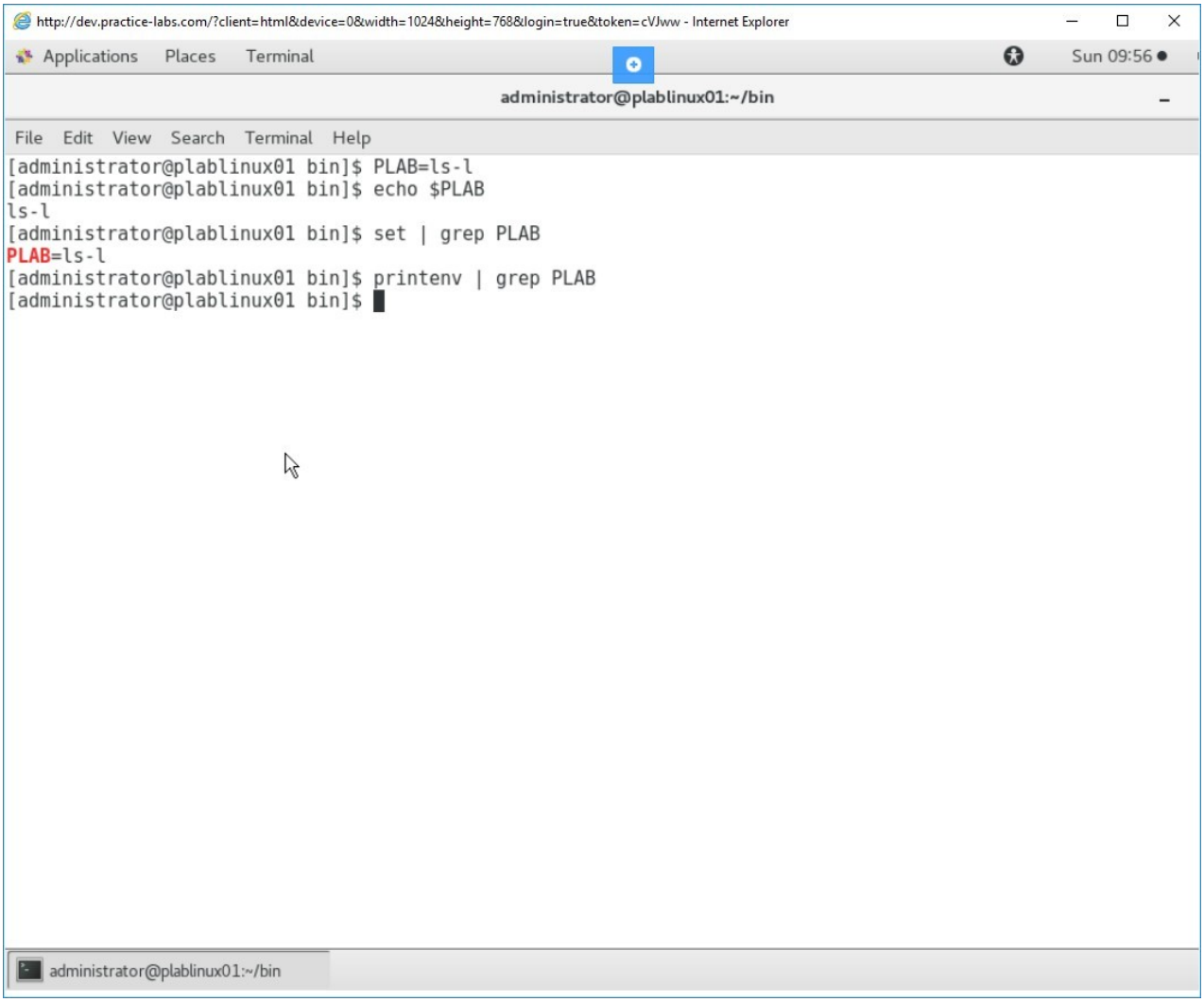


Figure 1.13 Screenshot of PLABLINUX01: Displaying the local variable value with the printenv and grep commands.

Step 5

Clear the screen by entering the following command:

```
clear
```

To make the local variable environment variable, type the following command:

```
export PLAB
```

Press **Enter**. Notice no output is returned.

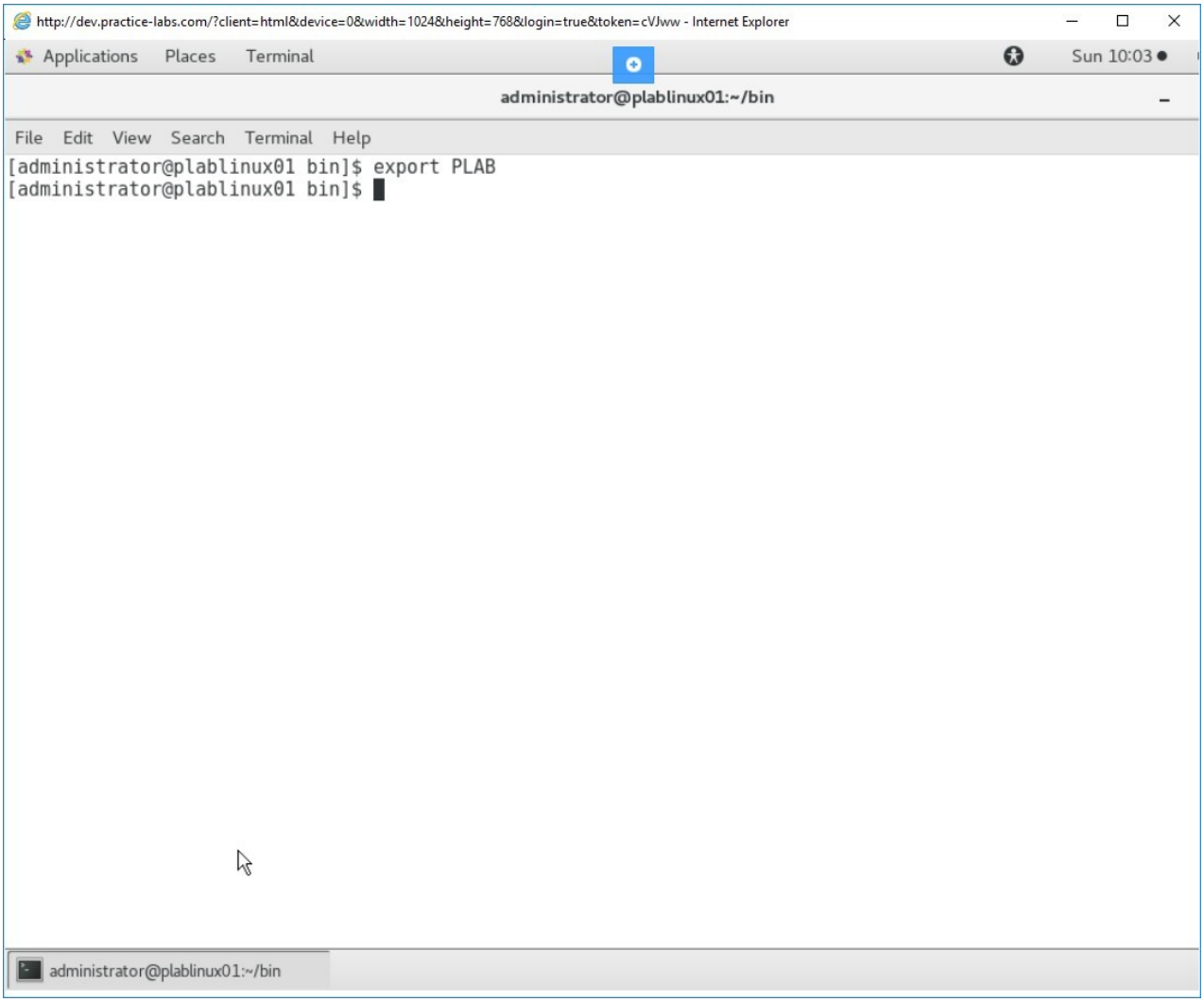


Figure 1.14 Screenshot of PLABLINUX01: Converting the local variable into an environment variable.

Step 6

To test if the variable has been exported as an environment variable, type the following command:

```
printenv | grep PLAB
```

Press **Enter**.

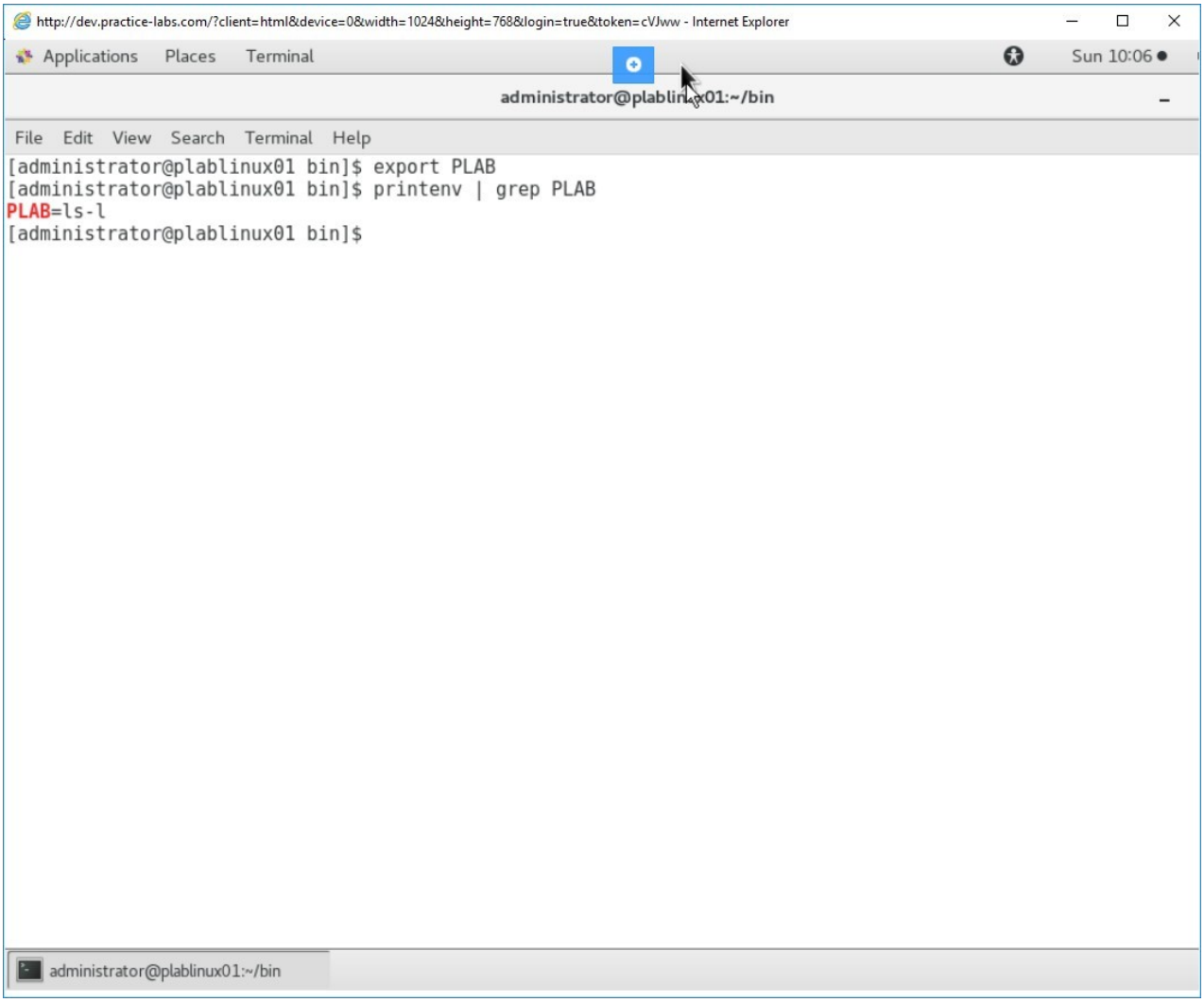


Figure 1.15 Screenshot of PLABLINUX01: Displaying the local variable value with the printenv and grep commands.

Step 7

Clear the screen by entering the following command:

```
clear
```

You can convert an environment variable to local variable. Type the following command:

```
export -n PLAB
```

Press **Enter**. Notice no output is returned.

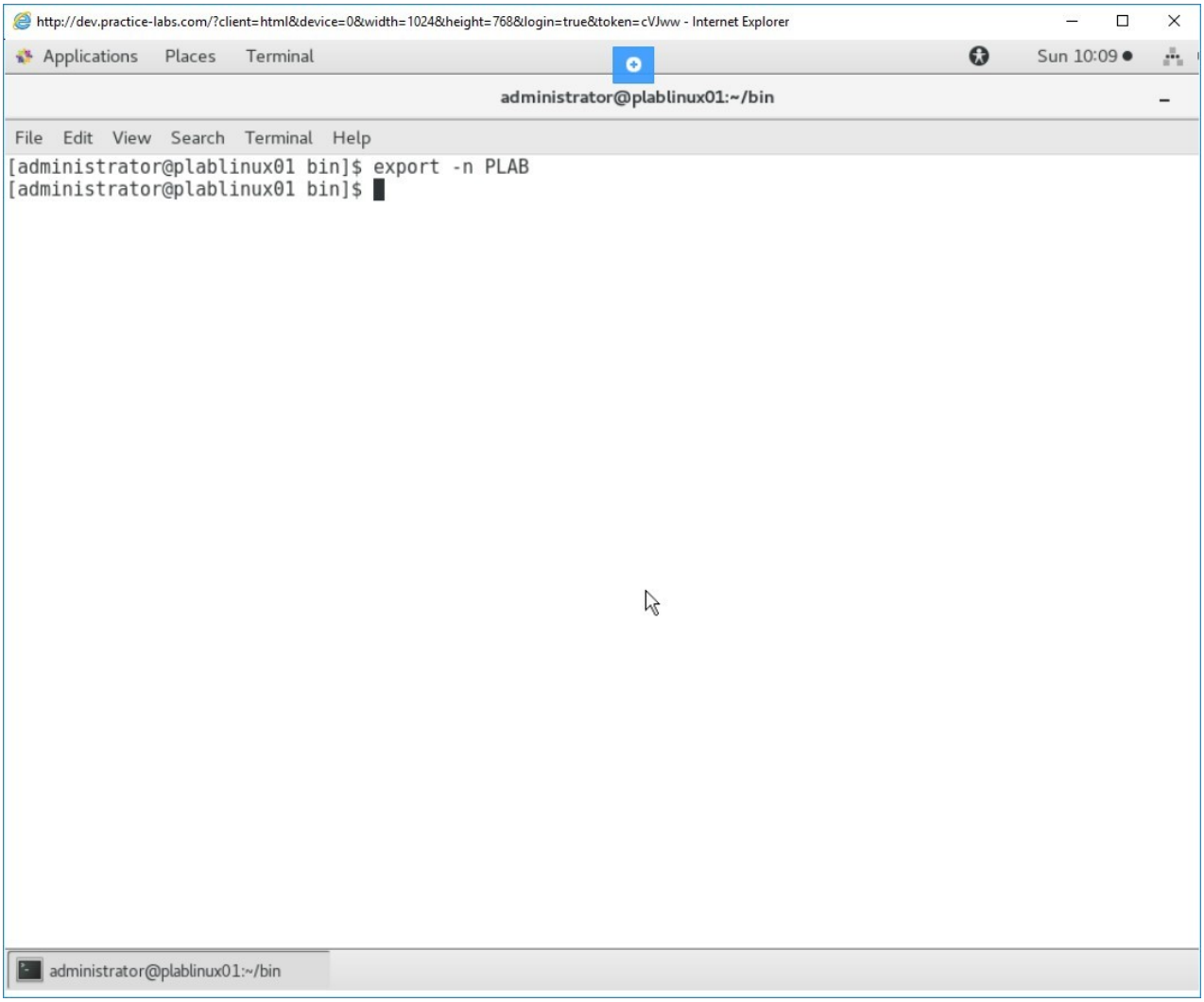


Figure 1.16 Screenshot of PLABLINUX01: Converting the environment variable to local variable.

Step 8

To test if the variable is still an environment variable, type the following command:

```
printenv | grep PLAB
```

Press **Enter**. Notice no output is returned.

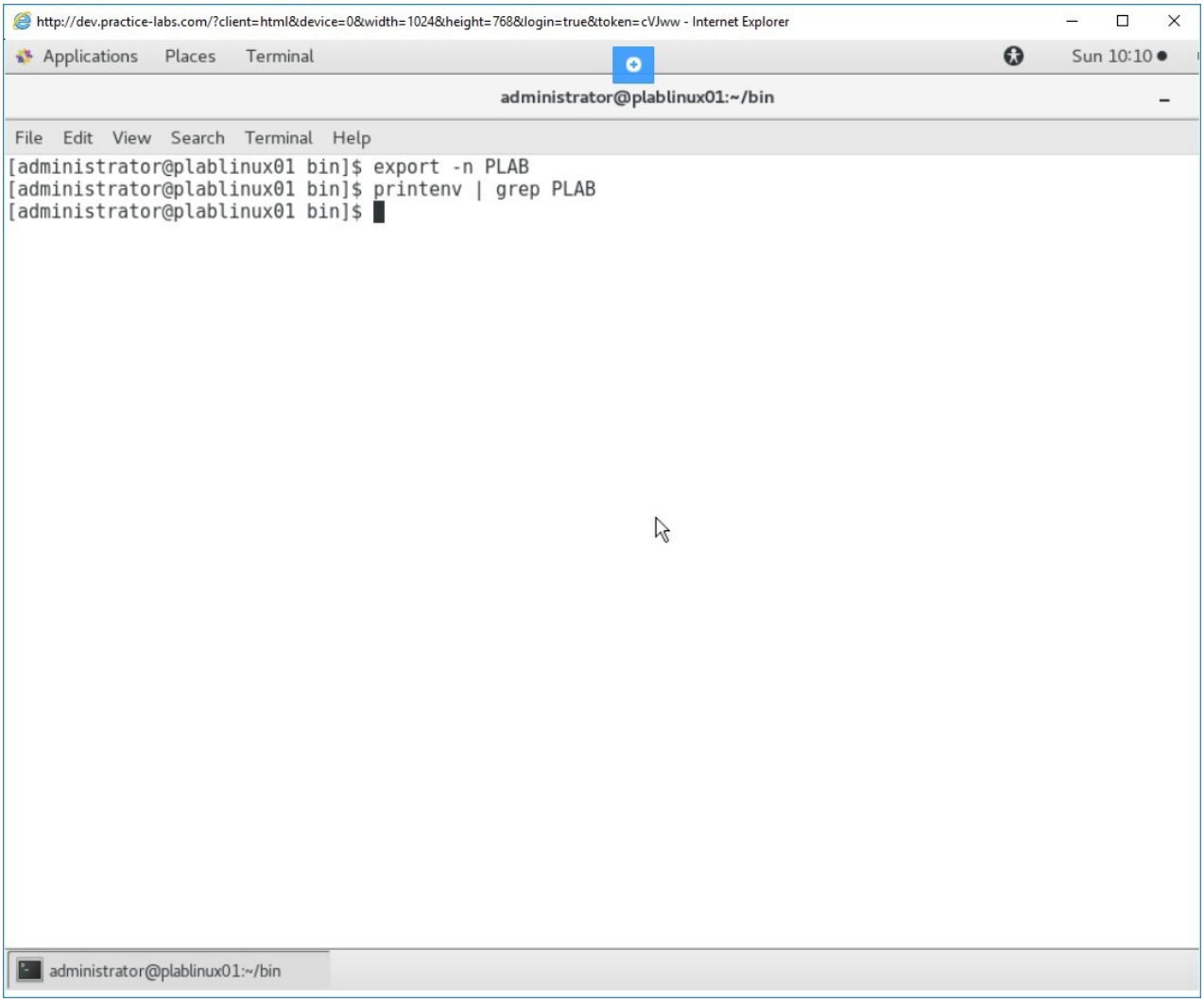


Figure 1.17 Screenshot of PLABLINUX01: Displaying the local variable value with the printenv and grep commands.

Step 9

To test if the variable is still a local variable, type the following command:

```
set | grep PLAB
```

Press **Enter**. Notice the local variable and its value is returned.

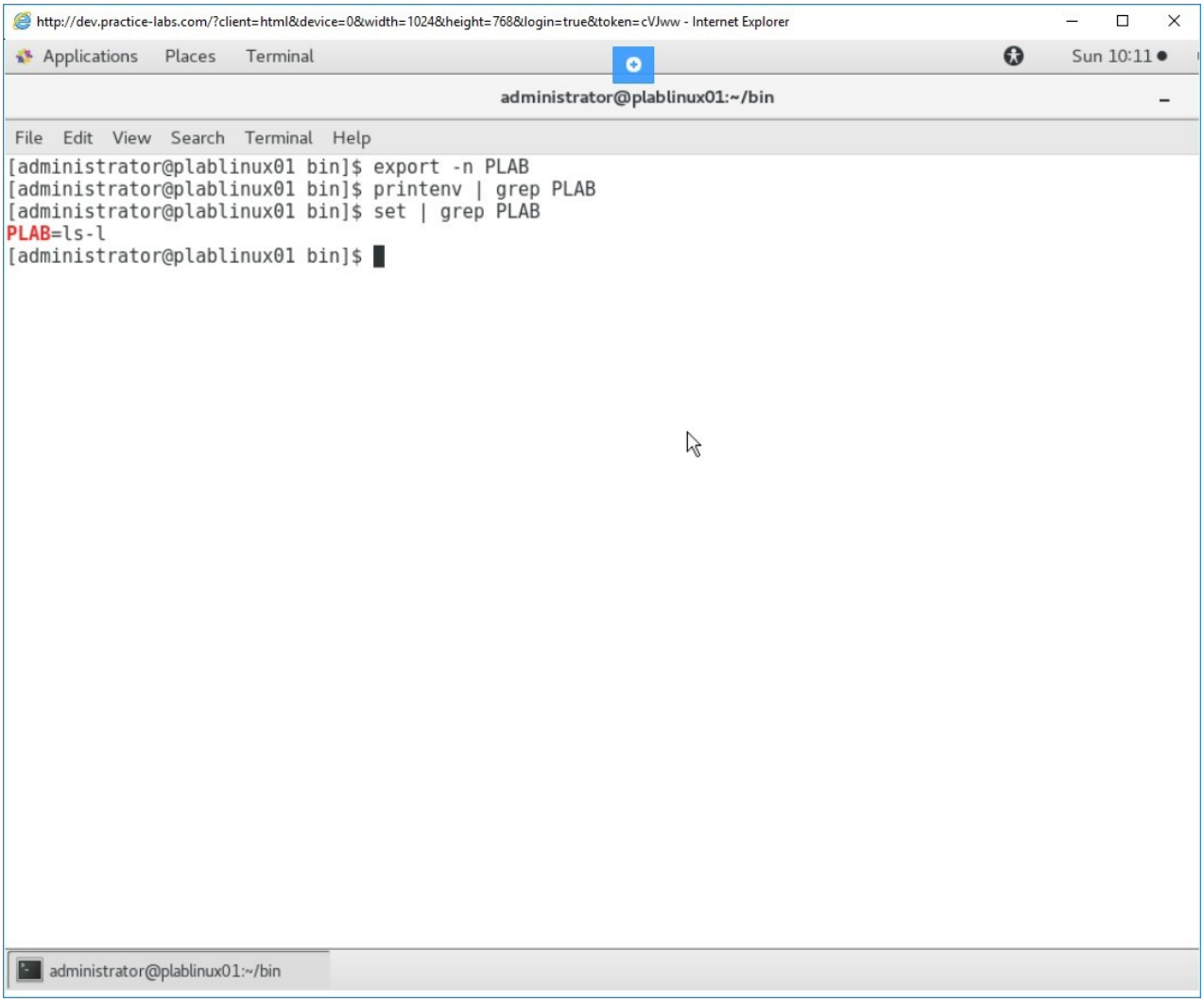


Figure 1.18 Screenshot of PLABLINUX01: Displaying the local variable value with the set and grep commands.

Step 10

You can unset a variable, either local or environment, using the unset command. Type the following command:

```
unset PLAB
```

Press **Enter**.

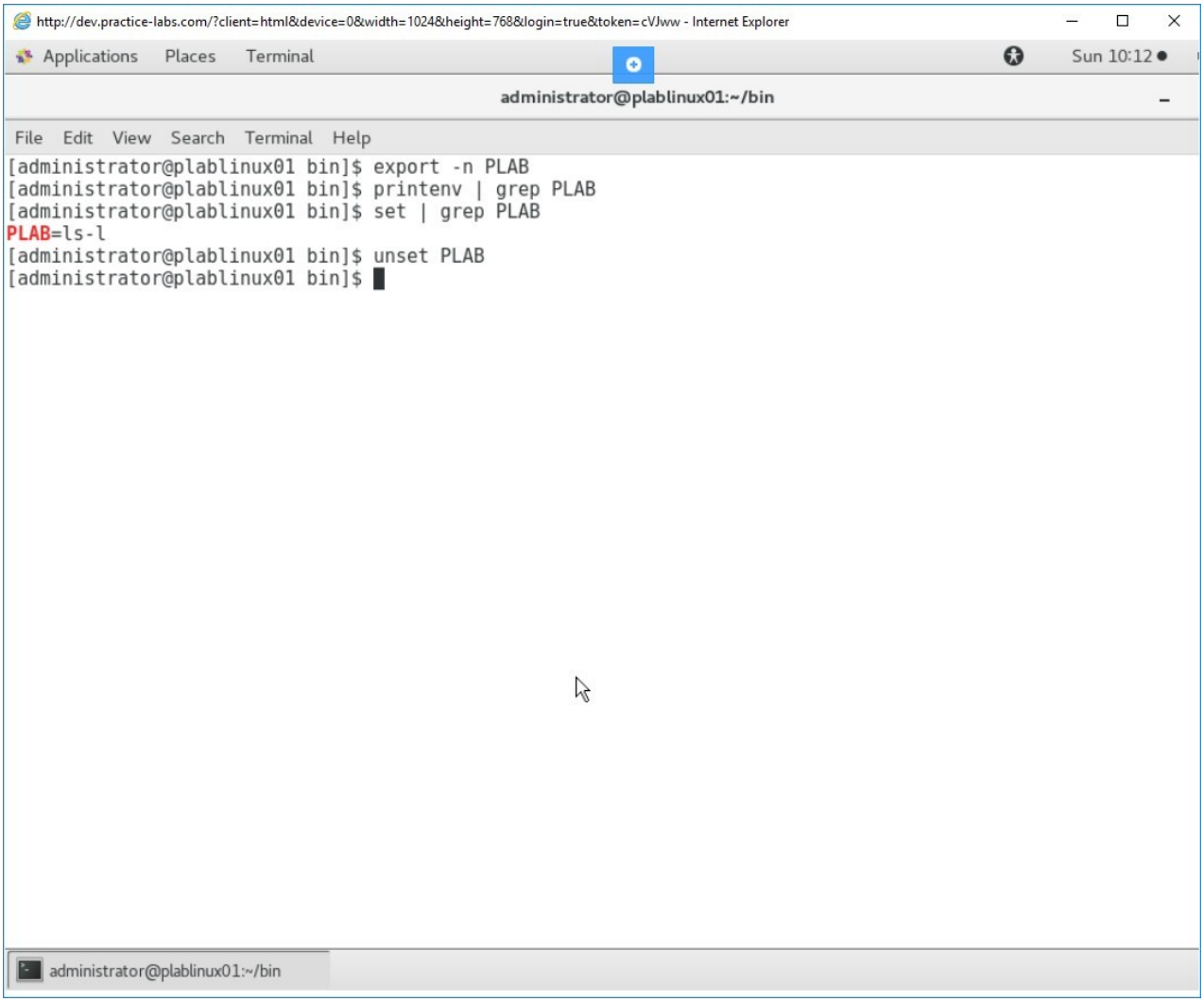


Figure 1.19 Screenshot of PLABLINUX01: Unsetting a variable.

Step 11

You can verify whether the variable has been unset. Type the following command:

```
echo $PLAB
```

Press **Enter**. Notice no value is now returned.

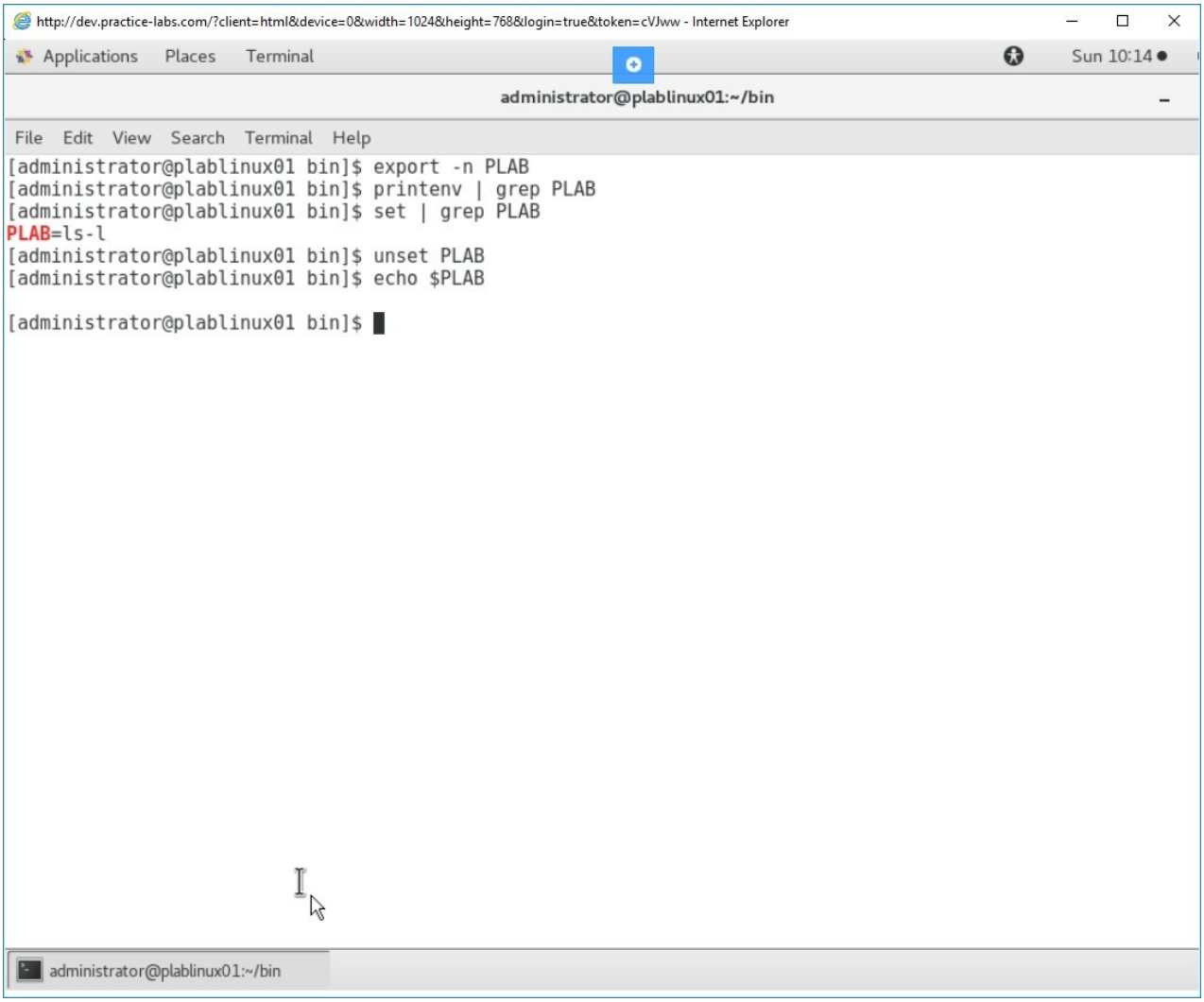


Figure 1.20 Screenshot of PLABLINUX01: Displaying the variable value.

Step 12

You can define a permanent environment variable in the **~/.bashrc** file. The variable defined here will be available for both login and non-login shells. Type the following command:

```
vi ~/.bashrc
```

Press **Enter**.

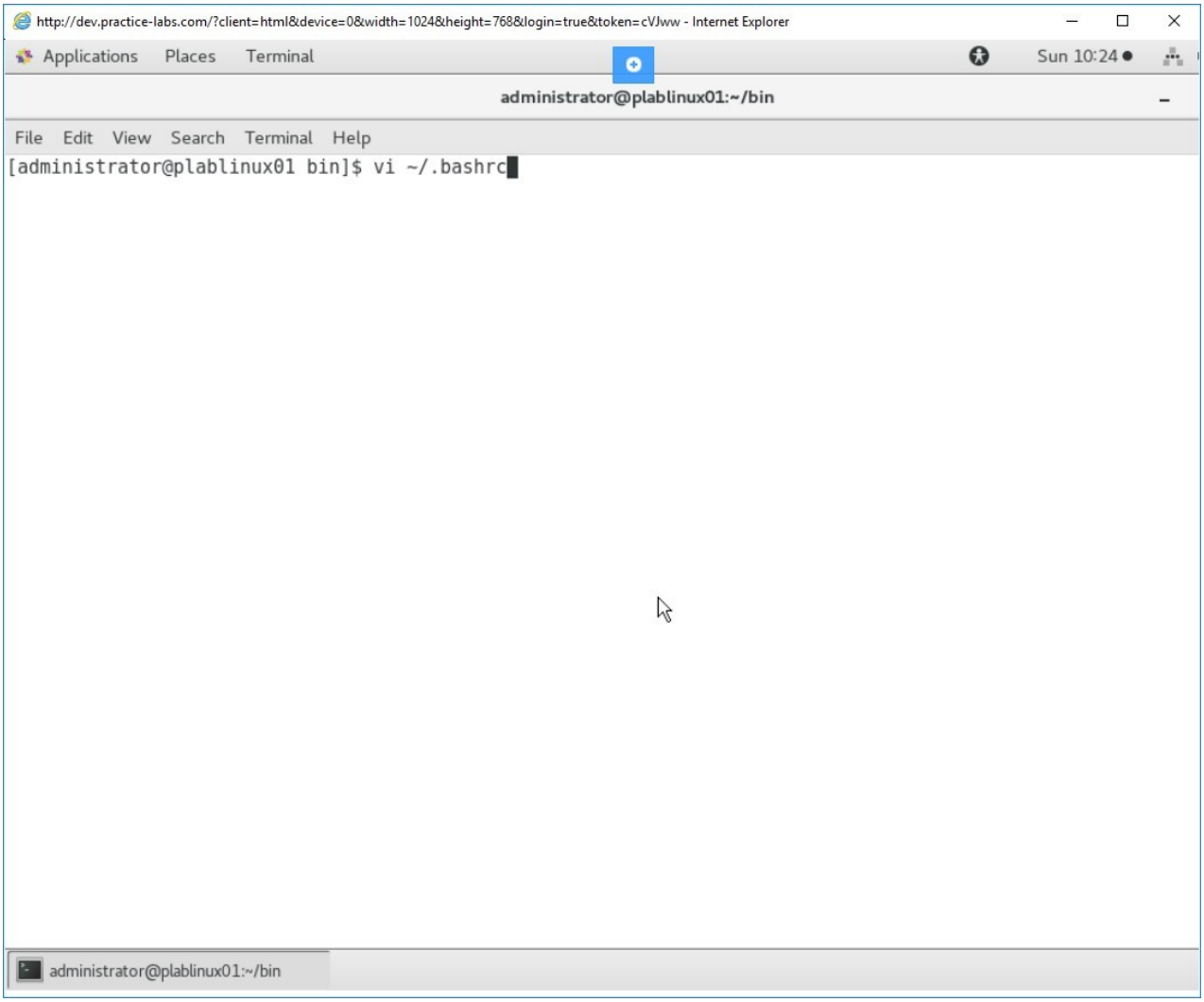


Figure 1.21 Screenshot of PLABLINUX01: Opening the ~/.bashrc file using the vi editor.

Step 13

Scroll down to an empty line. Press i to get into the insert mode and add the following line:

```
export PLAB=ls-l
```

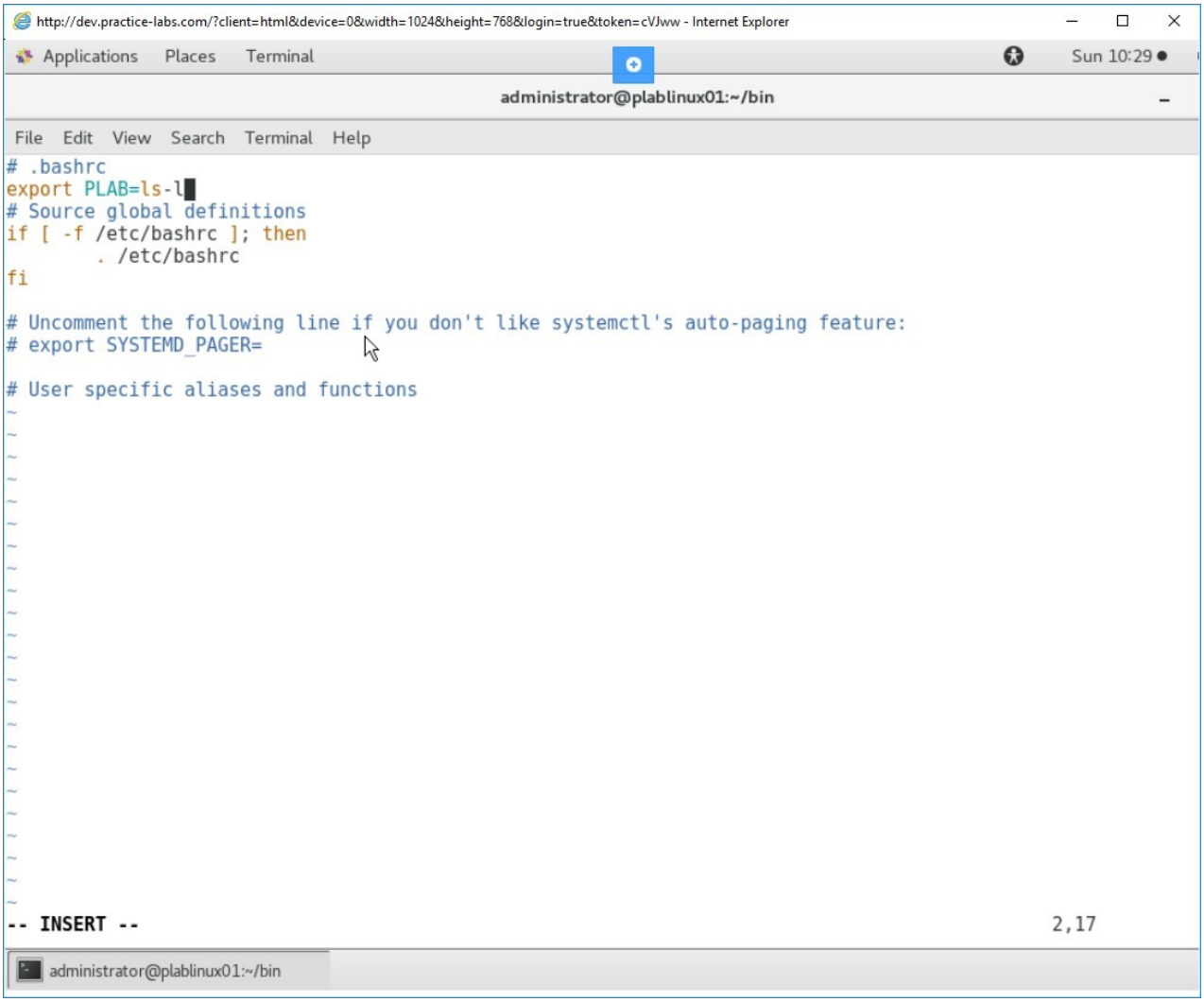


Figure 1.22 Screenshot of PLABLINUX01: Adding an environment variable.

Step 14

Press **ESC** and type the following command:

```
:wq
```

Press **Enter**.

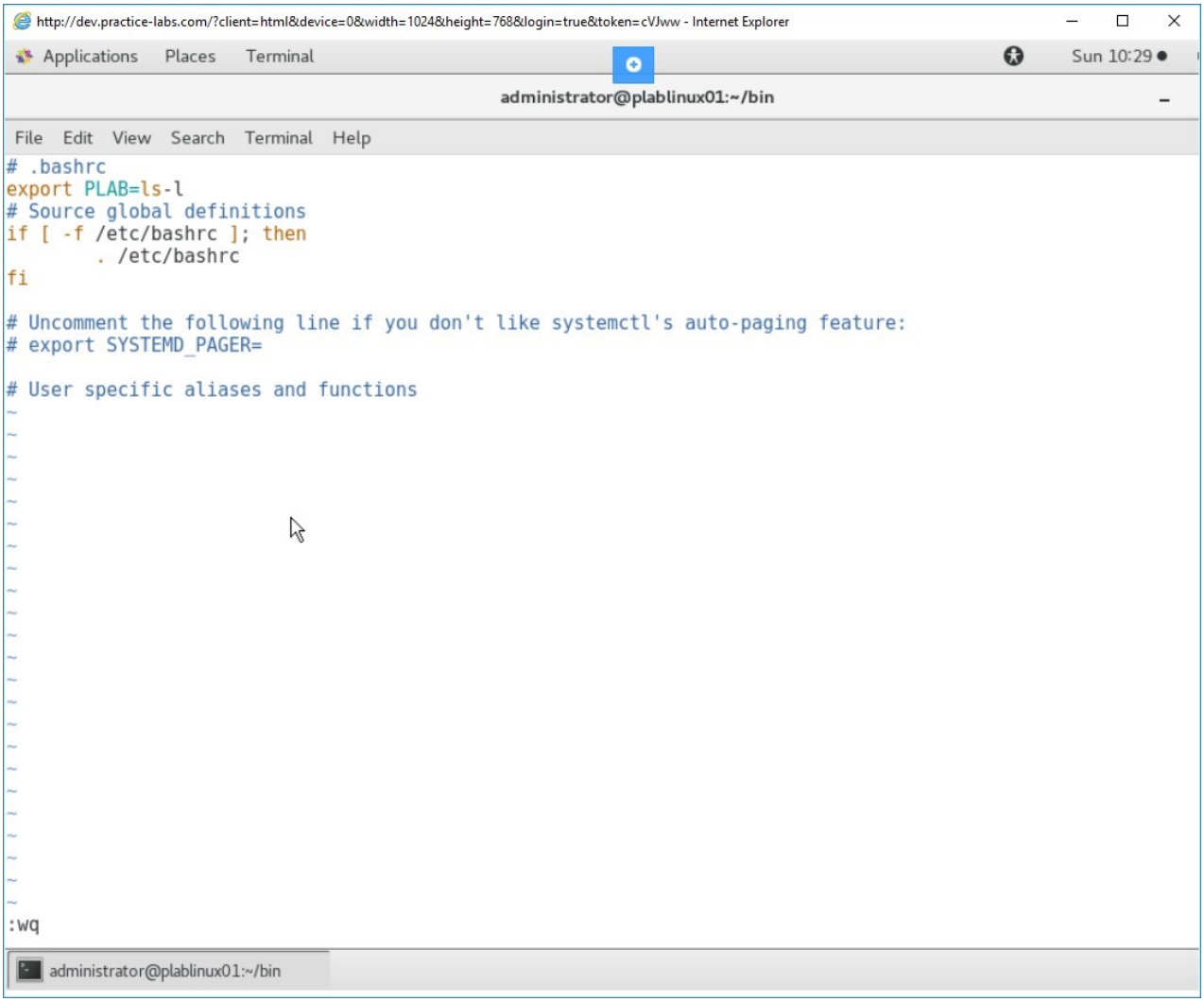


Figure 1.23 Screenshot of PLABLINUX01: Saving and exiting the file.

Step 15

The declared variable will be available in the next shell session. However, if you need the variable to be available in the current session, type the following command:

```
source ~/.bashrc
```

Press **Enter**. Notice no output is returned.

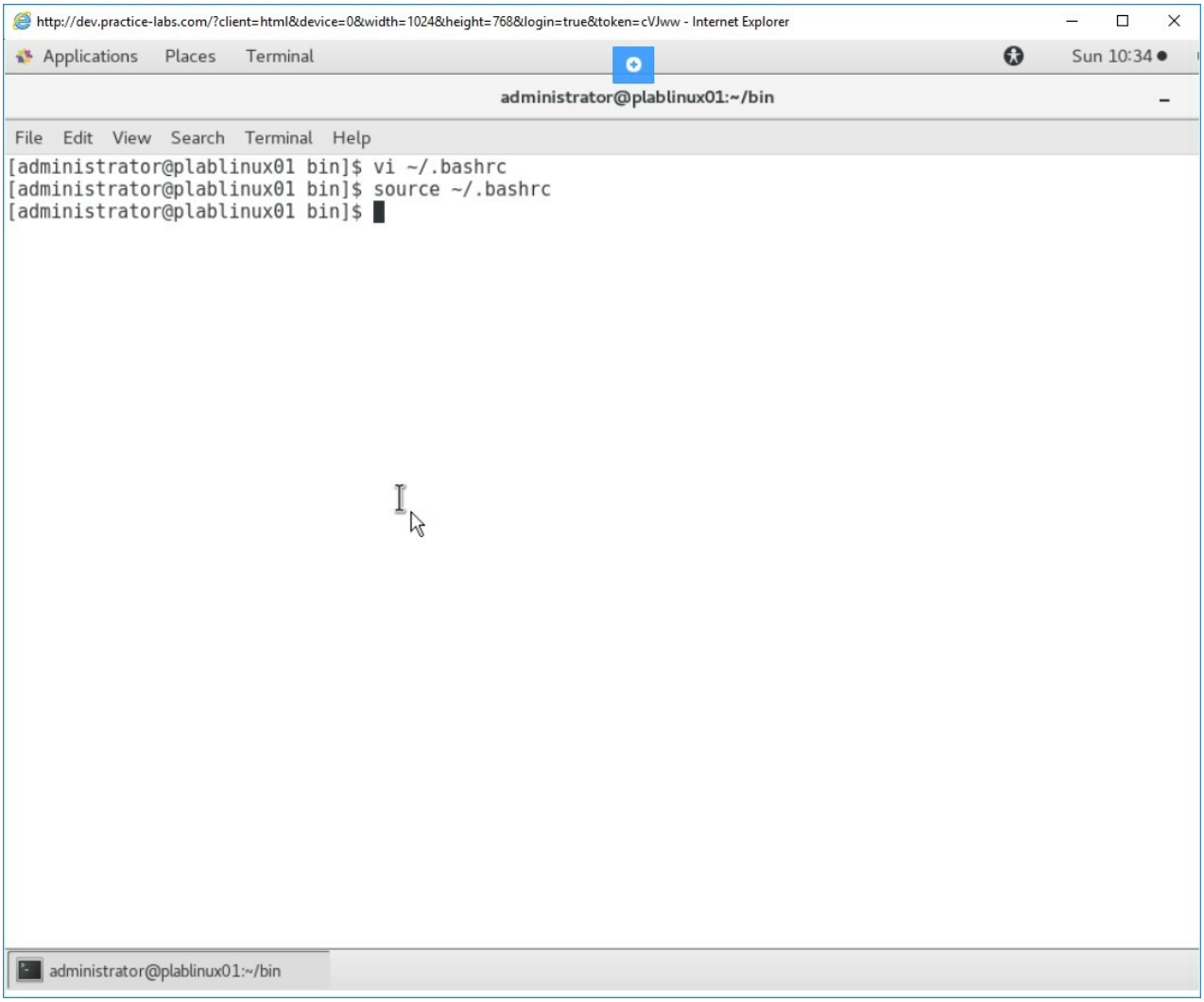


Figure 1.24 Screenshot of PLABLINUX01: Loading the ~/.bashrc file in the current session.

Step 16

To test if the variable is available as an environment variable, type the following command:

```
printenv | grep PLAB
```

Press **Enter**. Notice the variable and the value is returned.

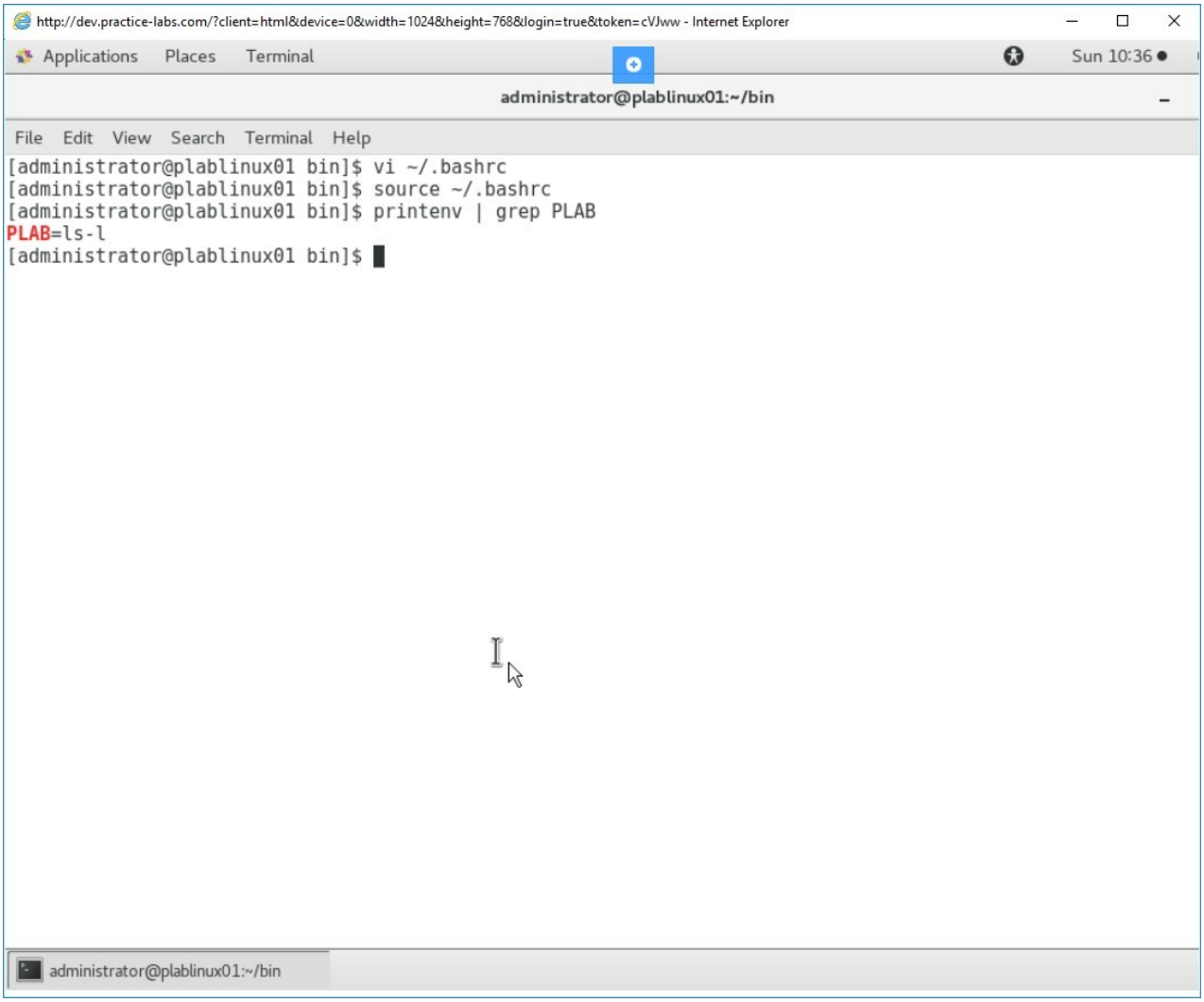


Figure 1.25 Screenshot of PLABLINUX01: Displaying the local variable value with the printenv and grep commands.

Keep all devices in their current state and proceed to the next exercise.

Review

Well done, you have completed the **Working with the Environment Variables** Practice Lab.

Summary

You completed the following exercise:

- Exercise 1 - Working with the Environment Variables

You should now be able to:

- Read environment variables
- Set environment variables including the permanent ones

Feedback

Shutdown all virtual machines used in this lab. Alternatively, you can log out of the lab platform.